Optimization Of Multimodal Analysis Of Image And Audio Pairs

Junrui Ni (junruin2) Yuxi Gu (yuxigu2) Xuanyi Zhu (xzhu42)
University of Illinois at Urbana-Champaign

Abstract

Given a dataset of images and spoken audio captions, the papers written by Harwath et al. in [1][2][3] presents a method for discovering word-like acoustic units from continuous speech at the waveform level with no additional text transcriptions or conventional speech recognition apparatus and grounding them to semantically relevant image regions.

Based on their work, we extend and optimize their model to achieve better performance and accuracy to find better matches between images and audios. Once the system is trained, it can be used to search images based on audio captions or search captions based on images.

Introduction

First, we briefly introduce our dataset. The image dataset we used is called MIT Places Database for Scene Recognition, or Places 205. It is a database for scene recognition with 205 scene categories and 2.5 millions of images with a category label. The audio dataset we used is called The Places Audio Caption Corpus. It contains approximately 400,000 spoken captions for natural images drawn from the Places 205 image dataset. It was collected to investigate multimodal learning schemes for unsupervised co-discovery of speech patterns and visual objects, starting from 2016 by Harwath et al.[1][2][3] As limited by computational resource, we only use a subset of 110k image and audio pairs to train all our models. Therefore, our baseline result may show some slight difference between the results in [2], which uses exactly the same architecture and training scheme as our baseline model, but trained on more than 200k image/audio pairs.

We will now briefly talk about the baseline architecture in the remaining of this section. Although the architecture is the same as in [2], we wanted give an overview of how the whole model is built for multimodal matching between images and audio captions.The overall baseline architecture is showed in Figure 1. Baseline Architecture.

The above architecture takes a pair of image/audio and generates a similarity score between the image and audio caption pair. It has two branches: one is an image branch for processing the input image and the other is an audio caption branch for processing input audio captions. The feature vectors extracted by the two branches are then taken as inputs of a dot product, where the result of the dot product is considered as the similarity score between the input image and audio caption pair, with a higher score indicating higher level of similarity.The final goal is to enforce a margin between matched image and audio caption pair and unmatched pairs, which is achieved by a triplet-loss like loss function during training.

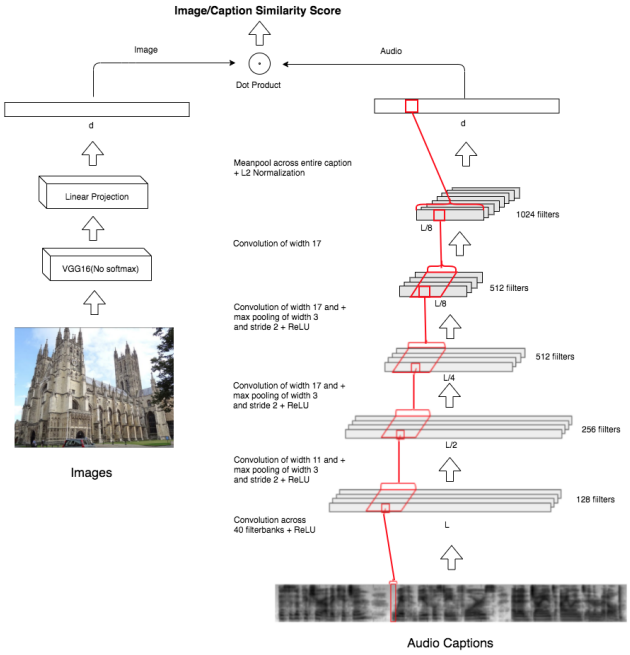


Figure 1: Baseline Architecture

Image Branch: All images are first resized from 256 by 256 to 224 by 224, and then fed into a pre-trained VGG16 model[4]. The weights of all layers up to the second fully connected layer are kept fixed, and we replaced the last softmax layer with a 1024-dimensional linear projection layer, which maps the 4096-dimensional activations of the second fully connected layer into the 1024-dimensional multimodal embedding space.

