Audio Calptioning

Alperen Ozcelik [1] **Yunus Alper Bagcilar** [1]

Abstract

Audio captioning is focusing on identifying the human-perceived information in a general audio signal and expressing it through text, using natural language. This information includes identification of sound events, acoustic scenes, spatiotemporal relationships of sources, foreground versus background discrimination, concepts, and physical properties of objects and environment. In this project, we aimed to design a CNN-based model for audio files feature extraction and a transformer-based NLP model for caption generation. We evaluate our model on Clotho dataset's test set, which has approximately 1,000 audio files, along with 5 captions for each audio file.

1. Introduction

For us, audio signals are one of the most important sources to receive information from the outside world in our daily life. Automated audio captioning is a multimodal problem in which the model takes audio signals as inputs and gives a natural language description of that audio signal. Understanding only the acoustic events isn't enough to solve the audio captioning problem. The audio captioning models also have to understand natural language and some other pieces of information from the audio signal such as discrimination of foreground from the background (e.g. Cars driving by outside with heavy rain falling), space-time relationships of audio sources (e.g. A car with a siren passes by and then a bicycle passes by), and physical information of objects and environment (e.g. Droplets of water are falling on a metallic surface). Therefore, audio captioning can be useful in a variety of areas, such as helping the hearing impaired understand sounds and sound-based security systems.

Like image captioning, audio captioning aims to extract features from input space and transfer those features into natural language space. To do this, effective feature extraction and natural language modeling are required. Compared to audio tagging and image captioning, limited data is available for audio captioning, so it may not be sufficient to do the whole process directly with audio captioning data. However, pre-trained models can be used in audio captioning as well as in image captioning. These pre-trained models are usually Convolutional Neural Network (CNN) based models trained on large datasets.

While image captioning and video captioning are areas that have been studied for a while, audio captioning is a topic that has been started to be studied much more recently. With the participation of the audio captioning challenge in DCASE 2020 and 2021, the interest in this subject has increased and various methods have been proposed, some of which we will talk about in this paper. On the other hand, the audio caption issue is vague compared to image captioning, and voice recognition is subjective even for different people, making it difficult to determine the best match.

The audio captioning problem is on an encoder-decoder architecture, where the decoder generates captions according to the audio features extracted by the encoder. In previous studies, the "RNN-RNN" architecture was generally adopted. However, RNNs can be limited in modeling long-term temporal dependencies in an audio signal. Recently, CNN is used as the encoder to extract image features and build a semantic vector, and many researchers using pre-trained CNNs as the audio encoder which significantly improved the performance in these systems and uses RNN as the decoder to transform the vector into a sequenced words list. In some of the work, we can see that transformer-based decoders can be used for this problem as well. So, we use a transformer for our decoder model.

2. Related Work

In the past studies, deep learning approaches based on encoder-decoder architecture have been presented for the automated audio captioning problem. The first approach to Automated Audio Captioning was proposed by Drossos who used an encoder-decoder-based RNN model(Drossos et al., 2017). Interest in automated audio captioning has grown with the release of two new freely available datasets Clotho(Drossos et al., 2020), AudioCaps(Kim et al., 2019).

[1]Department of Computer Engineering, Hacettepe University, Ankara, Turkey. Correspondence to: Alperen Ozcelik <alperenozcelik@hacettepe.edu.tr>, Yunus Alper Bagcilar <alper.bagcilar@hacettepe.edu.tr>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Some researchers replaced the RNN, which was used as an encoder, with CNN, and this gave a significant performance increase(Xu et al., 2021; Chen et al., 2020). M. Wu used the output from the encoder directly in the decoder, but in this way, the acoustic information could not be used at its full potential(Wu et al., 2019). H. Wang proposed a decoder with a temporal attention mechanism that can use more acoustic information for each time step(Wang et al., 2020). Both of these two studies used approaches that attempt to directly train the entire audio captioning model but due to lack of data, the encoder could not learn the representations of the audio signals in both studies. In another study, Y. Wu created a label pool of 300 words with the highest frequency and developed a pre-training strategy that includes multi-label classification(Chen et al., 2020). Then they trained a CNN encoder with these 300 labels. However, since these labels were created based on frequency, they also included some useless terms, for example: until, onto. After the classification, they used a transformer as the decoder. In some studies, we have seen the use of pre-trained audio neural networks (PANNs)(Kong et al., 2020). Although these pre-trained networks are networks trained on very large data sets, they are mostly used for audio tagging and sound event detection and directly take raw audio files as input. However, instead of using raw audio files directly as input in our project, we want to use the spectrogram images of these audio files. At the same time, since we do not want to use these PANNs directly, we aim to create our own pre-trained model. We also saw that there is a transformer model called T5(Text-To-Text Transfer Transformer) which can be used in multiple NLP tasks(Raffel et al., 2019). A modified version of T5 is KeyToText which is suitable to be our decoder(Bhatia).

3. The Approach

3.1. Dataset

The dataset named as Clotho we will use is hosted at zenodo.org. Clotho is an audio caption dataset of 4,981 audio samples with five captions of each audio file. Captions are 8 to 20 words and audio files are 15 to 30 seconds long.

Clotho is divided into three parts: 60% in the development, 20% in the evaluation, and 20% in the testing part. Also, in Clotho, there is not a word that appears only in one part. In dataset, there are many sounds that we hear in daily life, such as the sounds of nature, animal sounds, people's speech, the sounds of machines, and the sounds of various products in the house.

Although the main dataset we use and will use for testing is Clotho, we need to mention AudioSet as well. Because we also use AudioSet in our project, you will understand the reason more clearly in the rest of the report.

AudioSet is also one of the well-known datasets that can be used for audio classification and audio captioning like Clotho.

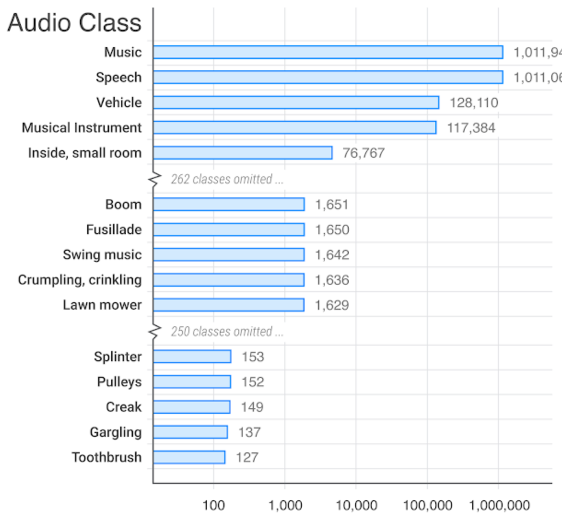


Figure 1. Histogram of the number of examples per class in AudioSet

3.2. Mel-Spectrogram

Leveraging mel-spectrograms is one of the ways to process sound used in a variety of Deep Learning and Machine Learning problems.

We decided to use mel-spectrograms as a feature in our model. Before moving on to the reasons for this, it would be better to talk about what spectrogram and mel scale are.

The mel scale is a perceptual pitch scale judged to be equidistant by listeners. The reference of 1000 mels was assigned as having a frequency of 1000 Hz, 40 dB above the threshold.

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

Figure 2. Formula to convert f hertz into m mels

The spectrogram is a visual way of representing the signal strength, or "loudness", of a audio signal over time at various frequencies present in a particular waveform. Thanks to spectrograms, we can clearly see how energy levels change over time.

A mel-spectrogram is a spectrogram where the frequencies are converted to the **mel scale**. In spectrogram graphs the x-axis representing time, the y-axis representing the frequency which is in log scale, and the z-axis representing the amplitude.

We saw that the mel-spectrogram graphs were used in the studies conducted in several articles we reviewed on audio captioning, and this situation was very interesting to us. At the same time, although we have not been working on sound files until now, we thought that it would be more suitable for us because we were doing projects related to image processing and we decided to proceed on this path.

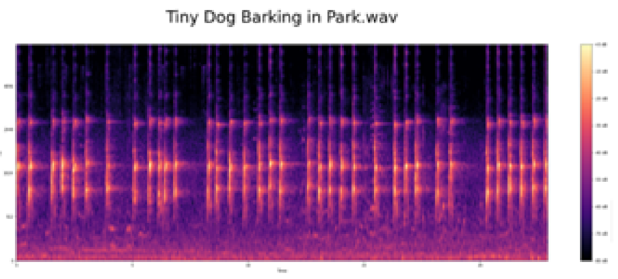


Figure 3. Example of mel-spectrogram graph

3.3. CNN Model

Convolutional Neural Network (CNN) is a Deep Learning algorithm that takes pictures as input, attaches importance to various aspects of the picture, and can distinguish them from each other. When using CNN, the required pre-processing phase is shorter than other classification algorithms. In primitive methods, the filters are determined by the person writing the code, while CNN has the ability to learn by itself.

