SpeechBERT: An Audio-and-text Jointly Learned Language Model for End-to-end Spoken Question Answering

Yung-Sung Chuang  Chi-Liang Liu  Hung-yi Lee  Lin-shan Lee

College of Electrical Engineering and Computer Science, National Taiwan University
{chuangyungsung,liangtaiwan1230,tlkagkb93901106}@gmail.com, lslee@gate.sinica.edu.tw

Abstract

While various end-to-end models for spoken language understanding tasks have been explored recently, this paper is probably the first known attempt to challenge the very difficult task of end-to-end spoken question answering (SQA). Learning from the very successful BERT model for various text processing tasks, here we proposed an audio-and-text jointly learned SpeechBERT model. This model outperformed the conventional approach of cascading ASR with the following text question answering (TQA) model on datasets including ASR errors in answer spans, because the end-to-end model was shown to be able to extract information out of audio data before ASR produced errors. When ensembling the proposed end-to-end model with the cascade architecture, even better performance was achieved. In addition to the potential of end-to-end SQA, the SpeechBERT can also be considered for many other spoken language understanding tasks just as BERT for many text processing tasks.

1. Introduction

Various spoken language processing tasks, such as translation [1], retrieval [2], summarization [3] and understanding [4] have been very successful with a standard cascade architecture: an ASR front-end module transforming the speech signals into text form, followed by the downstream task module (such as translation) trained on text taking the ASR output as normal text input. However, the end-to-end approach trying to consider the two modules as a whole is always attractive for the following reasons. The two modules in the cascade architecture locally optimize the two tasks with different criteria, while the end-to-end approach may obtain globally optimized performance for the overall task. The ASR module minimizes the WER, which is not necessarily directly proportional to the performance measure of the overall task. Much information is inevitably lost when the speech signals are transformed into text with errors, and the errors can’t be recovered in the following module. The end-to-end approach allows the possibility of capturing information directly from the speech signals not shown in the ASR output and offering overall performance less limited by the ASR accuracy.

Some spoken language tasks such as translation, retrieval, and understanding (intent classification and slot filling) have been achieved with end-to-end approaches [5, 6, 7, 8, 9] although it remains difficult to obtain significantly better results than the standard cascade architecture. These tasks are primarily sentence-level, in which extracting some local information in a short utterance may be adequate. Spoken question answering (SQA) considered here is known as a much more difficult problem. The inputs to the SQA task are much longer spoken paragraphs. In addition to understanding the literal meaning, the global information in the paragraphs needs to be organized, and sophisticated reasoning is usually required to find the answers. Fine-grained information is also important to predict the exact position of the answer span from a very long context. This paper is probably the first known attempt to try to perform such a difficult SQA task with an end-to-end approach.

Substantial improvements in text question answering (TQA) tasks trained and tested on text data have been observed after the large-scale self-supervised pre-trained language models appeared, such as BERT [10] and GPT [11]. Instead of learning the TQA tasks from scratch, these models were first pre-trained on a large unannotated text corpus to learn self-supervised representations for the general purpose and then fine-tuned on the downstream TQA dataset. Results comparable to human performance on SQuAD datasets [12, 13] were obtained in this way. However, previous work [14] indicated that such an approach may not be easily extendable to the SQA task on audio data by directly cascading an ASR module in the front. Not only the ASR caused serious problems, but very often the true answer spans included name entities or OOV words which cannot be correctly recognized at all, thus cannot be identified by the following TQA module trained on text. That is why all questions with recognition errors in answer spans were discarded in the previous work [14]. This led to the end-to-end approach to SQA proposed here.

The BERT model [10] useful in TQA tasks was able to transform word tokens into contextualized embeddings carrying plenty of information. For SQA task, it is certainly desirable to transform audio words (audio signals for word tokens) also into such embeddings, but much more challenging. The same word token can have millions of different audio signal realizations in different utterances. The boundaries for audio words in utterances are not available. The next problem is even much more challenging. BERT can learn semantic information of a word token based on its context in text form. Audio words in audio data have context only in audio form, which is much more noisy, confusing, unpredictable, and difficult to handle. Learning semantics from audio context is really hard.

