INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

SpeechYOLO: Detection and Localization of Speech Objects

Yael Segal[*], *Tzeviya Sylvia Fuchs*[*], *Joseph Keshet*

Bar-Ilan University, Ramat Gan, Israel
{segalya, fuchstz, jkeshet}@cs.biu.ac.il

Abstract

In this paper, we propose to apply object detection methods from the vision domain on the speech recognition domain, by treating audio fragments as objects. More specifically, we present SpeechYOLO, which is inspired by the YOLO algorithm [1] for object detection in images. The goal of SpeechYOLO is to localize boundaries of utterances within the input signal, and to correctly classify them. Our system is composed of a convolutional neural network, with a simple least-mean-squares loss function. We evaluated the system on several keyword spotting tasks, that include corpora of read speech and spontaneous speech. Our system compares favorably with other algorithms trained for both localization and classification.

Index Terms: keyword spotting, event detection, speech processing, convolutional neural networks

1. Introduction

Recently automatic speech recognition (ASR) has became ubiquitous in many applications. While ASR systems like *Deep-Speech 2* [2] and *wav2letter* [3] reached amazing results in transcribing read and conversational speech, sometimes it is desired to spot and locate a predefined small set of words with extremely high accuracy. For example, services like *Google Now* or *Apple's Siri* can be activated by pronouncing "OK Google" or "Hey Siri", respectively [4, 5]. It is also used by intelligence services to accurately find specific keywords while monitoring suspected phone calls. The task of detecting and localizing words can be used to automatically analyze the diadochokinetic articulatory task [6, 7] that is used to analyze pathological speech and hence cannot be performed effectively with ASR systems. In this work we present an end-to-end system that goes from a speech signal to the transcription and alignment of given keywords (this is in contrast to the spoken term detection task that makes predictions on keywords that it has not been trained on).

Our architecture performs both detection and localization of these predefined keywords. Previous works typically focus on only one of these two challenges. Namely, algorithms would either predict what words appear in a given utterance, thus performing detection [4, 5], or are given the audio signal and the target transcription and align them, thus performing localization mostly using forced alignment [8, 9]. Keshet *et al.* [10] proposed to use the confidence of a phoneme aligner and an exhaustive search to detect and localize terms that are given by their phonetic content.

In the vision domain, object detection algorithms combine the two aforementioned tasks: detection of the desired object and its localization in the image. Specifically, the YOLO [1] and SSD [11] algorithms identify objects in an image using bounding boxes. Inspired by the idea of using bounding boxes for object detection in images, we propose to identify speech objects in an audio signal. More specifically, consider the word classification task as a form of object detection for a speech signal.

Palaz *et al.* [12] presented work that is most similar to ours. Their algorithm was trained to jointly locate and classify words. However, they used a weakly supervised setting and did not use word alignments, and hence were unable to perfectly predict the whole time-span of the predicted words. In our work, however, our goal is both to detect and to locate the entire span of every word, so both tasks' results are strongly accurate.

This paper is organized as follows. In Section 2 we formally introduce the classification and localization problem setting. We present our proposed method in Section 3, and in Section 4 we show experimental results and various applications of our derived method. Finally, concluding remarks and future directions are discussed in Section 5.

2. Problem Setting

The input to our system is a speech utterance. The input speech utterance is represented as a series of acoustic feature vectors. Formally, let $\bar{\mathbf{x}} = (\mathbf{x}_1, \ldots, \mathbf{x}_T)$ denote the input speech utterance of a fixed duration T, where each $\mathbf{x}_t \in \mathbb{R}^D$ is a D-dimensional vector for $0 \leq t \leq T$. We further define the lexicon $\mathcal{L} = \{k_1, k_2, ..., k_L\}$ to be the target set of L keywords or terms that may appear in the audio signal $\bar{\mathbf{x}}$. Note that the utterance does not necessarily contain any of these keywords or may contain several of them. In our setting the speech objects are the L keywords, but the model proposed here is not limited to specific keywords and can be used to detect and localize any audio or speech object, e.g., the syllables */pa/, /ta/,* and */ka/* in the diadochokinetic articulatory task. Our goal is to spot all the occurrences of the keywords in a given utterance $\bar{\mathbf{x}}$ and estimate their corresponding locations.

