Jointly Learning to Locate and Classify Words using Convolutional Networks

Dimitri Palaz[1], *Gabriel Synnaeve*[2], *Ronan Collobert*[1]

[1]Facebook A.I. Research, Menlo Park, CA, USA
[2]Facebook A.I. Research, Paris, France

dimitri.palaz@idiap.ch gab@fb.com ronan@collobert.com

Abstract

In this paper, we propose a novel approach for weakly-supervised word recognition. Most state of the art automatic speech recognition systems are based on frame-level labels obtained through forced alignments or through a sequential loss. Recently, weakly-supervised trained models have been proposed in vision, that can learn which part of the input is relevant for classifying a given pattern [1]. Our system is composed of a convolutional neural network and a temporal score aggregation mechanism. For each sentence, it is trained using as supervision only some of the words (most frequent) that are present in a given sentence, without knowing their order nor quantity. We show that our proposed system is able to jointly classify and localise words. We also evaluate the system on a keyword spotting task, and show that it can yield similar performance to strong supervised HMM/GMM baseline.

Index Terms: convolutional neural networks, attention-based models, keyword spotting, weak supervision, acoustic models

1. Introduction

Recent advances in machine learning ("deep learning") have enabled training systems in an end-to-end manner. This has been proposed in natural language processing [2] or image recognition [3]. In speech recognition, early works have investigated global training of hybrid HMM/ANN (artificial neural networks) systems [4]. More recently, CRF/ANN (conditional random fields) based automatic speech recognition (ASR) systems have been proposed [5, 6, 7]. End-to-end training has also been applied to phoneme recognition [8, 9, 10]. State of the art supervised ASR systems for doing speech transcription use complete sentence transcriptions too. They either use force-alignment [11] (e.g. through an HMM/GMM) to recover the segmentation, or they use a sequence-based discriminative loss as connectionist temporal classification (CTC) [12, 13]. In both cases they train their acoustic model (that goes from sound/features to discrete units) to maximize the classification in phonemes/letters (or words), that they can time-align in the sequence.

There is a growing interest in applying the deep learning approach to weakly-supervised systems. At training time, these pattern recognition systems have only access to the "presence or absence" information of a pattern in a given input, and learn which part of the input is relevant for classifying the pattern. In computer vision, this approach has been successfully applied to image segmentation [1]. Attention-based recurrent models have also been proposed in computer vision [14], machine translation [15] and phoneme recognition [16]. In the speech domain however, it was always assumed that either the segmentation of the training data or at least the sequence information (order of the words) was provided.

We present a novel approach for weakly supervised word recognition. Our system is trained on a sentence basis, with only the speech signal (Mel filterbanks) and the presence or absence of words as a bag-of-word input. It outputs the words that are in the sentence, along with their position and (time-aligned) segmentation. The system is composed of two stages: a sequence modeling stage, based on a convolutional neural network (CNN) [17], which performs the acoustic modeling and outputs a score for each frame, for each word of the vocabulary. The second stage aggregates the score computed by the CNN along the temporal dimension. The output is thus a score for each word, for the whole sentence. During training, the network is able to learn the localisation of words by back-propagating through the aggregation. Such a model can be useful whenever one has access to speech with keyword annotations but not the full transcription (as for hotlines/voice user interfaces). This is also a step towards less supervised automatic speech recognition (ASR) systems that are trained end-to-end.

The remainder of the paper is organized as follows. Section 2 presents the proposed system. Section 3 presents the experimental setup, Section 4 presents the word localisation studies (in which we compare our model with the output of force alignment) and Section 5 presents a keyword spotting evaluation. Section 6 concludes the paper

2. Proposed approach

2.1. Overview

The proposed approach is a weakly-supervised multi-word detection system. It takes a feature sequence X as input, and outputs the probability of each word w in the dictionary $\mathcal{W}$ being present in the utterance. The main novelty of the proposed approach is that the system is trained in a weekly-supervised manner, using bag-of-words labels, and is able to *learn* the words localisation is the utterance.

2.1.1. Bag-of-word labels

In this work, we use *Bag-of-words* (BoW) labels. Based on the bag-of-word model used in natural language processing, these labels denote, for a given utterance, the "presence or absence" information of each word in the dictionary. They are extracted from the transcription, and are represented by a binary vector, of dimension equal to the dictionary size. Note that such labels do not take into account the words order nor quantity.

For example, given the transcription "*John likes to watch movies. Mary enjoys movies too.*", the resulting BoW labels

Dimitri Palaz is with Idiap Research Institute, Martigny, Switzerland and Ecole Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland. This work was done during an internship at FAIR.