Audio Word2Vec [15] was the first effort to transform audio words with known boundaries into embeddings carrying phonetic information only, no semantics at all. Speech2Vec [16] then tried to imitate the training process of skip-gram or CBOW in Word2Vec [17] to extract some semantic features. It was proposed [18] to learn to jointly segment the audio words out of utterance and extract the embeddings. Other efforts then tried to align the audio word embeddings with text word embeddings [19, 20]. Some more recent works [21, 22, 23, 24, 25, 26] even tried to obtain embeddings for audio signals using approaches very similar to BERT, but primarily extracting acoustic information with very limited semantics. These approaches may be able to extract some semantics with audio word embeddings, but the level of semantics obtained was still very far from that required for the SQA tasks.

In this paper, we propose an audio-and-text jointly learned
SpeechBERT model for the end-to-end SQA task. SpeechBERT is a pre-trained model learned from audio and text datasets, so as to be able to produce embeddings for audio words, and these embeddings can be properly aligned with the embeddings for the corresponding text words offered by a typical text BERT model trained with text datasets. Standard pre-training and fine-tuning as text BERT are also performed. When used in the SQA task, performance comparable to or better than the cascade architecture was obtained.

2. SpeechBERT for End-to-end SQA

Here we assume the training audio datasets include the ground truth transcripts, and an off-the-shelf ASR engine is available. So we can segment the audio training data into audio words (audio signals for the underlying word tokens) by forced alignment. For audio testing data we can use the ASR engine to obtain audio words including their boundaries too, although with errors.

The overall training process for the SpeechBERT for end-to-end SQA proposed here is shown in Fig. 1. We first use the text dataset to pre-train a Text BERT (upper right block, Sec. 2.1), based on which we train the initial phonetic-semantic joint embedding with the audio words in the training audio dataset (upper left block, Sec. 2.2). The training of SpeechBERT (or a shared BERT model for both text and audio) is then in the bottom of the figure, including pre-training (Sec 2.3) and fine-tuning on SQA (Sec. 2.4). The details are given below.

2.1. Text BERT Pre-training

Here we follow the standard Text BERT pre-training procedure [10]. For a sentence in the training text set with \( n \) tokens \( T = \{t_1, t_2, ..., t_n\} \), we represent them as vectors \( E_{\text{text}} = \{ e_1, e_2, ..., e_n \} \) and sum them with corresponding positional and sentence segment embeddings to get \( E_{\text{emb}} \) to be fed into the multi-layer Transformer. Masked language model (MLM) task is performed at the output layer of the Text BERT by randomly replacing 15% of vectors in \( E_{\text{emb}} \) with a special mask token vector and predicting the masked tokens at the same position of the output layers. Next sentence prediction (NSP) usually performed in BERT training is not used here because some recent studies indicated it may not be helpful [27, 28, 29, 30].

2.2. Initial Phonetic-Semantic Joint Embedding

For an utterance in the training audio dataset with \( n \) audio words \( X = \{x^{(1)}, x^{(2)}, ..., x^{(n)}\} \), the goal here is to encode these \( n \) audio words into \( n \) embeddings \( E_{\text{audio}} = \{ e_1, e_2, ..., e_n \} \). Let one of the audio words include a total of \( T \) speech feature vectors, \( x = \{x_1, x_2, ..., x_T\} \). The complete process of initial phonetic-semantic joint embedding is in Fig. 2. With the speech features for each audio word \( x = \{x_1, x_2, ..., x_T\} \), we use an RNN sequence-to-sequence autoencoder [15] as in Fig. 2. This includes an audio encoder (low left corner of the figure) transforming the input \( x \) into a vector \( z \) (in red in the middle), and an audio decoder reconstructing the output \( y = \{y_1, y_2, ..., y_T\} \) from \( z \). The autoencoder is trained to minimize the reconstruction error:

\[
L_{\text{recon}} = \sum_k \sum_{t=1}^T \|x_t - y_t\|^2_2, \quad (1)
\]

where \( k \) is the index for training audio words. This process enables the vector \( z \) to capture the phonetic structure information of the audio words but not semantics at all, which is not adequate for the goal here. So we make the vector \( z \) be constrained by a L1-distance loss (lower right) further:

\[
L_{L1} = \sum_k \|z - \text{Emb}(t)\|_1, \quad (2)
\]

where \( t \) is the token label for the audio word \( x \) and \( \text{Emb} \) is the embedding layer of the Text BERT trained in Sec. 2.1. In this way, we use the token label of the audio word to access the semantic information about the considered audio word extracted by the Text BERT as long as it is within the vocabulary of the Text BERT. So, the autoencoder model learns to keep the phonetic structure of the audio word so as to reconstruct the original speech features \( x \) as much as possible, but at the same time it learns to fit to the embedding distribution of the Text BERT which carries plenty of semantic information for the word tokens. This makes it possible for the model to learn a joint embedding space integrating both phonetic and semantic information extracted from both audio and text datasets.