We assume that N keywords were pronounced in the utterance $\bar{\mathbf{x}}$, where $N \geq 0$. Each of these N events is defined by its lexical content and its time location. Each such event e is defined formally by the the tuple $e = (k, t^k_{\text{start}}, t^k_{\text{end}})$, where $k \in \mathcal{L}$ is the actual keyword that was pronounced, and t^k_{start} and t^k_{end} are its start and end times, respectively. Our goal is therefore to find all the events in an utterance, so that for each event the correct object k is identified as well as its beginning and end times.

3. Model

As previously mentioned, our model is inspired by the YOLO model [1]. We now describe our model formally. Our notation is schematically depicted in Fig. 1. We assume that the input utterance $\bar{\mathbf{x}}$ is of a fixed size T (1 second in our setting). We divide the input-time to C non-overlapping equal sections called *cells* ($C = 6$ in our setting). Each cell is in charge of detecting a single event (at most) in its time-span. That is, the i-th cell, denoted c_i, is in charge of the portion $[t_{c_i}, t_{c_{i+1}} - 1]$, where t_{c_i} is the start-time of the cell and $t_{c_{i+1}} - 1$ is its end-time, for

[*]These authors contributed equally to this work

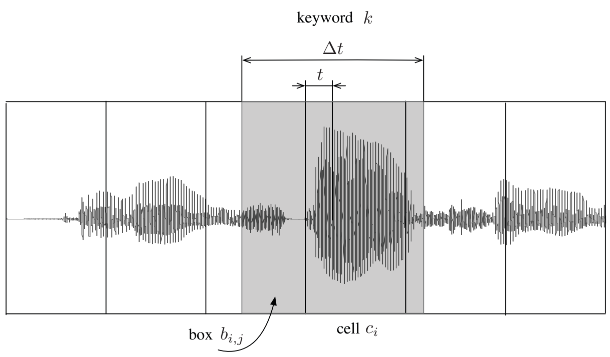


Figure 1: *The notation used in our paper. The keyword "star" is found within cell c_i. One of the timing boxes $b_{i,j}$ is depicted with a shaded box, and it defines the timing of the keyword relative to the cell's boundaries.*

$1 \le i \le C$. The cell estimates the probability $\Pr(k|c_i)$ of each keyword $k \in \mathcal{L}$ to be uttered within its time-span. We denote the estimation of this probability by $p_{\mathrm{c}_i}(k)$.

The cell is also in charge of localizing the detected event. The localization is defined relative to the cell's boundaries. Specifically, the location of the event is defined by the time $t \in [t_{c_i}, t_{c_{i+1}} - 1]$, which is the center of the event relative to the cell's boundaries, and Δt, the duration of the event. Note that Δt can be longer than the time-span of the cell. Using this notation the event spans $[t_{c_i} + t - \Delta t/2, t_{c_i} + t + \Delta t/2]$.

In order to localize effectively, each cell is associated with B *timing boxes* (called *bounding boxes* in the YOLO literature). Each box $b_{i,j}$ of the cell c_i tries to independently localize the event and estimate the probability of the timing given the presumed keyword, $\Pr(t, \Delta t|k)$. It is defined by the tuple $(t_j, \Delta t_j, p_{\mathrm{b}_{i,j}})$, where $p_{\mathrm{b}_{i,j}}$ is the confidence score of the localization $t, \Delta t$ and it can be considered as an estimation of the probability $\Pr(t, \Delta t|k, c_i)$.

We now turn to describe the model's inference. The inference for each cell is performed independently. For the i-th cell, c_i, the chosen event is composed of the keyword k' and the timing $t', \Delta t'$ that maximizes the conditional probabilities:

$$(k', t', \Delta t') = \arg\max_{k,t,\Delta t} \Pr(k, t, \Delta t|c_i) \tag{1}$$

$$= \arg\max_{k,t,\Delta t} \Pr(k|c_i)\ \Pr(\mathrm{t}, \Delta \mathrm{t}|k, c_i). \tag{2}$$

The first conditional probability in Eq. (2) is $p_{\mathrm{c}_i}(k)$, whereas the second conditional probability is $p_{\mathrm{b}_{i,j}}$ of box $b_{i,j}$. Since there are L keywords and B boxes the search space reduces to $L \times B$ elements, hence it is very efficient:

$$\max_{k \in \mathcal{L}}\ \max_{1 \le j \le B}\ p_{\mathrm{c}_i}(k)\ p_{\mathrm{b}_{i,j}}.$$

Finally, the event is considered to exist in the cell if its conditional probability from above is greater than a threshold, θ.

We conclude this section by describing the training procedure. Our model, *SpeechYOLO*, is implemented as a convolutional neural network. The initial convolution layers of the network extract features from the utterance while fully connected layers are later added to predict the output probabilities and coordinates. Our network architecture is inspired by PyTorch's [13] implementation of the VGG19 [14] architecture[1], and is presented in Section 4.

[1] `https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py`

The training set is composed of examples, where each example is an event that is composed of the tuple $(\bar{\mathbf{x}}, k, t^k_{\mathrm{start}}, t^k_{\mathrm{end}})$. We denote by $\mathbb{1}^k_i$ the indicator that is 1 if the keyword k was uttered within the cell c_i, and 0 otherwise. Formally,

$$\mathbb{1}^k_i = \begin{cases} 1 & t_{c_i} \le t^k \le t_{c_{i+1}} - 1 \\ 0 & \text{otherwise} \end{cases}.$$

where t^k is defined as the center of event k.

When we would like to indicate that the keyword is not in the cell we will use the notation $(1 - \mathbb{1}^k_i)$.

The model's loss function is defined as a sum over several terms, each of which took into consideration a different aspect of the model, as a follows:

$$\begin{aligned}\bar{\ell}(\bar{\mathbf{x}}, k, t^k_{\mathrm{start}}, t^k_{\mathrm{end}}) &= \lambda_1 \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}^k_i \left(t_j - t'_j\right)^2 \\ &+ \lambda_2 \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}^k_i \left(\sqrt{\Delta t_j} - \sqrt{\Delta t'_j}\right)^2 \\ &+ \sum_{i=1}^{C} \sum_{j=1}^{B} \mathbb{1}^k_i \left(1 - p_{\mathrm{b}_{i,j}}\right)^2 \\ &+ \lambda_3 \sum_{i=1}^{C} \sum_{j=1}^{B} \left(1 - \mathbb{1}^k_i\right) \left(0 - p_{\mathrm{b}_{i,j}}\right)^2 \\ &+ \sum_{i=1}^{C} \sum_{k \in \mathcal{L}} \mathbb{1}^k_i \left(1 - p_{\mathrm{c}_i}(k)\right)^2.\end{aligned}$$

We would like to note that our system is inspired by the first version of YOLO [1]. Further research on YOLO has been conducted in [15] and [16]. It seems, however, that most expansions made to their algorithm are irrelevant for our domain. In [15] the authors' main contributions are the addition of *anchor boxes*, which defines constraints on the shapes of the bounding boxes. This lead to specifying a separate class probability value for every bounding box. This is relevant when dealing with a multidimensional domain, and is less relevant for speech. In their paper, they additionally suggest the usage of a fully convolutional network, i.e. replacing the fully connected layers with convolutions. We found that this yielded inferior results for our dataset. In [16], the main development was the shift from multiclass classification to multilabel classification. This changed the loss function from using regression to using cross entropy instead. This too is irrelevant for our domain.

4. Experiments

We used data from the LibriSpeech corpus [17], which was derived from read audio books. The training set consisted of 960 hours of speech. This corpus had two test sets: *test_clean* and *test_other*, which summed up to 5 hours of speech. The first set was composed of high quality utterances and the second set was composed of lower quality utterances. The audio files were aligned to their given transcriptions using the Montreal Forced Aligner (MFA) [9]. We extracted the Short-Time Fourier Transform (STFT) as features to the sound files using the `librosa` package [18]. These features were computed on a 20 ms window, with a shift of 10 ms.