The mel-spectrogram is a useful technique for extracting hidden audio features and visualizing them as an image. A CNN model is capable of extracting information from images and then performing tasks like classification and captioning. As a result, we decided to use the CNN model as an encoder.

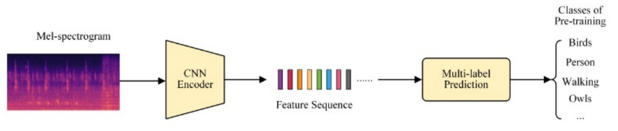


Figure 4. Encoder Task

So, the problem has now turned into an image processing problem as we wanted.

3.4. KeyToText Model

In our project, we used the KeyToText model as our decoder. This model is based on the T5(Text-To-Text Transfer Transformer) model. T5 is created by Google for multiple NLP tasks such as machine translation, document summarization, question answering, and classification tasks. T5 can also solve regression problems if we use string representations of numbers instead of numbers.

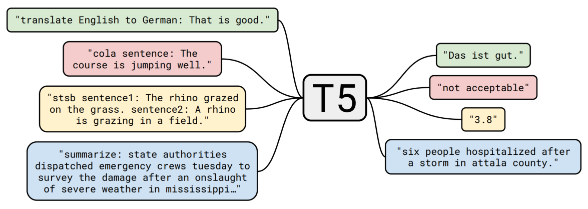
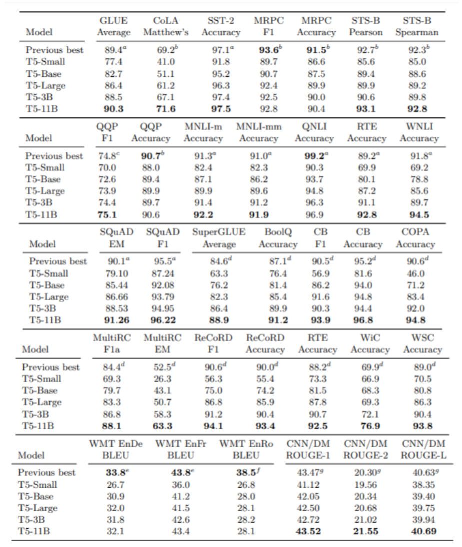

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 5. T5 Diagram

T5 is pre-trained on the C4(Colossal Clean Crawled Corpus) dataset which is also provided by Google. Using the T5 text-to-text framework and C4 pre-training dataset they did some experiments on model architectures, pre-training objectives, unlabeled datasets, training strategies, and scale.

They wanted to reach the limits of transfer learning on NLP so they did another set of experiments with the best methods from the previous experiments combined.



Model	GLUE Average	CoLA Matthew's	SST-2 Accuracy	MRPC F1	MRPC Accuracy	STS-B Pearson	STS-B Spearman
Previous best	89.4[a]	69.2[b]	97.1[a]	**93.6**[b]	**91.5**[b]	92.7[b]	92.3[b]
T5-Small	77.4	41.0	91.8	89.7	86.6	85.6	85.0
T5-Base	82.7	51.1	95.2	90.7	87.5	89.4	88.6
T5-Large	86.4	61.2	96.3	92.4	89.9	89.9	89.2
T5-3B	88.5	67.1	97.4	92.5	90.0	90.6	89.8
T5-11B	**90.3**	**71.6**	**97.5**	92.8	90.4	**93.1**	**92.8**

Model	QQP F1	QQP Accuracy	MNLI-m Accuracy	MNLI-mm Accuracy	QNLI Accuracy	RTE Accuracy	WNLI Accuracy
Previous best	74.8[c]	**90.7**[b]	91.3[a]	91.0[a]	**99.2**[a]	89.2[a]	91.8[a]
T5-Small	70.0	88.0	82.4	82.3	90.3	69.9	69.2
T5-Base	72.6	89.4	87.1	86.2	93.7	80.1	78.8
T5-Large	73.9	89.9	89.9	89.6	94.8	87.2	85.6
T5-3B	74.4	89.7	91.4	91.2	96.3	91.1	89.7
T5-11B	**75.1**	90.6	**92.2**	**91.9**	96.9	**92.8**	**94.5**

