Detecting words in speech using linear separability in a bag-of-events vector space

Maarten Versteegh,[1,2] *Louis ten Bosch*[2]

[1]International Max Planck Research School for Language Sciences, Nijmegen
[2]Centre for Language Studies, Radboud University Nijmegen, The Netherlands

m.versteegh@let.ru.nl, l.tenbosch@let.ru.nl

Abstract

This paper studies the properties of the Histograms of Acoustic Co-occurrences (HAC) approach to acoustic modeling. While HAC-vectors have been predominantly used with matrix decomposition algorithms, we show that the additivity and sparseness constraints inherent in HAC lead to a representational space in which utterances are linearly separable with respect to the words they contain. The method implies that it is possible to detect and locate words in test utterances by using a classifier that is trained without any information about word order or word location during training. We demonstrate this by showing that an ensemble of linear classifiers can reach excellent detection scores on the TIDIGITS dataset. We further explore the usefulness of the linear separability in the HAC space by demonstrating the use of a sliding window decoder for continuous speech recognition.

Index Terms: Speech analysis and representation, new paradigms

1. Introduction

Speech recognition technology has predominantly relied on continuous density Gaussian Mixture Models (GMMs) for acoustic modeling in combination with Hidden Markov Models (HMMs) that represent speech as a sequence of discrete units on several hierarchical levels. Underlying HMMs is the assumption that speech can be represented as a sequence of discrete phone untis. Already at the time of their introduction in the Automatic Speech Recognition (ASR) community, it was clear that this assumption is not in accordance with the physics of the speech production process [1]. It was also clear from the beginning that discrete or continuous density GMMs are not the only, and maybe not the best, way to estimate the likelihood that a given speech frame is associated with a phone [2, 3]. More than a decade ago already the ASR community started to ask the question whether the GMM-HMM approach was reaching a performance ceiling well below human performance, and whether there was a need for developing fundamentally different approaches than the GMM-HMM framework [4]. Therefore, it is not surprising that several research groups have explored the possibility to revive older approaches which made fewer assumptions that may violate our knowledge about the speech production process e.g. [5]. Other groups have investigated alternative approaches to acoustic modeling, for example exemplar-based sparse representations [6] or the classification of acoustic units in a latent feature space, e.g. [7, 8, 9, 10].

Histograms of Acoustic Cooccurrences (HAC) vectors [10] are yet another type of representation of acoustic signals compatible with the concept of a latent feature space. The HAC approach was developed in a context in which it was assumed that complex physical phenomena, such as speech signals or pictures, can be decomposed into a positive sum of a limited number of basic elements. Initially, these elements may be unknown, so that they must be discovered by processing some –perhaps large– amount of complex observations. To discover the basic elements, and te decompose unknown observations into the linear combination of basis elements, the Non-negative Matrix Factorization (NMF) [11] algorithm was used. Similar to other approaches based on a latent feature space, NMF requires that observations are in the form of a fixed length vector, a requirement that seems difficult to fulfil in the case of speech, because utterances may have different durations. The approach of combining HAC-vectors with NMF has shown promising results in speech recognition [12] and in developing computational models of word learning [13].

This paper argues that HAC-vectors have properties that make them interesting devices for speech processing independent of NMF. We show that the construction of HAC-vectors makes it possible to decide whether a continuous speech utterance does –or does not– contain a specific word or expression. We use this property to design a keyword detector based on linear separation in the HAC space. Subsequently, we introduce a sliding window technique that enables us to determine the order in which words appear in an utterance. Finally, we speculate about HAC-representations as an alternative model for human cognitive processing and storage of speech/internal representations of words.

2. Linear separability

This section first explains how HAC-vectors are constructed, and shows that an arbitrary HAC-vector can be decomposed in a positive sum of more elementary HAC-vectors. In subsection 2.2 we explain how the linear additivity property of HAC-vectors can be used to construct a linear multi-class classifier with as many classes as there are words that must be discriminated in some task. Finally, we explain how the multi-class classifier can be used to determine where a specific word is located in an utterance.

2.1. HAC-vectors

As can be inferred from the name *Histogram of Acoustic Cooccurrences*, HAC-vectors comprise counts of individual *acoustic events* and the cooccurrence of these events in some time win-

dow. In our implementation of the HAC-encoding of a speech recording the basic acoustic events are defined as the presence of one of the labels in a code book that is formed through a Vector Quantization of some (spectral) representation of speech signals. In our implementation this spectral representation consists of 13 Mel-frequency cepstral coefficients with Δ and Δ^2 coefficients, computed with a 25ms window and 10ms window shift, resulting in the speech signal being represented as a sequence of 39-dimensional vectors. This vector sequence is first transformed into a sequence of triplets of integer vectors, consisting of three VQ-labels from three code books, one with 150 labels for static MFCCs, one with 150 labels for the Δ's, and one with 100 labels for the Δ^2's. These code books were trained using a random sample, comprising equal amounts of speech from female and male speakers, from the read speech in the Spoken Dutch Corpus [14]. Code books can be constructed from other, perhaps more powerful representations of the speech signal [15], but in this paper we will not pursue such optimizations.

For a given speech utterance, we now have a sequence of triplets of vectors of codeword indices. The idea behind HAC is to sum over the time frames representing the utterances, effectively resulting in a single triplet of codeword indices, which is a histogram of acoustic occurrences. Directly summing over the time frames has the disadvantage that the representation becomes permutation invariant, which impedes the extraction of longer acoustic units, like keywords. To counteract this, instead of single time frames, HAC constructs a histogram of *co-occurrences* of acoustic events, by taking the co-occurrence counts of codeword indices for acoustic frames at a lag of τ ms apart. Multiple values for τ can be chosen to incorporate more temporal information in the resulting HAC-vector, but this has the disadvantage of greatly increasing the dimensionality. In our implementation, the lags are chosen to capture the variation in the signal within the duration of a phone, with two values for the lag τ at 20ms and 50ms. The eventual HAC-vector contains $2 \cdot (150^2 + 150^2 + 100^2) = 110,000$ elements.

2.2. Towards linear separation of words

By construction, HAC-vectors are linearly additive. The coefficients in a HAC-vector are counts of the number of occurrences in some acoustic signal. Except for minor differences at the borders of a speech segment, the counts obtained for a complete utterance must be equal to the sum of the counts obtained for the segments into which the utterance could be divided. Formally, if a sequence of code book labels $\mathbf{S}$ representing an utterance can be split into chunks $\langle s_1, \ldots, s_n \rangle$, then $h(\mathbf{S}) \approx \sum_{i=1}^{n} h(s_i)$, where $h(\cdot)$ denotes the HAC transform. This linearity property makes the HAC-representation suitable for algebraic decomposition.

The HAC-encoding was originally invented as a representation of acoustic signals that is compatible with NMF [10]. NMF is based on the algebraic decomposition of a (typically) large, non-negative $n \times p$ design matrix $\mathbf{V}$, with observations as the columns and features on the rows, in terms of a product of an $n \times r$ matrix $\mathbf{W}$ and an $r \times p$ matrix $\mathbf{H}$, such that both $\mathbf{W}$ and $\mathbf{H}$ are non-negative and $d(\mathbf{V}, \mathbf{WH})$ is minimized for some metric d. The result of the factorization is that the columns of $\mathbf{W}$ are basis elements in $\mathbb{R}^n$ and the columns of $\mathbf{H}$ in $\mathbb{R}^r$ are coefficient vectors representing the data points of $\mathbf{V}$ in the basis denoted by $\mathbf{W}$. Thus, NMF finds a representation of a data set $\mathbf{V}$ in terms of a set of basis vectors $\mathbf{W}$ and a set of linear combinations of these, $\mathbf{H}$.

One of the first applications of the combination of HAC and NMF was the discovery of recurrent acoustic patterns in continuous speech that can serve as the representation of a protolexicon (i.e., an association of objects in the physical environment with stretches of speech) [16]. Acoustic HAC-vectors were extended by a sub-vector that contained zeros and ones to indicate the absence or presence of a word or a semantic feature in the utterance from which the acoustic HAC-vector was constructed. NMF was used to find the set of basis vectors $\mathbf{W}$ that best represent a number of training utterances. Subsequently, the acoustic part of these basis vectors can be used to optimally approximate the acoustic part of HAC-vectors constructed from utterances that come without the sub-vector containing word or semantic labels. That sub-vector can be reconstructed by applying the weight vector $\mathbf{H}$ obtained with the acoustic part of the unknown utterance to the sub-vectors that contain the labels in $\mathbf{W}$ [16, 15].

2.2.1. Exploiting HAC representations

Although it might seem that the HAC representation is no more than a ruse to enable the use of NMF for speech processing tasks, we believe that the HAC representation deserves consideration independent of NMF. HAC representations of speech utterances are very similar to bag-of-words representations of texts in many language technology applications. It is well known that it is very difficult to develop representations based on conventional linguistic principles that outperform bag-of-words representations in most applications [17]. Therefore, it might well be that HAC representations can serve a range of applications in speech processing.

In the same way as representing texts as points in a space spanned by the words they contain we can think of HAC vectors as points in a space spanned by elementary acoustic events and their cooccurrences. We ask the question whether it is possible to find hyperplanes in the high-dimensional HAC space that separate the utterances containing some specific word from all utterances which do not contain that word. This is a two-class classification task, for which a large number of algorithms is available. In this paper we use logistic regression, arguably the best-known and simplest technique for this purpose. If more than one word must be handled, multiple binary classifiers must be constructed. Once the hyperplane that separates the utterances that contain a specific word from all other utterances has been constructed using a database of labeled training utterances, the presence of that word in an unknown test sentence can easily be determined by projecting the utterance in the HAC space, and checking whether the projected point is at the desired side of the word's hyperplane. The distance of the point to the hyperplane can serve as a confidence measure, or as an estimate of the probability that the utterance does indeed contain the word.

2.2.2. A sliding-window decoder

When applied to complete utterances, the binary classifiers introduced above can be used to determine whether an utterance contains a specific word, but not where that word is in the utterance. For the latter purpose we use a sliding window decoder, in which a window that is long enough to create a meaningful HAC-vector is shifted along the utterance. For each position of the window we use the resulting HAC-vector to compute the probability that the window contains each of the words for which hyperplanes have been trained. The resulting time series can be filtered to remove spurious and short false alarms. Note that the procedure outlined here can be used to detect the sequence of words in an unknown utterance without the need to

specify the phonetic make-up of the words. A verbatim transcription of the training data is sufficient.

3. Experiments

This section describes two experiments to test the procedure outlined above. We first construct an ensemble of classifiers and use that to detect keywords in speech utterances. Next, we use this same ensemble to construct a sliding window decoder to furthermore predict the order of occurrence and obtain rough estimates of the times of occurrence.

3.1. Data sets

The experiments in this section are performed on the TIDIGITS data set. The entire data set (both training and test sections) was randomly divided into three parts, one training set (N=10,000) and two test sets (N=3857 and N=3464). The training set is used to train the ensemble of classifiers. The first test set is used for the keyword spotting task in section 3.2 and the second to test the sliding window decoder 3.3.

3.2. Keyword spotting

The first test of the algorithm amounts to a simple keyword spotting task on the TIDIGITS data set. An ensemble of classifiers was trained, with a separate classifier for each digit ('oh' and 'zero' are counted as separate digits). Each of the classifiers is trained using logistic regression and separately optimized for the regularization parameter C using 5-fold stratified cross-validation over the training set.