A target event of the input speech signal can be defined as any discrete part of an utterance that is discernible to a human

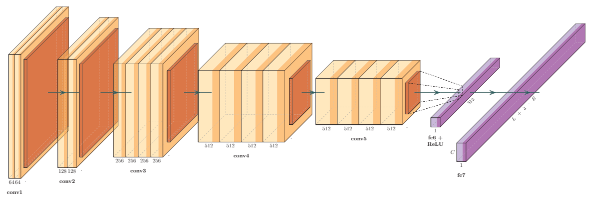


Figure 2: *The detection network has 16 convolutional layers followed by 2 fully connected layers. Every convolutional layer is followed by BatchNorm and ReLU. We pretrained the convolutional layers on the Google Command classification task and then replace the final layer for detection and localization.*

annotator. Hence, events could be defined as a set of words, phrases, phones, etc. It was assumed that only events from the selected lexicon $\mathcal{L}$ are available during training time.

We used a convolutional neural network that is similar to the VGG19 architecture. It had 16 convolutional layers and 2 fully connected layers, and the final layer predicted both class probabilities and timing boxes' coordinates. We denote this architecture as $VGG19^*$. The model is described in detail in Figure 2. For comparison, we also implemented a version of the VGG11 model (denoted by $VGG11^*$), which had less convolutional layers. Both models were trained using Adam [19] and a learning rate of 10^{-3}. We pretrained our convolutional network using the Google Command dataset [20] for $L = 30$. We later replaced the last linear layer in order to perform prediction on a different number of events, and further trained the network. The size of this new final layer is $C \times (L + 3 \cdot B)$.

We divided our experiments into two parts: in the first, we evaluated SpeechYOLO's capability to correctly predict and localize words within an utterance, and compared its performance to other similar systems. Then, we evaluated SpeechYOLO for the keyword spotting task on various domains.

4.1. Word prediction and localization

In this subsection, we evaluated the system's capability to learn word detection and localization. We defined the target events to be the 1000 most common words in the training set ($L = 1000$). It turned out that the average utterance time of a single word in our corpus was approximately 0.2 seconds. To assure that the timing cells properly covered the span of the speech signal, we chose to use $C = 6$ timing cells per utterance of $T = 1$ sec. We arbitrarily set the number of timing boxes per cell to be $B = 2$.

We chose the value of the threshold θ that maximizes the F1 score, which is defined as the harmonic mean of the precision and recall. We evaluated the model's detection capabilities using Precision and Recall. Results are presented in Table 1. It seems that the proposed system was able to correctly detect most of the words, with $VGG19^*$ outperforming $VGG11^*$, due to its size and enhanced expressive abilities.

4.1.1. Prediction and Localization

Due to the uniqueness of our system's aim to both classify and localize words, it is challenging to find an equivalent algorithm to justly compare with. Most other algorithms focus on either one of the tasks, but not on both. The system of Palaz, Synnaeve and Collobert [12] was developed for weakly-supervised word recognition; that is, its aim is to perform word classification and find word position, while training with a BoW supervision. As in [21], we refer to this system as PSC.

Table 1: *SpeechYOLO evaluations with two architectures on both of LibriSpeech's test sets. The threshold value that maximized the F1 score was chosen ($\theta = 0.4$).*

		precision	**recall**	**F1**
test_clean	$VGG11^*$	0.743	0.637	0.686
	$VGG19^*$	**0.836**	**0.779**	**0.807**
test_other	$VGG11^*$	0.547	0.456	0.498
	$VGG19^*$	0.697	0.553	0.617

PSC receives the Mel Filterbanks coefficients as input features. Their architecture is composed of 10 convolutional layers. The final convolution has 1000 output filters for every time span, with every filter corresponding to a word k in the lexicon $\mathcal{L}$. The idea is that the score for word k would be highest in the time span it occurred in. The system is trained using SGD with a learning rate of 10^{-5}.

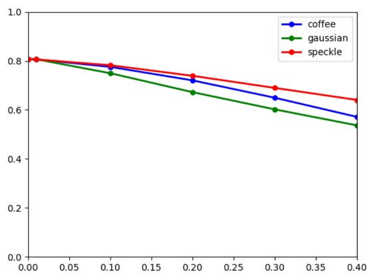
We compared SpeechYOLO's prediction and localization abilities to PSC's, as shown in Table 2. We calculated the F1 measure as before. The *Actual* accuracy measure was calculated as described in [12], and measures localization as well as prediction. For PSC, the *Actual* accuracy was calculated as follows: word detection was performed by thresholding the probability of a word being present in the sequence. For every word k that passed the chosen threshold, we chose the frame in which it received the highest score. We then assessed if this frame was located within the range of k stated by the ground truth alignment. The closest equivalent of this measure for our model was to choose this frame to be the center of the predicted timing box. This value was in turn compared to the ground truth alignment. As before, the threshold θ was chosen to maximize the F1 measure. SpeechYOLO clearly outperformed PSC for both the F1 score and the Actual accuracy measure.

