AUTOMATED AUDIO CAPTIONING

Technical Report

Nikita Kuzmin
Third-year student

Moscow State University
CMC Faculty
Mathematical Methods of Forecasting Dept.
GSP-1, 1-52, Leninskiye Gory
Moscow, 119991, Russia
paniquexx@gmail.com

Alexander Dyakonov
PhD in physics and mathematics, academic adviser

Moscow State University
CMC Faculty
Mathematical Methods of Forecasting Dept.
GSP-1, 1-52, Leninskiye Gory
Moscow, 119991, Russia
djakonov@mail.ru

ABSTRACT

This task can be stated as an automated generation textual content description from the raw audio file. We propose a method for the automated audio captioning task. We examined the impact of augmentations (MixUp, Reverb, Pitch, Over-drive, Speed) on method performance. Our method based on modified encoder-decoder architecture. The encoder consists of three bidirectional gated recurrent units (GRU). The decoder consists of one gated recurrent unit (GRU) and one fully-connected layer for classification. The encoder input is log-mel spectrogram features for every part of audio file segmented by Hann window [1] of 1024 samples with a 50% overlap. The decoder output is a matrix with probabilities of words for each position in a sentence. We used BLEU_1, BLEU_2, BLEU_3, BLEU_4, ROUGE_L, METEOR, CIDEr, SPICE, SPIDEr metrics to compare methods.

***Index Terms*—** Audio Captioning, Recurrent Neural Networks, Natural Language Generation, MixUp

1. INTRODUCTION

The automated audio captioning (AAC) problem can be stated as an annotation (textual description generation) of an audio track via automated system (for example neural networks). This task is very important, because the audio classification approach is unable to explain inner relationships in audio. Audio captioning methods can model concepts, physical properties of objects and environment, and high-level knowledge (ex. "a clock rings three times"). Also, audio captioning is more challenging than classification because it consists of two main tasks:

- audio processing,
- text generation.

There were two papers before about audio captioning - [2], [3]

At first glance, the audio captioning task seems like image captioning, but there are some significant differences. Humans can easily annotate pictures, since every object on the image has a specific shape, color, size, etc. Most people more familiar with a visual representation, than with an . For example, if we consider mel-spectrogram audio representation, we can notice the following differences between common images [4]:

1. Sound is "transparent" - highly likely one pixel of an image, which is assumed to belong to a single object. In spectrograms it is not true.
2. The axes of spectrograms do not carry the same meaning, but in pictures x and y have the same meaning.
3. The spectral properties of sound are non-local. It means that neighboring pixels can't be assumed to belong to the same object like in pictures.

Results of an automated audio captioning algorithm can be used in different spheres: for hearing-impaired people, in manufacture (for more accurate emergencies description), video analysis (we can describe video more explicitly with detailed information about the audio track).

Section 2 describes the dataset for evaluation, Section 3 tells about proposed method (preprocessing, augmentations, neural network architectures, post processing). The evaluation procedure, experiments, results are presented in Section 4. The conclusion and future work are described and discussed in Section 5.

2. DATASET DESCRIPTION

The task uses the freely available Clotho dataset [5]:

It consists of audio samples of 15 to 30 seconds duration, each audio sample has five captions of 8 to 20 words long.

There are a total of 4981 audio samples with 24 905 captions.

Clotho has a 4365 unique words and is divided into three splits: development (60%), evaluation (20%) and testing (20%). All words appear proportionally between splits.

Words that cannot be divided using the above scheme of 60-20-20 appear at least one time in the development split and at least one time to one of the other two splits!

2.1. Audio samples information

Clotho audio data are extracted from an initial set of 12 000 audio files collected from Freesound [6]. The 12k audio files have durations ranging from 10s to 300s, no spelling errors, good quality

(44.1kHz and 16-bit), and no tags on Freesound indicating sound effects, music or speech. Before extraction, all 12k files were normalized and the leading and trailing silences were trimmed.

2.2. Captions information

The captions were gathered by employing the crowdsourcing platform Amazon Mechanical Turk and a three-step framework. The three steps are:

1. audio description;
2. description editing;
3. description scoring.

For detailed description see [7].

Every audio file in Clotho dataset has five different captions.

3. PROPOSED METHOD

3.1. Dataset preprocessing

Our method takes as an input raw, mono audio data with 44.1 kHz sampling frequency and 16 bits sample width and extracts log-mel spectrogram features for every part of audio file segmented by Hann window of 1024 sample with 50% overlap. For each resulting part we extract $N_{\text{feats}} = 128$ features.

Captions preprocessing was taken from DCASE2020 Baseline code [8], which takes as an input csv file with names of audio files in the first column and five captions per each audio file in the following five columns. Baseline code encodes caption with the Bag of Words model. Each sentence starts from "start of sentence" token (<SOS>) and ends by "end of sentence" token (<EOS>).

3.2. Data augmentation

- MixUp
  First of all, we used **Input MixUp** implementation, which was introduced in [9], with some differences in target mixing.
  1. Audio tracks mixing
     The key idea is to mix random input audio files. For example, suppose we have in our train dataset audio x_i and audio x_j. The resulted audio after Input MixUp application will be:
     $$x = \lambda x_i + (1 - \lambda)x_j, \text{ where } \lambda \sim Beta(\alpha, \alpha), \quad (1)$$
     where $\alpha = 16$
  2. Captions mixing: Suppose we have y_i and y_j captions corresponding to x_i and x_j, y_i has length l_i and $y_j - l_j$. There are several ways to implement captions mixing:
     - Simple concatenation ($[y_i, y_j]$ or $[y_j, y_i]$)
     - For definiteness, we assume $l_i > l_j$. We suppose y_i^k - k-th word in y_i. After concatenation we will get $[y_i^1, y_j^1, y_i^2, y_j^2, \dots y_i^{l_i}]$
     - Random words shuffling after concatenation

     Our experiments have shown that MixUp with simple concatenation achieves the best score.

  Augmentations below were implemented by **pysndfx** library [10].

- Reverb
  with **reverberance**, **room_scale**, **stereo_depth** parameters from $U[0, 50]$ distribution.
- Pitch
  with **shift** parameter from $U[-300, 300]$ distribution.
- Overdrive
  with **gain** parameter from $U[2, 10]$ distribution.
- Speed
  with **factor** parameter from $U[0.9, 1.1]$ distribution.

Every type of augmentations was applied online with 0.5 probability to each sample.

3.3. Neural networks architectures

In this work we will compare following network achitectures:

1. DCASE2020 Baseline [11]:
   the baseline neural network consists of an encoder and decoder.
   The encoder is a three-layered bidirectional GRU [12] with tanh activation function and a residual connection [13] between the second and the third layer. All but the last layer have 128 cells and the last one 256. This results correspondingly to 256 and 512 total cells due in bidirectionality (2×128 and 2×256).
   The decoder is a one-layered GRU with tanh activation functions, followed by a fully-connected layer with softmax activation functions. The GRU layer consists of 256 cells, it takes as input the last output of the encoder and produces a matrix $Y \in R^{N_{\text{words}} \times K}$, where N_{words} – words amount in vocabulary, K – maximum tokens in prediction.
   An illustration of the DCASE2020 Baseline is in Figure 1.
2. Modification v1 DCASE2020 Baseline
   We modified DCASE2020 Baseline network architecture by adding an embedding layer (based on mean aggregation) between encoder and decoder.
3. Modification v2 DCASE2020 Baseline
   We modified DCASE2020 Baseline network architecture by adding a fully-connected layer between encoder and decoder.
4. Modification v3 DCASE2020 Baseline
   We modified DCASE2020 Baseline network architecture by adding a attention between encoder and decoder.

An illustration of the DCASE2020 Modifications v1, v2, v3 are in Figure 1 and Figure 2.

3.4. Training

The neural networks were trained using Adam optimizer with the 0.001 learning rate and crossentropy loss. Dropout rate was 0.5 and 0.25 for the input and recurrent connections, respectively, in the GRUs of the encoder and the decoder, and the batch size was 80. The neural network were trained for 50 epoches on GPU NVIDIA RTX 2080 TI. The code was written on PyTorch framework.