Table 1 shows the performance of this ensemble on the first test set (N=3857). The table documents the accuracy score and binary area under curve (AUC), computed from the receiver operating characteristic (ROC). The results show that the procedure we outlined above can effectively classify complete utterances based on their composing keywords, as the results are near ceiling. It validates our hypothesis that HAC-vectors can be used to construct effectively separating hyperplanes between utterances, even in the case of multiple labels per utterance.

Digit	AUC	Accuracy	Digit	AUC	Accuracy
1	0.9744	0.9838	7	0.9636	0.9762
2	0.9658	0.9780	8	0.9422	0.9643
3	0.9747	0.9852	9	0.9607	0.9618
4	0.9809	0.9874	'oh'	0.9292	0.9618
5	0.9729	0.9830	'zero'	0.9818	0.9903
6	0.9841	0.9895	**overall**	**0.9667**	**0.9795**

Table 1: Performance on the digits spotting task described in section 3.2. The table shows the AUC and accuracy per keyword as well as the overall binary AUC and accuracy.

3.3. Sliding window decoder

The procedure described in section 2.2 and empirically validated in section 3.2 classifies whole speech utterances based on the keywords they contain. It is able to identify which keywords occur in an utterance, but cannot indicate at which time point in the utterance the keywords occur.

This section describes a simple sliding window decoder, which classifies subsequences of a signal using HAC-vectors. Using this approach, information can be gained about the time at which a keyword appears in an utterance and so about the order of the words in that utterance. A sliding window of length 0.5s is moved over the utterance in steps of 50ms. At each step, the part of the speech utterance under the window is transformed into a HAC-vector and this vector is then compared against the classifier ensemble developed above. The classifiers which show an activation greater than 0 are taken to indicate the presence of a keyword. If the activation level for a keyword rises above 0 and stays there for subsequent windows, the decoder detects a single instance of that keyword. If the activation level drops below zero after an activation, and then rises above zero again, it counts as two subsequent instances. In practice, if the same keyword occurs twice in a row, we find that the system often fails to detect these repeated instances, leading to detection errors. In the evaluation of this decoder we therefore exclude errors in these kinds of repetitions.

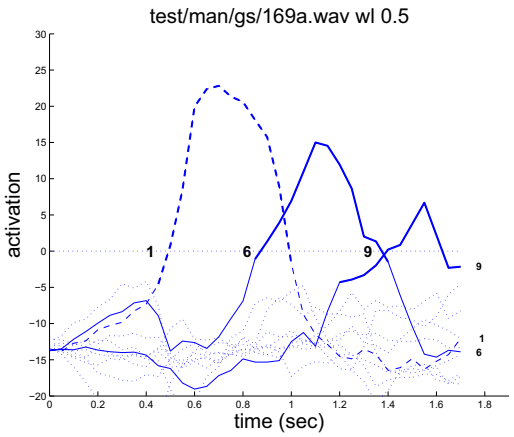

Figure 1: Activation levels for the digits in the utterance '169'. Each of the eleven curves correspond to a digit. The horizontal dashed line corresponding to $y = 0$ marks the theoretical detection threshold. The curves for the keywords which reach activation levels beyond this threshold are plotted in bold style.

Figures 1 and 2 show examples of the activation levels of keywords over the length of utterances from the TIDIGITS data set. The horizontal line at $y = 0$ corresponds to the theoretical detection threshold. Figure 1 shows typical behavior of the decoder in the case of non-repeated keywords. Observe that the activation for the digits which are not present in the utterances never rises above zero. The advantage of the proposed procedure is that it requires no tuning on the threshold, which is simply defined by the hyperplanes constructed by the classifiers. Figure 2 shows the activation levels for an utterance with an immediate repetition of the digit '8'. It demonstrates the inability of the system to deal with repeated keywords appropriately.

Table 2 shows the word confusion matrix of the sliding window decoder. In all, the procedure has a word error rate of 9.66% (5.40% insertions, 3.93% deletions and 0.33% substitutions). The word errors are mainly caused by insertions and deletions. This is in line with the findings in [10] and is not surprising given the simple decoding procedure we employ here.

