Automatic segmentation and clustering of speech using sparse coding

Wiehan Agenbag*, Willie Smit† and Thomas Niesler‡
*‡ Department of Electrical and Electronic Engineering
† Department of Mechanical and Mechatronic Engineering
Stellenbosch University, South Africa
Email: {15988732*, wjsmit†, trn‡}@sun.ac.za

***Abstract*—We investigate the application of sparse coding and dictionary learning to the discovery of sub-word units in speech. The ultimate goal is to generate pronunciation dictionaries that could be used for automatic speech recognition (ASR). A dictionary of sparse coding atoms is trained to code a subset of the TIMIT corpus. Some of the trained units exhibit strong correlation with specific reference phonemes. It is found that our sparse coding model does not place sufficient constraints for the activation of atoms to be temporally isolated, which rules out its direct application to speech segmentation. We also investigate the consistency with which orthographically identical utterances are coded. We find that the sparse coding model used in this study generates codes that contain too much variation for it to be useful for generating pronunciation dictionaries for ASR.**

I. Introduction

We investigate the application of sparse coding and dictionary learning to the task of automatic segmentation and clustering of speech into a compact set of acoustically relevant sub-word units. The ultimate purpose is the automatic generation of pronunciation dictionaries suitable for automatic speech recognition (ASR).

Currently, subword units used for ASR are based on phonemes and must be defined and identified by highly trained linguists. Since this procedure is expensive and requires the availability of linguistic experts in the target language, it presents a major obstacle to the implementation of ASR for severely under-resourced languages.

Previous attempts at automatic discovery of sub-word units have relied upon a linear two-stage approach, where some corpus of speech is first segmented into relatively stationary sub-units. The resulting acoustic fragments are presented to a clustering algorithm, the output of which is a hypothesized set of phonemic units [1].

In this paper, we investigate a novel approach to automatic speech segmentation and clustering by attempting to accomplish the segmentation and clustering in tandem. We hypothesize that such an approach may yield an improvement over blind segmentation and subsequent clustering, since a segmentation informed by matching a speech signal to a relatively small set of clusters that develop in the same step, should be more robust.

II. