Abstract
A lot of work has been done recently to build sound language models for the textual data, but not much such has been done in the case of speech/audio type data. In the case of text, words can be represented by a unique fixed-length vector. Such models for audio type data can not only lead to great advances in the speech-related natural language processing tasks but can also reduce the need for converting speech to text for performing the same. This paper proposes a novel model architecture that produces syntactically, contextualized, and semantically adequate representation of varying length spoken words. The performance of the spoken word embeddings generated by the proposed model was validated by (1) inspecting the vector space generated, and (2) evaluating its performance on the downstream task of next spoken word prediction in a speech.

1 Introduction
Natural Language Processing (NLP) primarily deals with the communications between humans and how the computers interpret it. There are several methods in which humans and computers can converse, like speaking (audio) and writing (plain text). At present, research in the field of NLP has advanced a lot to attain a good understanding of textual data but there are still some ways to go to properly contemplate the audio/speech data.

The Word2Vec model by Mikolov et al. (2013) is widely used for learning sound vector representations of words using the context words called, Word Embeddings. Word embeddings are extensively used in NLP applications since they have proven to be an extremely informative representation of the textual data. The Word2Vec model successfully transforms textual words from its raw form to semantically and syntactically correct, fixed dimensional vectors. Compared to textual data, not much work has been done to build language models like Word2Vec to generate a similar representation for the audio type data. It would not only reduce the time to convert speeches to text but also expand the field of NLP applications for audio type data.

Such word representations of the spoken words can be widely used in NLP tasks like Automatic summarization (Kägebäck et al., 2014), Machine translation (Jansen, 2017), Named entity resolution (Luo et al., 2019), Sentiment analysis (Liu, 2017), Information retrieval (Rekabsaz et al., 2017), Speech recognition (Palaskar et al., 2019), Question answering (Tapaswi et al., 2016) etc.

There has not been a lot of work done in this field because of several major problems. First, not much large, reliable, clean datasets are available publicly on which the spoken word language models can be trained. Second, spoken words unlike textual words have a different meaning when they are spoken in a different tone, and incorporating tone in language models exponentially increases the difficulty of building such language models. Such models also face difficulties such as different people can have different pronunciations, tones, and pauses for the exact same words.

The proposed model, instead of using fixed length audio files with multiple word utterances, aims at generating syntactically contextualized, and semantically adequate vector representation of the variable length audio files, where each file corresponds to a single spoken word in a speech, and further validates the generated vector space by evaluating its performance on the downstream next word prediction task. To further increase the interpretability, this paper also provides illustrations of the vector space generated by the proposed model.
2 Related Work

A lot of work has been done in the field of NLP to give textual words sound representations. There have been numerous approaches to depict textual data and derive the relationship between words of different languages. The recent advances in deep neural network methods have led to numerous attempts to improve the ability of computers to interpret the textual data. With the amount of data growing exponentially, it is paramount that the data collected is comprehended to the best extent.

Word2Vec (Mikolov et al., 2013) model is extensively used to learn word embeddings using a shallow neural network. Word2Vec model loops through all the words of each sentence in the corpus and either it tries to predict neighboring words based on the current word (Guthrie et al., 2006), or it tries to predict the current word using the context words (Wang et al., 2017). After looking at the nearby and the context words, the model aims to generate vector representations of each of the words such that similar context words occupy a close spatial position in the vector space. Mathematically speaking, the cosine similarity between such words should be approaching 1.

There exists literature that corroborates the possibility to apply deep learning to transform spoken word segments into fixed dimensional vectors. Chung et al. (2016), uses fixed-length audio files, and passes them through a Sequence-to-Sequence Autoencoder (SA) and Denoising SA (DSA) to generate word embeddings. Chung and Glass (2017) used a dataset of 500 hours of speeches from multiple speakers divided into fixed audio segments aiming to achieve audio files in the form of separate words. This work uses an RNN Encoder-Decoder Framework and similar to the previous work, uses a Sequence-to-Sequence Auto-Encoder. Tagliasacchi et al. (2020), proposed an audio2vec model which was built on top of the Word2Vec models (Skip-gram & CBOW) to reconstruct spectrogram slices using the contextual slices and temporal gaps. They were able to show that Audio2Vec performed better than the other existing fully-supervised models. The aforementioned work was closest to what is being proposed in this paper.