Audio Caption Branch: All audio segments are first converted into log mel spectrogram with 40 filter banks, and then padded/truncated into a fixed length of 1024 frames. After that, are fed into a customized convolutional neural network as shown below. The final layer turns the audio segment into a 1024-dimensional vector by performing average pooling across the entire caption.

Connection Part: The overall multimodal network is formed by tying together the learned visual feature vectors and learned audio feature vectors and map them into a shared embedding space. Then, it computes an inner product between them, representing the similarity score between a given image/caption pair.

After defining how to compute the similarity score between image and audio caption pairs, a loss function using

similarity score terms is defined in order to minimize the training loss of the whole model. The loss function $L(\theta)$ is showed below (**Imposter Image/Caption**: An image imposter is and an audio imposter are randomly selected from the current batch, with indices different from the current ground truth image/caption.)[1][2][3]:

$$L(\theta) = \sum_{j=1}^{B} \max(0, S_j^{c\_imp} - S_j^p + 1) + \max(0, S_j^{i\_imp} - s_j^p + 1)$$

B: Number of Image/Caption pairs.
S_j^p:The Similarity score between j^{th} image and j^{th} caption, which is the ground truth image and audio caption pair.
$S_j^{i\_imp}$: The Similarity score between j^{th} option and its imposter image.
$S_j^{c\_imp}$:The Similarity score between j^{th} image and its imposter caption.

Note: In the loss function, the index of the imposter image and the index of the imposter caption are randomly selected and they can be different or the same.
And a brief visualization of the original loss function is showed in Figure 2. Original Loss Function.

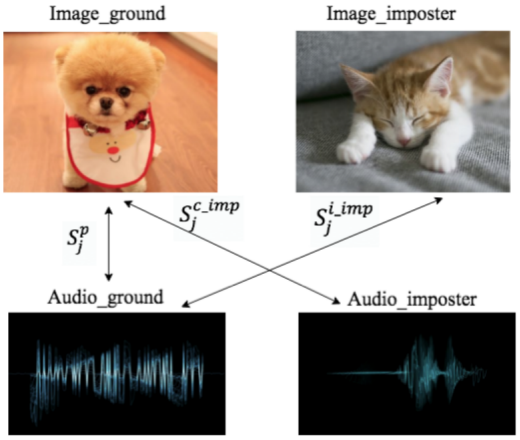


Figure 2: Original Loss Function

After the loss function is defined, we minimize the average loss using gradient descent, on 110k image/audio pairs from the places 205 image and audio dataset. A separate validation set with 1024 pairs of matched image and audio is used to monitor the validation loss during training, as well as for evalulating top 1, top 5 and top 10 recall score of the trained model. We used an Adam Optimizer with an initial learning rate of $2.5e-4$, and decays by a factor of 2 every 5 epochs. The whole model is trained for 10 epochs, instead of 50 epochs as in [2], as we find that validation loss almost converges at that point.

First Optimization - Efficient use of training data

The basic idea for the first optimization is to allow for more efficient use of training data. Every summation term in the original loss function only takes into consider three similarity pairs: the ground truth image/caption pair as the anchor, the audio imposter pair with a randomly selected audio replacing the ground truth audio, and the image imposter pair with a randomly selected image replacing the ground truth image. As the above may improve training speed, however, there might be a small problem sometimes: since choosing imposter image and audio is completely random, it might be the case that some images or audio in the current batch never get to be chosen as imposters. As we are using a margin-enforcing loss function, we therefore believe that allowing every image and audio in the current batch to be imposters at least once might help the network to learn the multimodal relationship better, given the same amount of data.

We do the above as the following:
First, for each ground truth pair in the batch, we randomly select two indices from the current batch, one as the imposter caption index and the other as the imposter image index. This step is the same as in the original loss function.
Second, we find the matched image for the selected imposter caption and find the matched caption for the selected imposter image. We now have three image/caption pairs that can be used as ground truth pairs in turn, which is 1)the current index pair, 2)the first imposter pair and 3)the second imposter pair.
Finally, we augment the original loss function as follows: in addition to using the current index pair as anchor, we used its image and its audio caption as imposters as well, by treating the first imposter pair as ground truth pair and the second imposter pair as ground truth pair, in turn. The modified loss function is showed below:

$$L(\theta) = \sum_{j=1}^{B} t1 + t2 + t3$$

B: Number of Image/Caption pairs.
t1: $\max(0, S_j^{imp1Caption\_gImage} - S_j^{gCaption\_gImage} + 1)$
$+ \max(0, S_j^{gCaption\_imp2Image} - S_j^{gCaption\_gImage} + 1)$
t2: $\max(0, S_j^{gCaption\_imp1Image} - S_j^{imp1Caption\_imp1Image} + 1)$
$+ \max(0, S_j^{imp1Caption\_gImage} - S_j^{imp1Caption\_imp1Image} + 1$
t3: $\max(0, S_j^{gCaption\_imp2Image} - S_j^{imp2Caption\_imp2Image} + 1)$
$+ \max(0, S_j^{imp2Caption\_gImage} - S_j^{imp2Caption\_imp2Image} + 1$
$S_j^{ACaption\_BImage}$: The Similarity score between A's caption

and B's image when we are using j^{th} image/caption pair as the ground truth image/caption pair, where A and B could be g, imp1 or imp2. And g represents the ground truth pair, imp1 represents the first imposter pair and imp2 represents the second imposter pair.

First Optimization Results

We implemented the original architecture from scratch. After the model is trained, we evaluated the model on the validation set with 1024 image/audio pairs, using recall score as the criterion. The comparison between using the original loss function and the optimized loss function is showed in the table.

As can be seen from the table, the top 1 recall score of both image search(searching an image using audio) and image annotation(searching an audio caption using image)increase significantly, showing that the trained network shows more ability to associate ground truth pairs with each other. The top 5 and top 10 recall scores haven't been improved that much, expect for top 10 recall score for image search, but an overall improvement still can be clearly seen.

Recall Score(110k data pairs)	R@1	R@5	R@10
Baseline(Image Search)	0.069	0.223	0.319
Improved Loss(Image Search)	**0.090**	**0.229**	**0.354**
Baseline(Image Annotation)	0.074	0.249	0.366
Improved Loss(Image Annotation)	**0.094**	**0.267**	0.363

Second Optimization - Adding temporal imformation

In the audio network branch proposed by [1][2], the final feature vector is pooled from the entire caption for all 1024 final feature map of the CNN. While [2] uses fairly large temporal kernels in his architecture(11 and 17), the receptive field at the very last layer is still not enough to cover the entire caption. Therefore, during caption-wise pooling, temporal information may not be as well captured by the final feature vector.

We proposed to augment the spectrogram feature vector by explicitly modelling temporal information in another branch parallel with the baseline architecture. Suppose our spectrogram is an image of size (40,1024,1), where the first dimension is the frequency axis, the second dimension the time axis, and the third the channel axis.First, three sets of vertical frequency filters of size (10,1,128) with ReLU activations after each set of filter, with two sets of vertical max pooling between them. Since the third set of vertical frequency filter uses valid padding, the frequency axis of the spectrogram is therefore collapsed, but the number of channels has increased from 1 to 128. The output is therefore (1,1024,128).We then feed the output to some layer that explicitly does sequence modelling. Here we used Temporal Convolution Networks.[5]

We now describe the architecture of temporal convolution network, or TCN. This kind of network is basically a 1D fully convolutional network with two modifications, causal convolution and dilation. Causal convolution is used to model the relationship $f(y_t|y_{t-1}...y_{t-k})$, as in standard recurrent architectures, while dilation is used to exponentially increase the receptive field of convolutional filters. Also, residual block is used between previous layer and current layer to to learn modications to the identity mapping, rather than full transformation, which has repeatedly been shown to benet very deep networks.[5] As suggested by its novel performance over both GRU and LSTM in long sequence tasks such as sequential MNIST, TCN is very good at capturing long term temporal information accurately, which is the case for us as spectrogram we used contains hundreds of frames[5]. An illustration of a single TCN stack is shown below.[5]