To assess the strength of SpeechYOLO's localization ability, we calculated both systems' average intersection over union (IOU) value with the ground truth alignments. While SpeechYOLO's IOU value clearly outperformed that of PSC, one must remember that PSC was not trained with aligned data.

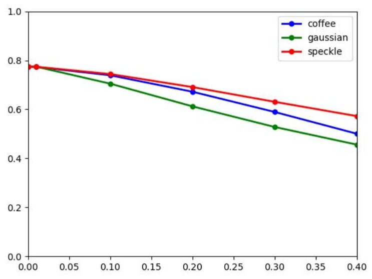
Table 2: *Comparing SpeechYOLO and PSC [12]'s evaluations of the F1 score, Actual accuracy and average IOU value. The threshold value that maximized the F1 score was chosen for every algorithm separately ($\theta = 0.4$).*

	F1	**Actual**	**IOU**
SpeechYOLO	**0.807**	**0.774**	**0.843**
PSC	0.767	0.692	0.3

We further checked the quality of SpeechYOLO's localization capability. To do so, we compared SpeechYOLO with MFA, after both had been trained on LibriSpeech. We tested them on the training set of the TIMIT corpus. TIMIT is a corpus of read speech, and presents a different linguistic context compared to LibriSpeech. The IOU measure was used to compare both algorithm's output alignments with TIMIT's given word alignments for the 1000 most common words in the LibriSpeech training set. In order to predict SpeechYOLO's IOU values, it was assumed that its predictions were perfect. This was due to the fact that SpeechYOLO does not receive transcriptions as an



(a) *F1 score*



(b) *Actual accuracy*

Figure 3: *SpeechYOLO's performances when injecting background noise. The y-axis is the measure and the x-axis is the strength of the noise added (α).*

input, and because our goal was to asses the localization task alone. The IOU of SpeechYOLO on TIMIT was 0.673, while MFA achieved 0.827.

The forced aligner, MFA, performs its alignments using a complete transcription of the words uttered in a speech signal. On the other hand, SpeechYOLO receives no information about the words uttered. Hence, given MFA's extended knowledge, we considered its localization ability as an "upper bound" to ours. Therefore, we found that SpeechYOLO's IOU value, while lower than MFA's, were sufficiently high.

Additionally, an aligner could naturally go wrong if there are incorrect or missing words in its transcription, or alternatively if the audio signal contains long silences or untranscripted noises between words (e.g. a laugh or a cough) [22]. It should be noted that given SpeechYOLO's lack of knowledge about the transcription, these problems do not affect it.

4.1.2. Robustness to noise

We further demonstrated SpeechYOLO's robustness by artificially adding background noise to LibriSpeech's audio files with a relative amplitude α. We injected 3 types of background noises: a coffee shop, gaussian noise, and speckle. In Figure 3 we show SpeechYOLO's F1 score and *Actual* accuracy measures when increasing the α variable, thus intensifying the injected noise. It is apparent that minor amounts of noise do not degrade SpeechYOLO's performances. Note that SpeechYOLO was able to deal even with higher α values, although it yielded somewhat reduced results.

4.2. Keyword spotting

In this part, we evaluate SpeechYOLO on a real-world application: keyword spotting. For evaluation, we use the F1 metric, and the Maximum Term Weight Value (MTWV) metric [23]. MTWV is defined as one minus the weighted sum of the probabilities of miss and false alarm.

4.2.1. LibriSpeech Corpus

We compare SpeechYOLO's keyword spotting capabilities with those of the PSC system. In their work, they use a set of keywords that is a subset of the 1000 words used previously for prediction and localization. The chosen keywords are in Table 2 of [12], and are evaluated on both of LibriSpeech's test sets. A comparison of our results are presented in Table 3. Here too SpeechYOLO's results outperformed those of PSC.