There exist limited works in the field of verbal data processing. Audio data visualization in the form of vectors was done by (Chung et al., 2016), which used fixed-length audio files and passed them through a Sequence-to-Sequence Autoencoder (SA) and Denoising SA (DSA) to generate word embeddings. However, the main aim of the work was to show that phonetically similar words will have close spatial representations in the vector space. The paper also worked on raw audio files and tried to learn the semantic relation between spoken words but failed to meet the result standards similar to those by GloVe on Wikipedia. These are the issues that are incurred when raw audio files are used, the reason being that they need to pass through an autoencoder and hence need to be of fixed dimensions.

Following the above work, Chung and Glass (2017) has used a dataset of 500 hours of speeches from multiple speakers divided into fixed audio segments aiming to achieve audio files in the form of independent words. This work goes on to compare the results based on 13 different comparison measures with the GloVe model producing results consistently lower than the existing models. The reason for the same was that the use of fixed length audio files fails to capture the English words properly. Not all words have fixed audio lengths and different speakers give different emphasis to the various syllables.

3 Model

This section will describe the architecture of the proposed model. The proposed model uses sequential utterances of words from a speech to learn their corresponding contextualized representations. These learned contextualized representations capture the semantic and syntactic properties of these spoken words. The input to the model is a speech \( S \). This speech is split into individual spoken word utterances (independent audio files). The proposed model used audio spectrograms for representing the audio files of these spoken word utterances. An audio spectrogram is a visual representation of sound. So to get spectral representations, all the spoken word utterances are converted to their corresponding spectrogram images (which depicts the spectral density of a sound (in our case an utterance) with respect to time). The spoken word utterance spectrograms are represented by \( W_i \) as shown in equation 1.

\[
S = [W_1, W_2, ..., W_n], n \in \mathbb{R}
\]
In the above equation $n$ represents the total number of spoken words present in the speech and $W_1 \in \mathbb{R}^{l_1 l_2 l_3}$ where $l_1, l_2 \& l_3$ are the width, length and no. of color channels of the spectrogram image.

Words have different meanings when they are spoken in different contexts. To capture the corresponding to spoken words, the proposed model uses a context window of size $m$. So the representation of a spoken word (target word) is learned based on $m$ spoken words after and before it. This context window of size $m$ slides over the whole speech having a target spoke word $W_t$ (where $1 \leq t \leq n$) at the middle and $m$ context spoken words before and after it (a total of $2m$ context words). These context spoken words are represented by $W_t + j$ where $-m \leq j \leq m \& j \neq 0$.

Next, the model passes all the pairs of the target spoke word spectrograms $W_t$ with its corresponding context spoken word spectrograms $W_{t+j}$ into a convolutional autoencoder individually to learn the contextual representation of the target spoken word corresponding to $W_t$. The convolutional autoencoder is composed of two independent neural networks namely, an encoder network and a decoder network. The encoder network is represented by $f_\phi$ and the decoder network is represented by $g_\theta$, where $\phi$ and $\theta$ are the learnable parameters corresponding to both the networks. Both $f_\phi$ & $g_\theta$ are used to extract the spatial features of the input spectrogram w.r.t to the output spectrogram. The target spoke word spectrogram $W_t$ is given as input to the encoder network, which outputs a latent representation $h$. This latent representation is then given as input to the decoder network, i.e.

$$h = f_\phi(W_t) = \sigma(W_t * \phi) \quad (2)$$
$$W_{t+j} = g_\theta(h) = \sigma(h * \theta) \quad (3)$$
$$W_{t+j} = g_\theta(f_\phi(W_t)) \quad (4)$$