	‘zero’	‘oh’	1	2	3	4	5	6	7	8	9	Del
‘zero’	903	4	0	0	0	0	0	0	0	0	0	14
‘oh’	0	790	0	0	0	1	0	0	2	0	5	77
1	1	0	912	0	0	0	1	0	0	0	2	38
2	0	2	0	917	0	0	1	0	1	1	0	50
3	0	0	0	0	964	0	0	0	0	2	0	35
4	0	0	0	0	0	888	0	0	0	0	0	27
5	0	1	0	0	0	1	946	0	0	0	1	21
6	1	0	0	0	0	0	0	928	1	0	0	20
7	0	1	0	1	0	0	0	0	965	0	0	29
8	0	0	0	0	0	0	0	0	0	901	1	53
9	0	3	0	0	0	0	1	0	0	0	911	48
Ins	8	284	37	19	15	26	20	25	38	62	31	

Table 2: Performance of the sliding window decoder described in section 3.3. The table shows the word confusion matrix of the digitis in the TIDIGITS test set and the number of insertion and deletion errors for each. Overall, the sliding window decoder has a word error rate of 9.66%, made up of 5.40% insertions, 3.93% deletions and 0.33% substitutions.

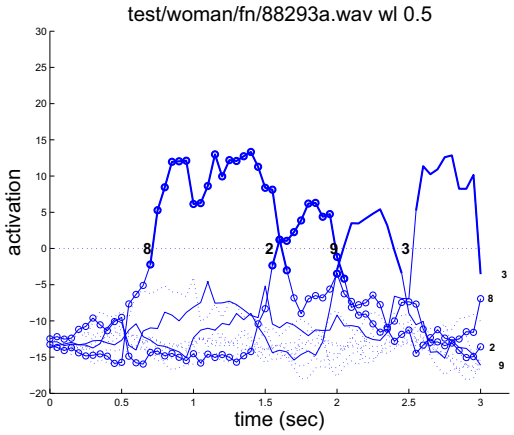

Figure 2: Activation levels for the digits in the utterance ‘88293’. Notice the drop in activation for ‘8’ as the first instance slides out of focus and the second slides in. In this case the activation did not drop below 0, and so only one instance of ‘8’ is detected.

4. Conclusions

This paper proposed that the HAC approach to speech recognition has the potential to find useful applications outside of the combination with NMF. We have shown that the linearity and additivity properties of HAC-vectors lead to a straightforward procedure using logistic regression classifiers for keyword detection and a sliding window decoder based on the same classifiers for continuous speech recognition.

This suggest that HAC-vectors are powerful representations that promise to have interesting applications outside of the simple examples proposed here. The HAC approach (and other latent feature approaches) are interesting for speech recognition and worth pursuing. They also provide a direct link to human cognition.

The literature on the mental lexicon and on the cognitive processes in spoken word recognition is surprisingly silent when it comes to theories (or speculations) about the way in which the acoustic properties of words, syllables and sounds is physically represented in the human brain. It is known that the tonotopical representation of audio signals on the basilar membrane is preserved at least up to the cochlear nucleus [18]. This makes it reasonable to assume that audio signals can be represented in higher levels of the brain as basic acoustic events and their cooccurrences in a short time window, not unlike what we see in HAC representations. Representations of linguistically relevant discrete units can then be modeled as weighted connections between cooccurrences of basic tonotopical patterns, reminiscent of the weight matrices $\mathbf{H}$ in NMF and of the coefficients that specify the hyperplanes in the logistic regression approach used in this paper.

Future work in this area will include the application of these procedure to larger vocabulary datasets. This will explore whether the approaches proposed here scale well enough to be feasible in real-life examples. It will also be interesting to investigate how much of the separability property we describe here is due to the high dimensionality of the HAC-vectors. With over 100,000 features and a great degree of sparseness, there is a lot of opportunity for a regression algorithm to find a suitable hyperplane. It might be interesting to explore the same procedure under lower-dimensional embeddings of the same space. Lastly, in the construction of the HAC-vectors used here and in most other work using the same acoustic representation, the vector quantization is based on a k-means clustering of the acoustic feature space. It might be interesting to investigate softer quantization approaches, for example by clustering the space using Gaussian mixture models.