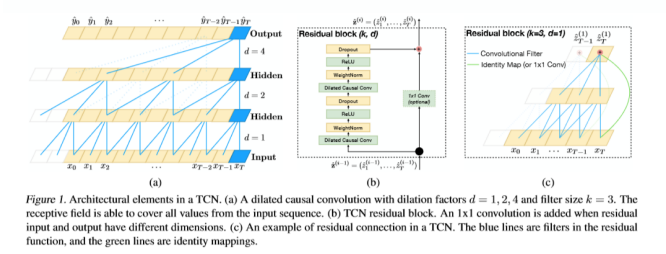


Figure 1. Architectural elements in a TCN. (a) A dilated causal convolution with dilation factors $d = 1, 2, 4$ and filter size $k = 3$. The receptive field is able to cover all values from the input sequence. (b) TCN residual block. An 1x1 convolution is added when residual input and output have different dimensions. (c) An example of residual connection in a TCN. The blue lines are filters in the residual function, and the green lines are identity mappings.

Figure 3: TCN Stack

We use two stacks of TCN, where the first stack models the forward direction, and the second stack models backward direction. Every stack uses 512 filters, and a dilation up to 1024 to get a receptive field large enough for the entire sequence. Average pooling is applied on both stacks separately at each time step, which,concatenated, gives a final temporal feature vector of size 2048. This temporal vector is then concatenated with the caption-wised pooled vector of the baseline architecture, with the last set of feature maps removed to control the total number of parameters. The concatenated feature vector of size 2560 is then fed through a dense layer with ReLU activation, in the hope to learn a meaningful correlation among the concatenated vector within the shared emdedding space.

Second Optimization Results

Adding temporal information can be tricky. During our experiment, we found that if we threw away the original

baseline architecture entirely, and used a one branch full-TCN to model the spectrogram directly, in a pyramidal fashion, or if TCN is appended to the baseline architecture, the whole model would be stuck in some bad local optima shortly after training starts. If we modelled the temporal information in the temporal branch at some other location, like after the spectrogram feature vector is pooled once/twice/three times, the recall score slightly suffered. If we used GRU/LSTM instead of TCN, we'd get an unbearably long training time and some slightly worse results, due to the temporal length of the spectrogram we are using.
Yet, despite our best effort, we are only able to increase the recall score marginally here. Due to the extremely long training time using all 110k image/audio pair(it would take 10-20 hrs for one model to get fully trained), to help us iterate through different models faster, our experiment in this section are conducted on a subset of 30k image/audio pairs only. Based on our experience developing the baseline model, we find that although the recall score using 30k pairs is only half of that of using all 110k pairs, it would still give enough capacity to test potential models.

Results

There are two ways to do searching, image search and image annotation:

1)Image Annotation: The input query is an image, and the output is related captions.
2)Image Search: The input query is a caption, and the output is related images.

We now display the search result conducted on the validation set with 1024 image/audio pairs. As both of our optimization methods only changed the recall score slightly, we believe that the search results should also be very similar, and thus is only displaying baseline results here.For better understanding and visualization, we displayed the ASR text captions instead of spectrograms and provided both images and text captions in pairs regardless of the type of searching we are doing. An example of searching related captions for an image is showed below. Shown on the top is an image with its ground truth audio transformed into text caption. Below are its five highest scoring audio files transformed into text caption form.