In the equations 2 & 3, $(*)$ represents the convolution operator, $\sigma$ is the LeakyReLU activation function. The encoder network $f_\phi$, consist of two convolutional layers on top of the input image spectrogram. These convolutional layers are used for extracting hierarchical location invariant spatial features. The output of the last convolutional layer in $f_\phi$ is then flattened and passed to a $d$-dimensional dense layer ($h$). This dense layer ($h$) is the embedding layer which learns the contextual representation of the spoken word corresponding to input $W_t$ (contextualized on the context spoken word spectrograms). The decoder
network takes the embedding layer \((h)\) as input and generates a reconstruction image \(W_r^t\) by passing \((h)\) through a dense and two transpose convolutional layers. The \(d\)-dimensional embedding layer \((h)\) learns an efficient contextualized representation of the word corresponding to \(W_t\) by minimizing the loss function \(L\) (shown in equation 5). In the equation below, \(N\) represents the batch size and \(m\) represents the size of the context window.

\[
L(\phi, \theta) = \frac{1}{N} \sum_{t=1}^{N} \frac{1}{2m} \sum_{j=-m;j\neq0}^{m} ||g_\theta(f_\phi(W_t)) - W_{t+j}||_2^2
\] (5)

The loss function defined above helps the latent embedding to learn the contextual relationship between the target spoken word spectrograms and its corresponding context by calculating a reconstruction loss between the reconstructed image \(W_r^t\) and the corresponding contextual spectrograms \(W_{t+j}\). Since a word spoken in different tones has different spectrograms, the model also captures the tone in which the words are uttered in its contextual embedding. So in summary the proposed model can not only incorporate context in its spoken word representations but can also incorporate its tone.

These learned representations can be of great use in speech-related natural language processing tasks like the next spoken word prediction. The next spoken word prediction task is the task of predicting the next word to be spoken, given the previous sequence of spoken words. This task is a baseline task for evaluating the performance of any type of word embeddings (textual or spoken), so the proposed model was also evaluated based on its performance on this task.

Let the \(i^{th}\) sentence of the speech \(S\) is represented by \(s_i\), where each sentence \(s_i\) is represented by a sequence of \(n\) spoken words \([w_1, w_2, ..., w_n]\). In this task, given an initial sequence of words \([w_1, w_2, ..., w_k]\), the model tries to predict the word that should come next (word by word). This is done by computing the probability of occurrence of several words, i.e.

\[
P(w_1, ..., w_{k+1}) = \prod_{i=1}^{i=k} P(w_i/w_1, ..., w_{i-1})
\] (6)

Long Short Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) are the baseline models used in the next spoken word prediction tasks. They are a special type of recurrent neural network that uses three different gates (input, output and forget) to persist information in a sequence. The LSTM takes a sequence \([w_1, w_2, ..., w_n]\) as input and couple each timestamp to an input \(i_j\), cell state \(c_j\) and an output gate \(o_j\). The hidden state at time \(j\) by the LSTM is denoted by \(h_j\), can be computed as:

\[
i_j = \sigma(W_{iw}w_j + W_{ih}h_{j-1} + b_i)
\] (7)

\[
f_j = \sigma(W_{fw}w_j + W_{fh}h_{j-1} + b_f)
\] (8)

\[
o_j = \sigma(W_{ow}w_j + W_{oh}h_{j-1} + b_o)
\] (9)

\[
g_j = \tanh(W_{gw}w_j + W_{gh}h_{j-1} + b_g)
\] (10)

\[
c_j = f_j \odot c_{j-1} + i_j \odot g_j
\] (11)

\[
h_j = o_j \odot \tanh(c_j)
\] (12)

Here, \((\odot)\) represents the element-wise product, \(\sigma\) represents a sigmoid function, \(W_{iw}\) represents a transformation matrix from the input to the LSTM states, \(W_{ih}\) represents a recurrent transformation matrix between hidden states \(h_j\) and \(b\) represents the bias vector. The proposed model will be evaluated on the next word prediction task using an LSTM.
4 Evaluation Setup

4.1 Dataset

The proposed model uses Trump speeches (Audio and word transcription)\(^1\) dataset for training and testing. This dataset contains audio files and their corresponding word split JSON file. This JSON file contains a direct mapping between each word spoken and the duration in which it was spoken. This mapping was used to split the full audio file into multiple audio files for each word spoken. The audio transcript \(T\) was split into sentences \(S\) and these sentences were further split into words \(W\). Then a mapping was made between the target word spoken \((t)\) and its contextual spoken words \((c)\). A context window of size 2 was used. This context mapping was used to create input-output pairs for the proposed model as shown in the algorithm shown below.

A 1 hour 20 min speech by Donald Trump was split into 5818 audio files corresponding to 5818 spoken words and generated 23,266 context mappings. These contextually mapped images were used as input and output of the proposed model.

---

**Figure 2: Algorithm Used for Creating Contextual Mappings**

```
// Algorithm
Data: T - Audio transcript
Result: Spoken word vectors
initialization:
c ← split(T', ')
for all s ∈ C do
  x ← split(s, ');
  w ← w ∪ \{x\};
end
forall all j ∈ w do
  for all k ∈ j do
    context ← window(k);
    d ← d ∪ \{context\};
  end
end
for all row ∈ d do
  input ← row[0];
  output ← row[1];
autoencode(input, output);
end
```

---

4.2 Training Details

The proposed model was evaluated on 10% of the data, and the rest was used for training. From the training set, 10% data was used as the validation set. Both the proposed model and the LSTM model was trained for 50 epochs having a mini-batch size of 5. For optimization, adam optimizer was used having an initial learning rate of 0.01. Early stopping with the patience of 5 epochs and dropout with a dropout rate of 0.7 was used to avoid over-fitting. The size of the latent representation \(d\) was set to 16. The size of the filters in the convolutional and de-convolutional layers was set to \((4\times4)\). The input size of the spectrogram images were \((1004\times644\times2)\) and the context window size of 2 was used.

4.3 Results

The performance of the proposed model was validated by (1) by inspecting the vector space generated and (2) by evaluating the performance on the next word prediction task. The inspection of the vector space was done by comparing the vector space generated by the proposed model with the vector space generated by the Word2Vec (Mikolov et al., 2013) model. The Word2Vec model was trained on the transcript of the speech on which the proposed model was trained. To visualize the performance of the proposed model, the dimensionality of the audio vectors \((16 \text{- dimensional})\) was reduced, to plot the

---

\(^1\)Dataset - https://www.kaggle.com/etaifour/trump-speeches
- audio-and-word-transcription/metadata
words in a $2 - dimensional$ vectors space. This reduction in dimensionality was done using principal component analysis (PCA) (Gewers et al., 2018).

The vector space generated by the proposed model and by the Word2Vec model are illustrated in figure 3 & 4 respectively. Figure 3 demonstrates that the spoken words having similar semantics or the spoken words usually used together in the conversational context are present in close proximity in the vector space generated by the proposed model. This validates the proposed model’s capability to capture the syntactic and semantic relationships between different spoken words. On comparison of vector space generated by the proposed model with that generated by the Word2Vec model demonstrates that both the vector space generates vectors having similar trends and relationships between the spoken words.

To further validate the proposed model, it’s performance was evaluated on the next spoken word prediction task. For this, the speech dataset was segmented into sentences. $16 - dimensional$ contextualized spoken word representations were generated using the proposed model for every word spoken in the sentences. To create uniformity in the length of the sentences, all of them were padded with $16 - dimensional$ zero vectors such that the length of all the sentences becomes equal to the longest sentence (which in our case was 50). The LSTM model described above was used to predict the last few words of each sentence by taking all the previous words of the sentence as its context. The mean cosine similarity between all the predicted next spoken word and the actual spoken word was used as the accuracy metrics for evaluating the performance of the proposed model on the next spoken word prediction task. The cosine similarity measures the similarity between two non-zero vectors of the same vector space. The proposed model performed comparably to the Word2Vec model in establishing syntactic and semantic relationships between spoken words by giving a mean cosine similarity of 0.795 compared to Word2Vec giving a mean cosine similarity of 0.847 as shown in table 1.