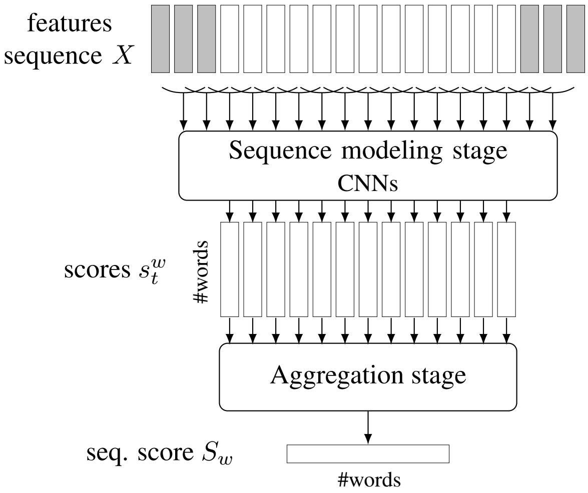

Figure 1: *Illustration of the proposed system. The gray input frames represent the padding.*

are: {“enjoys”,“likes”, “movies”,“to”, “too”,“watch” }, assuming that ”John” and ”Mary” are not in the dictionary. The binary label vector for this utterance can then be built, setting to 1 the entries corresponding to the indices of the words, and -1 all the other entries of the vocabulary.

2.2. Two stages CNN-based system

The proposed system is composed of two stages: the sequence modeling stage processes a sequence of features, and outputs a score for each word, for each frame. The second stage performs the aggregation of the scores along the temporal dimension, and outputs a score for each word, for the whole utterance. Both stages are trained jointly. The proposed architecture is presented in Figure 1.

2.2.1. Sequence modeling stage

The sequence modeling stage performs the acoustic model of a speech utterance. The network is given a sequence of features $X = [x_1 \ \ x_2 \ \ \ldots \ \ x_T]$, where x_t stand for the frame at time t. The output is a score $s_t^w(X)$ for each frame t and each word $w \in \mathcal{W}$. This score is referred to as the *localisation score*.

This stage is implemented by a succession of n convolution layers. A convolutional layer applies the same transformation over each successive (or interspaced by dW frames) windows of kW frames. Formally, the transformation at frame t is written as:

$$C(X) = h(M[x_{t-(kW-1)/2} \ \cdots \ x_{t+(kW-1)/2}]^T) \qquad (1)$$

where M is a $d_{out} \times d_{in}$ matrix of parameters, d_{in} denotes the input dimension, d_{out} denotes the output dimension of each frame and $h(\cdot)$ is the Rectifier Linear Unit [18] non-linearity. The localisation score can thus be expressed as:

$$s_t^w(X) = C_n(C_{n-1}(...C_1(X))) \qquad (2)$$

where C_i denote the i^{th} convolutional layer.

2.2.2. Aggregation stage

For a given sentence X of length T, the sequence modeling stage produces a score $s_t^w(X)$ for each frame t and each word $w \in W$. Given that at training time we have only access to the sequence-level bag-of-word labels, we need a way to aggregate these frame-level scores into a single sequence-level detection score $S_w = aggreg(s_t^w)$.

The aggregation $aggreg(\cdot)$ should drive the network towards correct frame-level assignments. A possible aggregation would be to take the sum over all frames: $S_w = \sum_t s_t^w$. This would however assigns the same weight on all frames of the speech sequence during the training procedure, even to the ones which do not belong to the words corresponding to the labels. On the other hand, one could apply a max aggregation: $S_w = \max_t(s_t^w)$. This would encourage the model to increase the score of the frame which is considered as the most important for the classification of a given word. With this approach, the position of a given word would be correctly predicted, but its duration would not, as only one frame is encouraged. We propose a trade-off solution between these two cases, which is the `LogSumExp` [19] (LSE):

$$S_w^r(X) = \frac{1}{r}\log\left(\frac{1}{T}\sum_t \exp(r s_t^w(X))\right) \qquad (3)$$

where r denotes the hyper-parameter controlling how smooth one wants the approximation to be: high r values implies having an effect similar to the max, very low values will have an effect similar to the score averaging (sum). The advantage of this aggregation is that the frames which have similar scores will have a similar weight in the training procedure.

2.3. Training

In the proposed approach, we assume that only the bag-of-word labels are available at training time. As more than one word can be present in a sequence, the standard cross-entropy cost function is not suited in this case. We propose to treat the task as a separate binary classification problem for each word. The loss function $\mathcal{L}$ is thus a sum of of $|\mathcal{W}|$ binary logistic regression classifiers:

$$\mathcal{L}(S(X), y) = \sum_{w=1}^{|\mathcal{W}|} \log(1 + e^{-y_w S_w(X)}) \qquad (4)$$

with $S_w(x)$ being the score for the word w and the sequence input x and y being the bag-of-word label for sequence X, with $y_w = \{-1, 1\}$ denoting the presence or absence of the word w in the sequence.