3.5. Post-processing

We have noticed that predicted captions have a lot of sequences with repeated tokens. So we decided to compress each

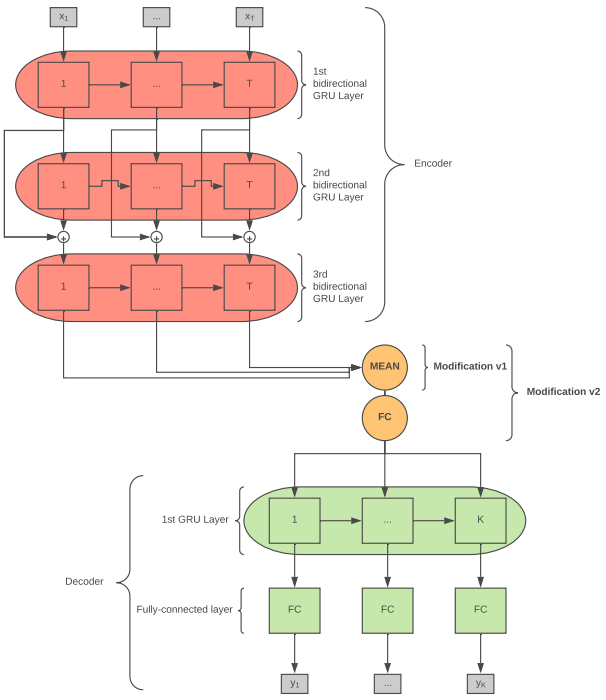

Figure 1: DCASE2020 Baseline and v1, v2 modifications architectures

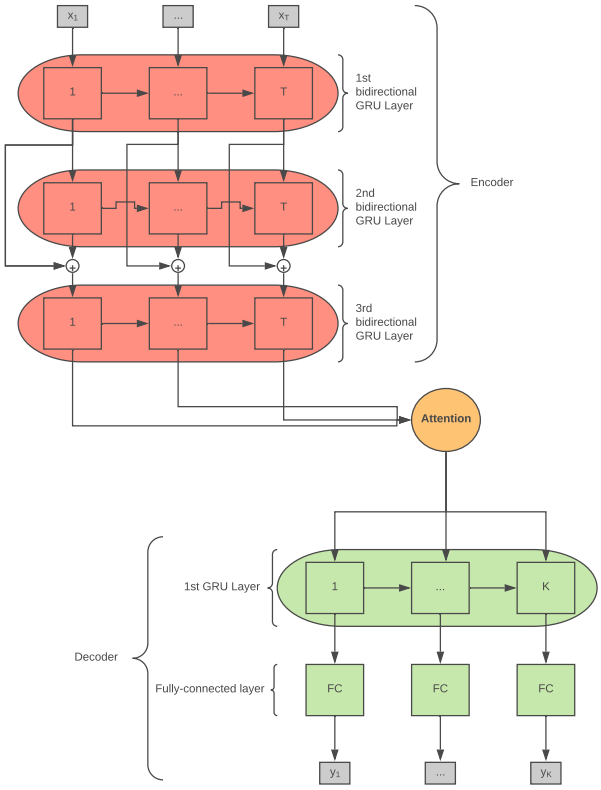

Figure 2: Modification v3 (attention) DCASE2020 Baseline architecture

sequence with repeating tokens in captions into one element (ex. caption $[<SOS>, a, a, a, b, g, g, c, <EOS>] \longrightarrow^{compress} [<SOS>, a, b, g, c, <EOS>]$), but the SPIDEr score got worse.

4. EVALUATION

We used a lot of metrics for evaluation – BLEU_1, BLEU_2, BLEU_3, BLEU_4 [14], ROUGE_L [15], METEOR [16], CIDEr [17], SPICE [18], **SPIDEr**. **SPIDEr** - main metric [19], so we will describe it in details:

- **CIDEr** (Consensus-based Image Description Evaluation) – metric, which was introduced in [17]. CIDEr measures the similarity of generated sentence against a set of ground truth sentences written by humans. This metric shows a high agreement with humans judgments of consensus. Using sentence similarity, the notions of grammaticality, saliency, importance and accuracy (precision and recall) are inherently captured by CIDEr metric.
- **SPICE** (Semantic Propositional Image Caption Evaluation) – metric, which was introduced in [18]. SPICE captures human judgments over model-generated captions better than other automatic metrics (CIDEr, METEOR, etc.). Furthermore, SPICE can answer questions such as "Which caption-generator best understands colors?" and "Can caption-generators count?".
- **SPIDEr** - metric, which is an average of CIDEr and SPICE metrics, so it has combined benefits from both metrics:

$$\text{SPIDEr} = \frac{\text{CIDEr} + \text{SPICE}}{2} \quad (2)$$

4.1. Experiments

In this part we will compare DCASE2020 Baseline with its modifications (v1, v2, v3).