The observed mean cosine similarity of the proposed model was marginally less than the Word2Vec model, which could be explained by a number of reasons. The primary reason being the ever-changing vocal tract of human beings. A person cannot say the same word in the same exact tone twice, let alone
different speakers. This means that there will always be some marginal difference in the frequency-time
distribution of each word. This would indeed reduce the mean cosine similarity of the proposed model,
but nonetheless, will still capture the semantic relations. However, as evident from the results above
we can see that the algorithm proposed in this paper produced better results than the other previously
proposed models.

5 Conclusion

The proposed model, not only was able to successfully generate semantically and syntactically accurate
spoken word representation but was also able to perform adequately on the downstream task of next
spoken word prediction. The proposed model trained on the spoken words was compared with the
widely used Word2Vec model trained the text of the corresponding spoken words. Despite the fact that
the word representation in the case speech/audio depend highly on the tone of words spoken (which is
not the case with text), the proposed model was able to perform comparably to Word2Vec on the next
word prediction task.

References

Yu-An Chung and James Glass. 2017. Learning word embeddings from speech.
Yu-An Chung, Chao-Chung Wu, Chia-Hao Shen, Hung-Yi Lee, and Lin-Shan Lee. 2016. Audio word2vec:
Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder.
Felipe L. Gewers, Gustavo R. Ferreira, Henrique Ferraz de Arruda, Filipi Nascimento Silva, Cesar H. Comin,
Diego R. Amancio, and Luciano da F. Costa. 2018. Principal component analysis: A natural approach to data
exploration. CoRR, abs/1804.02502.
David Guthrie, Ben Allison, Wei Lu, Louise Guthrie, and Yorick Wilks. 2006. A closer look at skip-gram
modelling. In LREC, pages 1222–1225.
Sepp Hochreiter and Jrgen Schmidhuber. 1997. Long short-term memory. Neural Computation, 9(8):1735–1780.
Stefan Jansen. 2017. Word and phrase translation with word2vec. CoRR, abs/1705.03127.
Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using
continuous vector space models. In Proceedings of the 2nd Workshop on Continuous Vector Space Models
and their Compositionality (CVSC), pages 31–39, Gothenburg, Sweden, April. Association for Computational
Linguistics.
Haixia Liu. 2017. Sentiment analysis of citations using word2vec. CoRR, abs/1704.00177.
Ying Luo, Hai Zhao, and Junlang Zhan. 2019. Named entity recognition only from word embeddings.
Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in
vector space.
Shruti Palaskar, Vikas Raunak, and Florian Metze. 2019. Learned in speech recognition: Contextual acoustic
word embeddings. CoRR, abs/1902.06833.
Navid Rekabsaz, Bhaskar Mitra, Mihai Lupu, and Allan Hanbury. 2017. Toward incorporation of relevant docu-
ments in word2vec. CoRR, abs/1707.06598.
M. Tagliasacchi, B. Gfeller, F. d. C. Quirty, and D. Roblek. 2020. Pre-training audio representations with self-
supervision. IEEE Signal Processing Letters, 27:600–604.
Makarand Tapaswi, Yukun Zhu, Rainer Stiefelhagen, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2016.
Movieqa: Understanding stories in movies through question-answering. In The IEEE Conference on Computer
Vision and Pattern Recognition (CVPR), June.
Qi Wang, Junchang Xu, Hong Chen, and Ben He. 2017. Two improved continuous bag-of-word models. In 2017
International Joint Conference on Neural Networks (IJCNN), pages 2851–2856. IEEE.