Image Annotation

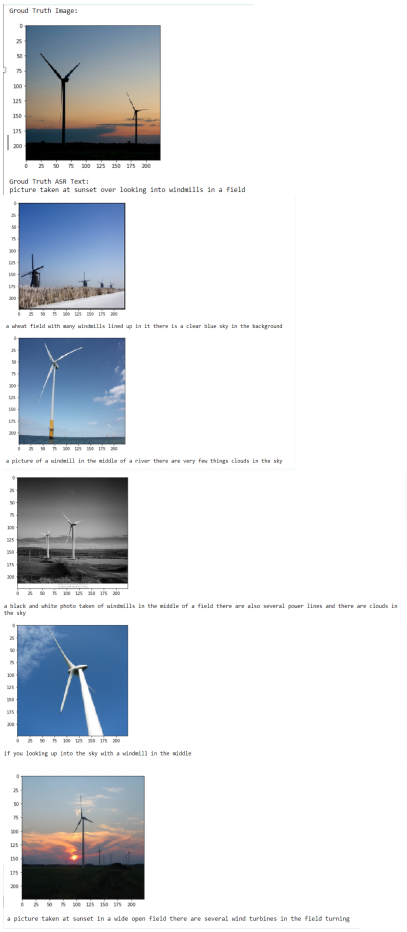


Image Search

Future Work

In [3], a new similarity score has been developed: instead of taking the dot product between a single image feature vector and a single audio feature vector in the same shared embedding space, they tried to match individual time frames in the last convolutional layer of the audio branch with individual spatial pixels in the last convolutional layer of the image branch. First they calculate the "matchmap" M: "Let I represent the output feature map output of the image network branch, A be the output feature map of the audio network branch, and I_p and A_p be their globally average-pooled counterparts. One straightforward choice of similarity function is the dot product between the pooled embeddings, $S(I,A) = I_p^T A_p$. Notice that this is in fact equivalent to first computing a 3rd order tensor M such that $M_{r,c,t} = I_{r,c,:}^T A_{t,:}$". [3]After calculating M, they defined three kinds of similarity socres: SISA(sum image, sum audio), MISA(max image, sum audio),SIMA(sum image, max audio), as follows:

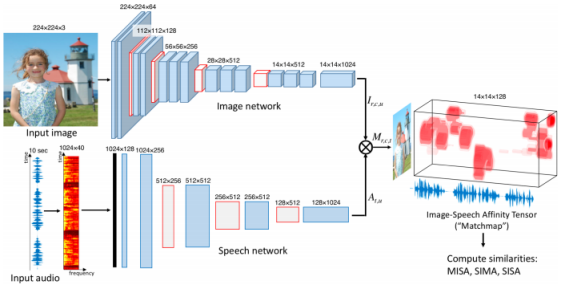
$$SISA(M) = \frac{1}{N_r N_c N_t} \sum_{r=1}^{N_r} \sum_{c=1}^{N_c} \sum_{t=1}^{N_t} M_{r,c,t}$$

$$MISA(M) = \frac{1}{N_t} \sum_{t=1}^{N_t} \max_{r,c}(M_{r,c,t})$$

$$SIMA(M) = \frac{1}{N_r N_c} \sum_{r=1}^{N_r} \sum_{c=1}^{N_c} \max_{t}(M_{r,c,t})$$

The architecture of their new network is shown below, copied from t[3].

This architecture, compared to results in [2], showed



slightly better recall score on both image search and image annotation, if the best among $MISA, SIMA, SISA$ was being used. Since they were not using any fully connected layer in VGG, they showed possibility of traning the network end-to-end without any pre-training. More importantly, this architecture allowed several things beyond basic image search and image annotation, such as concept discovery and speech prompted object localization.

We believe that the two methods discussed above can also be used to improve the result of the architecture above: we can directly apply the first optimization with minimal change, and try to design a TCN-based or TCN-embedded network that better captures audio time dependency so that better matchmaps could be obtained. For example, a local convolutional network might only be able to "see" the word "house" in the audio, but incorporating time dependencies, it might actually be able to see the word "red house" instead, which should be able to match image features better, if fully trained. Limited by time and computational resources, however, we had no choice but to leave this as future work.

References

1. D. Harwath, A. Torralba, and J. Glass, "Unsupervised Learning of Spoken Language with Visual Context," Proc. of Neural Information Processing Systems (NIPS), Barcelona, Spain, December 2016

2. D. Harwath and J. Glass, "Learning Word-Like Units from Joint Audio-Visual Analysis, "Proc. of the Annual Meet-

ing of the Association for Computational Linguistics (ACL), Vancouver, Canada, July 2017
3. D. Harwath, A. Recasens, D. Suris, G. Chuang, A. Torralba, and J. Glass, "Jointly Discovering Visual Objects and Spoken Words from Raw Sensory Input," Proc. of the 2018 European Conference on Computer Vision (ECCV), Munich, Germany, September 2018
4. Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", arXiv, September 2014
5. Shaojie Bai, J. Zico Kolter, Vladlen Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling", arXiv, March 2018