Treating a multi-label classification problem as a sum of independent classifier seems to be inadequate, but in our approach, the binary classifiers are not independent as they share hidden layers (in the sequence modeling stage), which can model the inter-label dependencies, if any.

2.4. Inference

During inference, the unseen utterance X is given as input to the system. The system will produce as output the detection score $S_w(X)$ (as defined in (3)) for each word in the dictionary. Using this score, the probability $P(w|X)$ of the word w being present in the utterance can be computed:

$$P(w|X) = \frac{1}{1 + e^{-S_w(X)}} \qquad (5)$$

This probability can be used for word detection tasks, such as keyword spotting.

As presented in the previous sections, the proposed system is designed such as it is able to learn the word localisation. During training, the model increases the localisation score s_t^w, as defined in (2), of the frames which are considered the most important for the word detection. At inference time, we make the assumption that for a given word, the score s_t^w is a measure of the likelihood of the word being in the utterance at time t. Based on that assumption, the most likely position p_w of a given word, i.e. the most probable frame, can be computed as:

$$p_w = \operatorname{argmax}_t(s_t^w) \tag{6}$$

In order to localise a given word, a simple model is proposed: a threshold is applied to the localisation score for the given word. Thus, the word localisation is given by each frame whose scores are higher than the threshold. A threshold per word is used, and is determined experimentally.

$$s_t^w > \theta_w, \quad \forall t \tag{7}$$

with θ_w being the threshold for the word w. Note that it is possible to detect more than one occurrence of a given word in the utterance with this method.

3. Experimental Setup

We use Mel Filterbanks coefficients as input features. They were computed using the `Spectral` package[1]. These features consist of 40 coefficients, computed on a 25 ms window, with a 10 ms shift, without any speed or acceleration coefficients. The hyper-parameters of the network were tuned on the validation set by maximizing the F1 score. In the results, we used a detection probability threshold of 0.4, that yields a F1 score (on words) of 0.72 on the clean development set, and 0.6 on the other development set. The proposed architecture is composed of 10 convolutions layers. The first layer has a kernel width of 5 frames, the 9 other layers have a kernel width of 10 frames. They all have a shift of 1 frame, and 80 filters. The 1000th most common words in the training set were used as targets. We train the network using stochastic gradient descent [20] with a learning rate of 10^{-5}. The experiments were implemented using the *torch7* toolbox [21].

The LibriSpeech corpus [11] is an English corpus derived from read audio books, sampled at 16 kHz, The trainset consists of 280k utterances, representing 960 hours of speech. Two development and test sets are available. In both cases, the first set is composed of high quality utterances and is referred to as *dev_clean* and *test_clean*. The second one is composed of lower quality utterances, and referred to as *dev_other* and *test_other*. Each of these sets consists of 40 speakers, and represents about 5 hours of speech. To obtain the word alignments, we use the `s5` recipe, provided by the Kaldi toolbox [22]. It is a HMM/GMM system, taking MFCC as input; more details can be found in [11]. We extract the word alignment from the phoneme-based forced alignment.

4. Word localisation study

In this section, we evaluate the capability of the proposed approach to *learn* the word localisation in a weekly-supervised manner. To this aim, we propose two experiments: first, we evaluate the system capability to detect if a correct word position is an utterance. Secondly, the duration of words learned by the proposed system is evaluated. For these two studies, we use the frame-level word alignment as ground-truth.

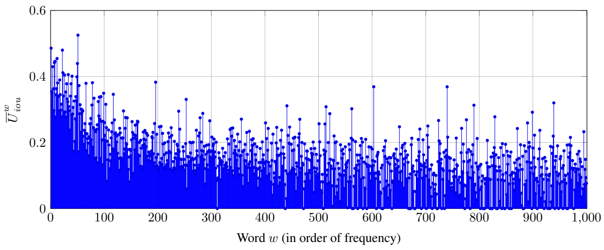

Figure 2: *Mean IoU for each word on the test_clean set.*

4.1. Word position

For each utterance, the most probable position of a given word is computed using Equation (6). We then check if this position is correct (i.e. if the word is present at this frame on the ground-truth labels). We propose two evaluation settings. In the first one, referred to as *oracle*, the word detection capability of the system is assumed to be perfect (i.e. we use the ground-truth frame-level word labels). In the second setup, referred to as *actual*, we perform a word detection by thresholding the probability of the word being present in the sequence using (5), and then compute the position accuracy as presented above. In this case, the threshold was tuned to maximize the F1 score on word classification. The results are presented in Table 1. One can observe that the proposed system is able to correctly detect the position of most of the words.