2.3. MLM Pre-training for SpeechBERT with Both Text and Audio Data

This is the lower left block of Fig. 1 with details in Fig. 3 (a). Here we pre-train the SpeechBERT with the MLM task using both text and audio datasets, before fine-tuning it on the downstream QA task.

As in Fig. 3 (a), for the SpeechBERT learning to take embeddings for both discrete text words and continuous spoken words, we jointly optimize the MLM loss with the mixture of both audio and text data. The same training target for text MLM as described in Sec. 2.1 is used here. For audio data input, after obtaining the phonetic-semi joint embeddings \( E_{\text{audio}} = \{ e_1, e_2, ..., e_n \} \) for each utterance as in Sec. 2.2, we also randomly replace 15% of those embeddings with mask token vectors as we do in Sec. 2.1. With the supervised setting, we can similarly predict the corresponding tokens behind the masked spoken words. During training, we freeze the audio encoder in Fig. 2 to speed up the process, while have the text word embedding unfrozen to keep the joint audio-and-text embeddings flexible, considering the earlier experiences reported for end-to-end SLU [31].

2.4. Fine-tuning on Question Answering

This is the last block in Fig. 1, with details in Fig. 3 (b). Here the downstream QA task is fine-tuned to minimize the loss for predicting the correct start/end positions of the answer span, as proposed in BERT [10]. By introducing a start and an end vector \( S, E \in \mathbb{R}^H \), we compute a dot product for \( S \) or \( E \) with each hidden vector \( T_i \in \mathbb{R}^H \) for audio word \( i \) from the
3. Experimental Setup

3.1. Data and Evaluation

The SpeechBERT model was trained on the Spoken SQuAD dataset [14], in which the text for all audio paragraphs and all questions are from original SQuAD [12] dataset. It also includes SQuAD format ASR transcripts plus 37k question-answer pairs as the training set and 5.4k as the testing set. This is smaller than the official SQuAD dataset (with 10.6k questions in its development set) since 5.2k of questions in SQuAD for which the answers couldn’t be found in the ASR transcripts were removed and not shown in the SpeechBERT testing set. These removed questions in SQuAD were collected to form another testing set referred to as SQuAD-lost (lost by ASR errors). This is helpful below in evaluating whether the end-to-end approach can better handle the questions with incorrectly recognized answer spans. All questions in SQuAD-lost share the same audio files as the Spoken SQuAD testing set.

The evaluation metrics we used are the Exact Matched (EM) percentage and F1 score for the word tokens as in the normal QA tasks, but for SQuAD-lost frame-level F1 and Audio Overlapping Score (AOS) [14] based on the boundaries of the answer spans as illustrated in Figure 4 were used instead. For the latter case, the boundaries for the audio words were found by forced alignment with Kaldi [32], based on which the start/end points of the answer for both the ground truth and predicted results in training and testing sets were obtained. The ground truth text of the testing set was used only in evaluation.

3.2. Model Setting and Training

3.2.1. Initial Phonetic-semantic Joint Embedding

For the autoencoder in Fig. 2, we used a bidirectional LSTM as the encoder and a single-directional LSTM as the decoder, both with input size 39 (MFCC-dim) and hidden size 768 (BERT embedding-dim). Two layers of the fully-connected network are added at the encoder output to transform the encoded vectors to fit the BERT embedding space. We directly used the audio from Spoken SQuAD training set to train this autoencoder.

3.2.2. Text BERT and SpeechBERT

We used the PyTorch implementation1 of BERT to build the SpeechBERT model with 12 layers of bert-base-uncased setting. We randomly initialized a new embedding layer with our vocabulary set counted in the dataset rather than using the WordPiece tokenizer [33] in processing the text because it is inconsistent with the audio word units. The official pre-trained weights were loaded as the weights of the BERT model. We first trained the MLM task with the text of Spoken SQuAD training set for three epochs. Next, we used the Spoken SQuAD training set by directly feeding text and audio into the BERT model. After pre-training, we fine-tuned the model with the Spoken SQuAD training set for another two epochs. The other hyper-parameters used were identical to the original PyTorch implementation.