Model	SQuAD EM	SQuAD F1	SuperGLUE Average	BoolQ Accuracy	CB F1	CB Accuracy	COPA Accuracy
Previous best	90.1[a]	95.5[a]	84.6[d]	87.1[d]	90.5[d]	95.2[d]	90.6[d]
T5-Small	79.10	87.24	63.3	76.4	56.9	81.6	46.0
T5-Base	85.44	92.08	76.2	81.4	86.2	94.0	71.2
T5-Large	86.66	93.79	82.3	85.4	91.6	94.8	83.4
T5-3B	88.53	94.95	86.4	89.9	90.3	94.4	92.0
T5-11B	**91.26**	**96.22**	**88.9**	**91.2**	**93.9**	**96.8**	**94.8**

Model	MultiRC F1a	MultiRC EM	ReCoRD F1	ReCoRD Accuracy	RTE Accuracy	WiC Accuracy	WSC Accuracy
Previous best	84.4[d]	52.5[d]	90.6[d]	90.0[d]	88.2[d]	69.9[d]	89.0[d]
T5-Small	69.3	26.3	56.3	55.4	73.3	66.9	70.5
T5-Base	79.7	43.1	75.0	74.2	81.5	68.3	80.8
T5-Large	83.3	50.7	86.8	85.9	87.8	69.3	86.3
T5-3B	86.8	58.3	91.2	90.4	90.7	72.1	90.4
T5-11B	**88.1**	**63.3**	**94.1**	**93.4**	**92.5**	**76.9**	**93.8**

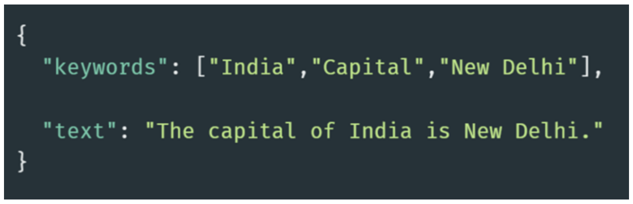
Model	WMT EnDe BLEU	WMT EnFr BLEU	WMT EnRo BLEU	CNN/DM ROUGE-1	CNN/DM ROUGE-2	CNN/DM ROUGE-L
Previous best	**33.8**[e]	**43.8**[e]	**38.5**[f]	43.47[g]	20.30[g]	40.63[g]
T5-Small	26.7	36.0	26.8	41.12	19.56	38.35
T5-Base	30.9	41.2	28.0	42.05	20.34	39.40
T5-Large	32.0	41.5	28.1	42.50	20.68	39.75
T5-3B	31.8	42.6	28.2	42.72	21.02	39.94
T5-11B	32.1	43.4	28.1	**43.52**	**21.55**	**40.69**

Figure 6. T5 Scores (Different variants of T5 and in the first row of each table state-of-the-art for the task)

Their largest model had 11 billion parameters and got the best scores on GLUE, SuperGLUE, SQuAD, and

CNN/Daily Mail benchmarks. So we can say that it achieved the state-of-art.

T5 is a flexible model which can be modified for many tasks. One of these modified versions is KeyToText. This model is a fine-tuned T5 on the CommonGen dataset. KeyToText takes a set of keywords as input and returns a sentence using these keywords as output. So it was the perfect fit for our project's decoder part.



```
{
  "keywords": ["India","Capital","New Delhi"],

  "text": "The capital of India is New Delhi."
}
```

Figure 7. Example of KeyToText

Thus, we have decided on the entire design architecture of our project.

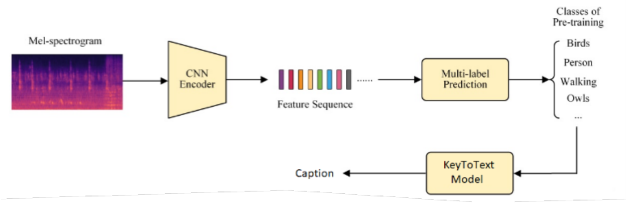


Figure 8. Final Architecture

4. Experimental Results

After we decided on the feature and model we would use, we had to decide what kind of output we would get from the model.

In an article we researched on this subject, we saw that audio captioning was turned into a multi-label classification task by taking the keywords in the sentences and using the 300 keywords with the highest frequency among all keywords(Chen et al., 2020).