Table 1: Word position accuracies

Set	Oracle	Actual
test_clean	87.1 %	60.1 %
test_other	83.5%	55.2%

4.2. Word duration

The duration of a given word is inferred by thresholding the localisation score, as presented in Section 2.4. To evaluate the capacity of the proposed system to predict the correct word duration, we use the Intersection-over-Union (IoU) metric. This metric can be seen as a proximity measure between two patterns, as it is equal to 0 if they do not overlap, and equal to 1 if they are perfectly matching. A IoU score of 0.5 indicates that half of the patterns match. It is well used for image segmentation (see [1] for example). Formally, it is defined as:

$$U_{\text{iou}}^{(w)}(\tilde{y}, y) = \frac{\sum_t \mathbb{1}_{\{\tilde{y}_t = w \wedge y_t = w\}}}{\sum_t \mathbb{1}_{\{\tilde{y}_t = w \vee y_t = w\}}} \tag{8}$$

with $\tilde{y}$ denotes the inferred sequence, y denotes the reference, $w \in \mathcal{W}$ denotes a given word and $\mathbb{1}_{\{predicate\}}$ denotes the indicator function, which is 1 if the predicate is true and 0 otherwise.

Figure 2 presents the mean IoU for each word in the dictionary. One can see that in average, about one third of the word duration is captured. Figure 3 presents an illustration of an inferred sequence and the ground-truth. Clearly, the proposed system predicts shorter duration. This aspect could be improved, for example by assigning the unassigned frames with neighbors word labels, and will be part of our future work.

[1] `https://github.com/mwv/spectral`

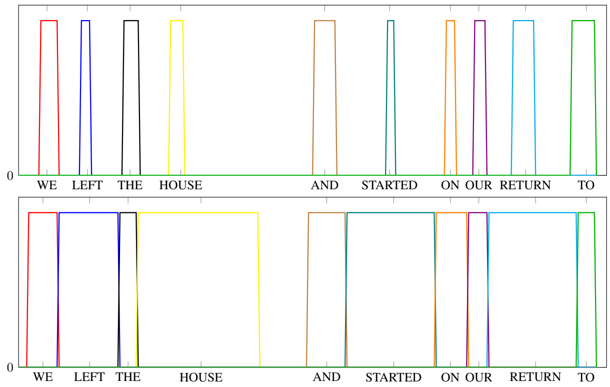

Figure 3: *Illustration of an inferred sequence on the top and its corresponding ground-truth, on the bottom.*

5. Keywords Spotting Study

As presented in the previous section, the proposed approach is able to learn the word localisation. In this section, we evaluate the system in an "real-word" application: keyword spotting. To demonstrate the viability of the system, we propose a preliminary study where the experiments are subjected to the following constraints:

- The keywords spotted are in-vocabulary words, i.e. words seen during training.
- As mentioned in Section 3, the word dictionary is limited to the 1000th most common words in the corpus. Thus, the keywords selected for the study are part of this subset. This is unusual for KWS studies, as the selected keywords are usually quite uncommon. This constraint is selected for practical reasons, mainly for training speed. However, the number of words is a hyper-parameter, and could be extended to any number of words.

Table 2: Keywords list (in vocabulary)

any	battle	birds	cannot
easily	fifty	filled	great
known	land	lie	never
only	perfect	perhaps	presence
show	thank	them	years

For evaluation, we use the Maximum Term Weight Value (MTWV) metric, presented in [23], which is defined as one minus the average loss of the system. A perfect system output gives a MTWV of 1 and a empty output gives a MTWV of 0. We use the F4DE tool [24] for scoring. The set of keywords that we used is presented in Table 2.

5.1. Keyword spotting method

For the keyword detection, a simple model is used: for each utterance, the likelihood of the keyword being present in the utterance is determined by thresholding the probability $P(w|X)$ as defined in Equation (5). The starting and ending time of the keyword is then computed by thresholding the localisation score, as presented in Equation (7).

5.2. Baseline

In order to compare the performance of the proposed system on keyword spotting task, we select as our baseline one the most common KWS system, provided by the Kaldi toolbox[2]. The baseline is trained in a supervised manner, and is based on a HMM/GMM-based LVCSR system. The KWS task is performed using the lattice indexing technique, as presented in [25]. This technique is based on generating, for each lattice computed by the ASR system, a transducer structure in which the start-time, the end-time and posterior probability of each word is stored. For evaluation, we did not use a language model for keyword decoding, as our system does not use one.