4. Experimental Results

4.1. Spoken SQuAD (no ASR errors in answer spans)

Table 1 Section (I) is for models trained on text data, with rows (a)(b)(c) for different QA models trained on the same SQuAD (in text), while column (A)(B) respectively for testing sets of SQuAD (text) and Spoken SQuAD (ASR). Rows (a)(b)(c) showed the prior arts and the superiority of BERT (rows (c) vs (a)(b)), and the serious performance degradation for these models when tested on ASR output directly (column (B) vs (A)). Row (d) is for BERT trained on ASR transcriptions of Spoken SQuAD, or the “cascade” of ASR and BERT, where we see BERT performed much better for ASR output if trained on ASR output (rows (d) vs (c) for column (B)).

Section (II) is for the end-to-end QA model proposed here, with row (e) for the SpeechBERT trained on Spoken SQuAD. We see the end-to-end model is still 4-5% lower than the “cascade” architecture (row (e) vs (d)), although much better or comparable to prior models directly used on ASR output (rows

---

1https://github.com/huggingface/transformers
Table 1: Experimental results on Spoken SQuAD. Sections (I)/(II)/(III) are respectively for models trained on text, end-to-end models trained on audio, and ensemble models; while column (A)/(B) are respectively for text and ASR testing sets.

<table>
<thead>
<tr>
<th>Models and Training set</th>
<th>(A) Text</th>
<th>(B) ASR</th>
</tr>
</thead>
<tbody>
<tr>
<td>(I) trained on text</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(a) BiDAF on Text [34]</td>
<td>58.40</td>
<td>69.90</td>
</tr>
<tr>
<td>(b) DrQA on Text [35]</td>
<td>62.84</td>
<td>73.74</td>
</tr>
<tr>
<td>(c) BERT on Text [10]</td>
<td>76.90</td>
<td>85.71</td>
</tr>
<tr>
<td>(d) BERT on ASR [10]</td>
<td>-</td>
<td>56.28</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(II) End-to-end trained on Audio</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(e) SpeechBERT (proposed)</td>
<td>51.19</td>
<td>64.08</td>
</tr>
<tr>
<td>(f) SpeechBERT w/o MLM</td>
<td>46.02</td>
<td>59.62</td>
</tr>
<tr>
<td>(g) SpeechBERT tested on better boundaries</td>
<td>53.42</td>
<td>66.27</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>(III) Ensembled models</th>
<th>EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>(h) ensembled [c(e) plus [d]]</td>
<td>60.37</td>
<td>71.75</td>
</tr>
<tr>
<td>(i) ensembled [d(e) plus [d]]</td>
<td>57.88</td>
<td>69.29</td>
</tr>
</tbody>
</table>

Table 2: Experimental results on SQuAD-lost, Spoken SQuAD and Total for the cascade and end-to-end models.

<table>
<thead>
<tr>
<th>Model</th>
<th>Spoken SQuAD</th>
<th>SQuAD-lost</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1</td>
<td>AOS</td>
<td>F1</td>
</tr>
<tr>
<td>Cascade (d)</td>
<td>66.56</td>
<td>63.87</td>
<td>30.78</td>
</tr>
<tr>
<td>End-to-end (e)</td>
<td>62.76</td>
<td>59.70</td>
<td>37.31</td>
</tr>
</tbody>
</table>

(e) vs (a)(b)(c)). Considering the high level of the difficulty for the end-to-end SQA model to learn the sophisticated semantic knowledge out of the relatively long audio signals directly in one model without using the word tokens from ASR, the results here showed the promising potential for end-to-end approaches for SQA when better data, better model and better training become possible.

5. Concluding Remarks

Audio signals are in form of phonetic structures, while carrying semantics. ASR has long been used to transform the phonetic structures into word-level tokens carrying semantics, but with inevitable errors causing troubles to the downstream application tasks. With reasonable performance of the end-to-end SQA task, the proposed SpeechBERT was shown to be able to do similar transformation, but the semantics were somehow directly tuned to the downstream task, or question answering here, bypassing the intermediate problem of ASR errors. This concept (and the SpeechBERT) is definitely useful to many other spoken language processing tasks to be studied in the future.