4.1. Train with Clotho Dataset

We applied the same process to our own dataset. First, we put the spectrograms we obtained using the train part of the Clotho dataset into three CNN models consisting of 4, 5, and 8 layers in total, and we got an output with 300 labels and fit the dataset we separated as train and validation and try our models.

Afterward, we tried to get a slightly higher result by applying hyperparameter tuning to the 8-Layer CNN model, where we got the best results.

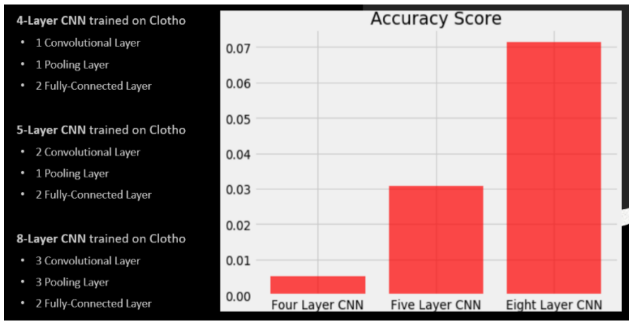


Figure 9. Scores of Different Model Designs (Trained on Clotho)

We plot the accuracy scores given by our CNN model as heatmaps in the best momentum and learning rate value range. We also made some other hyper-parameter tunings; this was the best accuracy score we got but the results were not very encouraging.

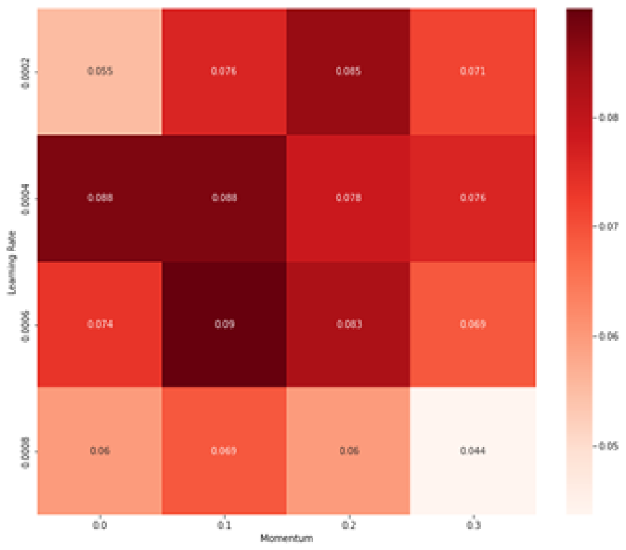


Figure 10. Hyperparameter Tuning Scores of the Best Initial Model

We read the previous works over and over for about five days. And we found that most of the time they don't use Clotho for every transaction. They generally used pre-trained CNN models that were already available. Then we reviewed several pre-trained CNN models for audio captioning, audio classification, and sound-event detection. After some more research, we came across a model called PANN (Pre-Trained Audio Neural Network). This model is a pre-trained model that can make classification by using spectrograms as we want.

Although we decided that it would be a better solution to try to create our own pre-trained model instead of using it directly, we thought that we can also use this model as a backup solution.

4.2. Train with AudioSet Dataset

To create our pre-trained model, we started collecting data from the AudioSet, which we were mentioned about. It consists of over 2 million audio files with over 500 labels. We can't use all the data because of some limitations such as time and system specs. Luckily the AudioSet has three parts: Balanced Train (over 20 thousand samples), Evaluation (over 20 thousand samples), and Unbalanced Train (approximately 2 million samples). So, we decided to use Balanced train and Evaluation for our pre-training purposes, but we can't directly access the raw audio files, we need to download them from their YouTube ID, crop them according to starting-ending times (given in the dataset), and transform them to spectrogram images for our project. It is really long process.

We collected audio samples from the AudioSet, simultaneously using 2 computers for about 10 days, and stored the mel-spectrograms of these samples in .png format as we did for Clotho. At the end of this 10-day period, we have stored 18875 available samples of the 22160 samples in the balanced train segment of the AudioSet.

We created three different CNN models consisting of 6, 10 and 14 layers in total, to train using these 18875 samples. After using different parameters (number of layers, layer sizes, learning rate, momentum) we decided on a model.

There are classification accuracy scores obtained from these models. We selected the model with highest score for prediction and used the predicted keywords in our decoder to get captions.