We will also compare DCASE2020 Baseline model and its modifications (v1, v2, v3) with different data augmentations:

1. MixUp;
2. Reverb, Pitch, Overdrive, Speed (Another augs);
3. MixUp, Reverb, Pitch, Overdrive, Speed (All augs).

Let's describe five approaches (1-4 were submitted):

1. Ensemble of 4 models, which are described below. We select unique predicted tokens by every model for each audiofile, then concatenate 4 strings of tokens.
2. This approach is based on Modification v2 of DCASE2020 Baseline and MixUp augmentation. It was trained with lr= 0.0001.
3. This approach is based on Modification v2 of DCASE2020 MixUp, Reverb, Pitch, Overdrive, Speed augmentations. It was trained with lr= 0.0001.
4. This approach is based on Modification v3 of DCASE2020 Baseline and MixUp augmentation. It was trained with ReduceOnPlateau scheduler with step= 5 and starting lr= 0.0004.

5. This approach is based on Modification v3 of DCASE2020 Baseline and Reverb, Pitch, Overdrive, Speed augmentations. It was trained with ReduceOnPlateau scheduler with step= 5 and starting lr= 0.0001.

Results can be found in yaml files.

5. CONCLUSION AND FUTURE WORK

5.1. Conclusion

In this paper we made an effort to improve machine understanding of audio data. We showed the impact of augmentations (MixUp, Reverb, Pitch, Overdrive, Speed) in the audio captioning task. We compared different modifications of DCASE2020 Baseline architecture. The best performance was showed by DCASE2020 Modification v2 with all augmentations.

5.2. Future work

We used **Input MixUp** implementation in this paper. In [20] paper authors presented **Manifold MixUp** implementation, which is more accurate and robust.

Currently there are no experiments with transformer architectures in papers about audio captioning task. We suppose that transformer architecture can reach competitive result.

6. REFERENCES

[1] https://en.wikipedia.org/wiki/Hann_function.

[2] K. Drossos, S. Adavanne, and T. Virtanen, "Automated audio captioning with recurrent neural networks," 2017.

[3] M. Wu, H. Dinkel, and K. Yu, "Audio caption: Listen and tell," *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019. [Online]. Available: http://dx.doi.org/10.1109/ICASSP.2019.8682377

[4] D. Rothmann, "What's wrong with CNNs and spectrograms for audio processing?" https://towardsdatascience.com/whats-wrong-with-spectrograms-and-cnns-for-audio-processing-311377d7c 2018.

[5] K. Drossos, S. Lipping, and T. Virtanen, "Clotho: An audio captioning dataset," 2019.

[6] https://freesound.org/.

[7] S. Lipping, K. Drossos, and T. Virtanen, "Crowdsourcing a dataset of audio captions," 2019.

[8] http://github.com/audio-captioning/dcase-2020-baseline/.

[9] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," 2017.

[10] C. Thome, "pysndfx library" https://pypi.org/project/pysndfx/.

[11] https://github.com/audio-captioning/dcase-2020-baseline/.

[12] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[14] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "Bleu: a method for automatic evaluation of machine translation," 10 2002.

[15] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," 01 2004, p. 10.

[16] A. Lavie and A. Agarwal, "Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments," pp. 228–231, 07 2007.

[17] R. Vedantam, C. L. Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," 2014.

[18] P. Anderson, B. Fernando, M. Johnson, and S. Gould, "Spice: Semantic propositional image caption evaluation," 2016.

[19] 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)*, Oct 2017. [Online]. Available: http://dx.doi.org/10.1109/ICCV.2017.100

[20] V. Verma, A. Lamb, C. Beckham, A. Najafi, I. Mitliagkas, A. Courville, D. Lopez-Paz, and Y. Bengio, "Manifold mixup: Better representations by interpolating hidden states," 2018.