Table 3: *MTWV values for SpeechYOLO and PSC on the keyword spotting task, evaluated on both of LibriSpeech's test sets.*

	SpeechYOLO	**PSC**
test_clean	**0.74**	0.72
test_other	**0.38**	0.27

4.2.2. Spontaneous speech corpus

We now present the results of SpeechYOLO for keyword spotting with spontaneous speech. This is relevant for mobile applications, where a device is activated by a voice command like 'OK Google" or "Hey Siri". To simulate this task, we use a corpus taken from a daily TV show *Good evening with Guy Pines*[2]. This corpus, which we will call "Hi Guy", consists of spontaneous and noisy recordings. In each recording, a celebrity is prompted to utter the phrase *Hi Guy!* These recordings vary greatly in terms of their environment and the speakers within them are highly diverse.

The corpus consists of 880 examples, out of which 445 contain the chosen keyword. We chose the phrase "Hi Guy" to be the keyword that our system searches for. The input length is 3 seconds. The system achieves an *Actual* accuracy of 0.624, and an F1 score of 0.755 (precision: 0.748, recall: 0.761). We find these results to be surprisingly satisfying due to the small size of the dataset and due to the diversity found in the corpus: the audio files are at times extremely noisy, the pronunciation of the speakers vary, and the keyword is sometimes sung instead of being spoken.

5. Conclusions

In this work, we introduce the concept of treating parts of audio signals as objects. We propose the SpeechYOLO algorithm for object detection and localization, and evaluate its performances for both of these tasks. We further show its use for the keyword spotting task. Future work includes expanding its keyword spotting ability for other speech parts. We would also like to extend SpeechYOLO to detect words it has not been trained on.

6. Acknowledgements

We would like to thank *Hi Guy! Guy Pines Communications Ltd.* for providing their data. T. S. Fuchs is sponsored by the Malag scholarship for outstanding doctoral students in the high tech professions.

[2] `https://www.timesofisrael.com/topic/good-evening-with-guy-pines/`

7. References

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[2] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[3] R. Collobert, C. Puhrsch, and G. Synnaeve, "Wav2letter: an end-to-end convnet-based speech recognition system," *arXiv preprint arXiv:1609.03193*, 2016.

[4] G. Chen, C. Parada, and G. Heigold, "Small-footprint keyword spotting using deep neural networks," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4087–4091.

[5] T. Sainath and C. Parada, "Convolutional neural networks for small-footprint keyword spotting," 2015.

[6] S. G. Fletcher, "Time-by-count measurement of diadochokinetic syllable rate," *Journal of speech and hearing research*, vol. 15, no. 4, pp. 763–770, 1972.

[7] J. Westbury and J. Dembowski, "Articulatory kinematics of normal diadochokinetic performance," *Annual Bulletin of the Research Institute of Logopedics and Phoniatrics*, vol. 27, pp. 13–36, 1993.

[8] J. Keshet, S. Shalev-Shwartz, Y. Singer, and D. Chazan, "A large margin algorithm for speech-to-phoneme and music-to-score alignment," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2373–2382, 2007.

[9] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi." in *Interspeech*, 2017, pp. 498–502.

[10] J. Keshet, D. Grangier, and S. Bengio, "Discriminative keyword spotting," *Speech Communication*, vol. 51, no. 4, pp. 317–329, 2009.

[11] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[12] D. Palaz, G. Synnaeve, and R. Collobert, "Jointly learning to locate and classify words using convolutional networks." in *INTERSPEECH*, 2016, pp. 2741–2745.

[13] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[14] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[15] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[16] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[18] B. McFee, M. McVicar, S. Balke, V. Lostanlen, C. Thom, C. Raffel, D. Lee, K. Lee, O. Nieto, F. Zalkow, and et al., "librosa/librosa: 0.6.3," Feb 2019. [Online]. Available: https://zenodo.org/record/2564164

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[21] H. Kamper, S. Settle, G. Shakhnarovich, and K. Livescu, "Visually grounded learning of keyword prediction from untranscribed speech," *arXiv preprint arXiv:1703.08136*, 2017.

[22] E. Chodroff, "Corpus phonetics tutorial," *arXiv preprint arXiv:1811.05553*, 2018.

[23] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddingtion, "Results of the 2006 spoken term detection evaluation," in *Proc. sigir*, vol. 7, 2007, pp. 51–57.