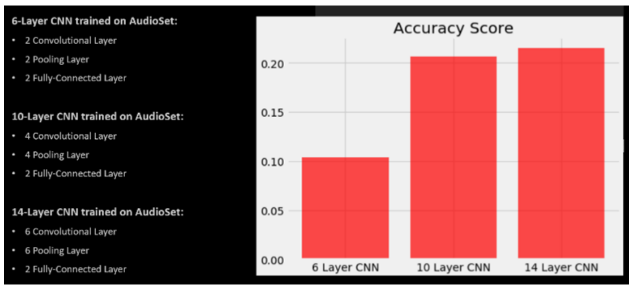


Figure 11. Scores of Different Model Designs (Trained on AudioSet)

4.3. PANN Model

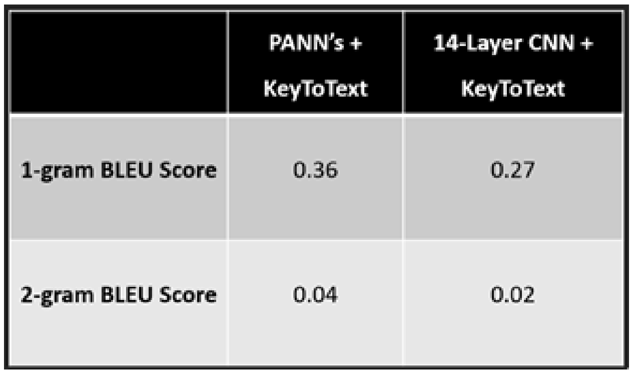
Even though we created our own pre-trained model, we still decided to make an experiment using the PANN model that we mentioned before, and we tried the results we obtained from the PANN model in the decoder section and got a score.

4.4. Scores

While collecting data for our own pre-trained model, we started to research the decoder model we will use after classification.

As a decoder, we would like to develop our own model, but we have no time so we decided to use the KeyToText which we were mentioned about. We tried different pipelines of KeyToText with Clotho metadata and decided to use the best pipeline based on the BLEU scores.

Finally, we gave these keywords from CNN models as input to our KeyToText model and we received sentences. Using the KeyToText with our PANN and 14-Layer CNN models, we got 1-gram and 2-gram scores from the BLEU metric, a widely used metric in NLP.



	PANN's + KeyToText	14-Layer CNN + KeyToText
1-gram BLEU Score	0.36	0.27
2-gram BLEU Score	0.04	0.02

Figure 12. BLEU Scores of Best Architectures

5. Conclusion

In this paper, we tried to approach the audio captioning problem as an image captioning problem using the mel-spectrogram representation of audio files instead of audio files. We proposed multiple models with CNN encoders and Transformer decoders. We also tried different approaches in the training process. We got the best results with the models which are pre-trained with very large datasets and then used in our test set. Also, we wanted to develop our encoder but we didn't have much time so we used a pre-trained model. In the future, we can develop our decoder model and can use more complex models with larger datasets to improve performance.

References

Bhatia, G. keytotext. URL https://github.com/gagan3012/keytotext.

Chen, K., Wu, Y., Wang, Z., Zhang, X., Nian, F., Li, S., and Shao, X. Audio captioning based on transformer and pre-trained cnn. In *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020). Tokyo, Japan*, pp. 21–25, 2020.

Drossos, K., Adavanne, S., and Virtanen, T. Automated audio captioning with recurrent neural networks. In *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, pp. 374–378. IEEE, 2017.

Drossos, K., Lipping, S., and Virtanen, T. Clotho: An audio captioning dataset. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 736–740. IEEE, 2020.

Kim, C. D., Kim, B., Lee, H., and Kim, G. 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)*, pp. 119–132, 2019.

Kong, Q., Cao, Y., Iqbal, T., Wang, Y., Wang, W., and Plumbley, M. D. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:2880–2894, 2020.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

Wang, H., Yang, B., Zou, Y., and Chong, D. Automated audio captioning with temporal attention. *DCASE2020 Challenge, Tech. Rep.*, 2020.

Wu, M., Dinkel, H., and Yu, K. Audio caption: Listen and tell. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 830–834. IEEE, 2019.

Xu, X., Dinkel, H., Wu, M., Xie, Z., and Yu, K. Investigating local and global information for automated audio captioning with transfer learning. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 905–909. IEEE, 2021.