5.3. Results

Table 3 presents the results for the keyword spotting study for the proposed system and the baseline, expressed in term of MTWV. On the *test_clean*, the proposed system yields similar results to the baseline. Note that the proposed system is trained only in a weakly-supervised manner and the baseline is trained in a supervised manner. This result clearly shows that the proposed system is able to jointly localise and classify words. On the *test_other* set, the performance gap between the proposed system and the baseline suggests that the proposed system is less robust than the baseline to mis-matched condition.

Table 3: Keyword spotting performance on the *test_clean* and the *test_clean* set of LibriSpeech.

Set	System	MTWV
test_clean	Baseline	0.72
	Proposed	0.69
test_other	Baseline	0.49
	Proposed	0.33

6. Conclusion

We presented a novel approach to jointly localise and classify words from speech, trained in a weakly-supervised manner using bag-of-words labels. The proposed system is based on sequence training, and is composed of a convolutional neural network, which performs the acoustic modeling, and of an aggregation stage, which aggregates the frame-level score into a sequence-level score for words. We showed that our system is able to localise words, and yield comparable performance to a strong baseline trained in a supervised manner for in-vocabulary keyword spotting. For future work, we will investigate out-of-vocabulary keyword spotting, in particular by using pairwise distances in our acoustic vectorial representation of (in-vocabulary) words and their similarity to out-of-vocabulary words, that we can project in this space. We will extend the proposed approach to connected word recognition task, by adding a decoder.

7. Acknowledgments

We thank Nicolas Usunier for discussions and constructive feedback on this article.

[2] http://kaldi.sourceforge.net/kws.html

8. References

[1] P. O. Pinheiro and R. Collobert, "From image-level to pixel-level labeling with convolutional networks," in *Proceedings of CVPR*, 2015, pp. 1713–1721.

[2] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *The Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[3] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012, pp. 1106–1114.

[4] Y. Bengio, R. De Mori, G. Flammia, and R. Kompe, "Global optimization of a neural network-hidden markov model hybrid," in *Proc. of IJCNN*, vol. ii, jul 1991, pp. 789 –794 vol.2.

[5] J. Morris and E. Fosler-Lussier, "Conditional random fields for integrating local discriminative classifiers," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 3, pp. 617–628, Mar. 2008.

[6] A. Mohamed, D. Yu, and L. Deng, "Investigation of full-sequence training of deep belief networks for speech recognition," in *Proceedings of Interspeech*, vol. 10, 2010, pp. 2846–2849.

[7] Y. Kubo, T. Hori, and A. Nakamura, "Integrating deep neural networks into structural classification approach based on weighted finite-state transducers." in *Proceedings of Interspeech*, 2012.

[8] A. Graves, *Sequence transduction with recurrent neural networks*. Springer, 2012, vol. 385.

[9] D. Palaz, R. Collobert, and M. Magimai. -Doss, "End-to-end Phoneme Sequence Recognition using Convolutional Neural Networks," *ArXiv e-prints*, Dec. 2013.

[10] D. Palaz, M. Magimai-Doss, and R. Collobert, "Joint phoneme segmentation inference and classification using crfs," in *Proceedings of GlobalSIP*. IEEE, 2014, pp. 587–591.

[11] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," *Proceedings of ICASSP*, 2015.

[12] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of ICML*. ACM, 2006, pp. 369–376.

[13] D. Amodei, R. Anubhai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, J. Chen, M. Chrzanowski, A. Coates, G. Diamos *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," *arXiv preprint arXiv:1512.02595*, 2015.

[14] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Proceedings of NIPS*, 2014, pp. 2204–2212.

[15] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of ICLR*, 2014.

[16] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proceedings of NIPS*, 2015.

[17] Y. LeCun, "Generalization and network design strategies," in *Connectionism in Perspective*, R. Pfeifer, Z. Schreter, F. Fogelman, and L. Steels, Eds. Zurich, Switzerland: Elsevier, 1989.

[18] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[19] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge University Press, 2004.

[20] L. Bottou, "Stochastic gradient learning in neural networks," in *Proceedings of Neuro-Nmes 91*. Nimes, France: EC2, 1991.

[21] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A matlab-like environment for machine learning," in *BigLearn, NIPS Workshop*, 2011.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *Proceedings of ASRU*. IEEE, Dec. 2011.

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

[24] "F4DE NIST tools," http://www.itl.nist.gov/iad/mig/tools/.

[25] D. Can and M. Saraclar, "Lattice indexing for spoken term detection," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 8, pp. 2338–2347, Nov 2011.