5. Acknowledgements

The work of MV is supported by grant number 360-70-350 from the Dutch Science Organisation NWO. The authors wish to thank Lou Boves for helpful comments.

6. References

[1] M. Ostendorf, "Moving beyond the 'beads-on-a-string' model of speech," in *Proceedings of the IEEE ASRU Workshop*, 1999, pp. 79–84.

[2] H. A. Bourlard and N. Morgan, *Connectionist Speech Recognition: A Hybrid Approach*. Norwell, MA, USA: Kluwer Academic Publishers, 1993.

[3] L. Golipour and D. O'Shaughnessy, "Contex-independent phoneme recognition using a K-nearest Neighbour Classification approach," in *Proceedings of ICASSP*, 2009, pp. 1341–1344.

[4] H. Bourlard, H. Hermansky, and N. Morgan, "Towards increasing speech recognition error rates," *Speech Communication*, vol. 18, no. 3, pp. 205–231, 1996.

[5] M. Dewachter, K. Matton, K. Demuynck, P. Wambacq, R. Cools, and D. Van Compernolle, "Template-based continuous speech recognition," *IEEE Transactions on Acoustics, Speech and Language Processing*, vol. 15, no. 4, pp. 1377–1390, 2007.

[6] J. Gemmeke, T. Virtanen, and A. Hurmalainen, "Exemplar-based sparse representations for noise robust automatic speech recogntion," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, no. 7, pp. 2067–2080, 2011.

[7] A.-R. Mohamed" and G. Hinton", "Phone recognition using restricted Boltzmann machines," in *Proceedings of ICASSP*, 2010, pp. 4354–4357.

[8] S. Sundaram and J. Bellegarda, "Latent perceptual mapping with data-driven variable-length acoustic units for template-based speech recognition," in *Proceedings of ICASSP*, 2012, pp. 4125–4128.

[9] ——, "Latent perceptual mapping: A new acoustic modeling framework for speech recognition," in *Proceedings of Interspeech*, 2010, pp. 881–884.

[10] H. Van hamme, "HAC-models: A novel approach to continuous speech recognition," in *Proceedings of Interspeech*, 2008, pp. 2554–2557.

[11] D. Lee and H. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788–791, 1999.

[12] V. Stouten, K. Demuynck, and H. Van hamme, "Discovering phone patterns in spoken utterances by non-negative matrix factorization," *IEEE Signal Processing Letters*, vol. 15, pp. 131–134, 2008.

[13] M. Versteegh, L. ten Bosch, and L. Boves, "Modelling novelty preference in word learning," in *Proceedings of Interspeech*, 2011.

[14] N. Oostdijk, "The spoken dutch corpus. overview and first evaluation," in *Proceedings of the Second International Conference on Language Resources and Evaluation*, 2000, pp. 887–893.

[15] J. Driesen, "Discovering words in speech using matrix factorization," Ph.D. dissertation, Arenberg School of Science, Engineering & Technology, KU Leuven, Belgium, 2012.

[16] L. ten Bosch, H. Van hamme, L. Boves, and R. Moore, "A computational model of language acquisition: the emergence of words," *Fundamenta Informaticae*, vol. 90, no. 3, pp. 229–249, 2009.

[17] S. Verberne, L. Boves, N. Oostdijk, and P. Coppen, "What is not in the bag of words for why-qa?" *Computational Linguistics*, vol. 36, no. 2, pp. 229–245, 2010.

[18] H. Spoendlin and A. Schrott, "Analysis of the human auditory nerve," *Hearing Research*, vol. 43, no. 1, pp. 25 – 38, 1989.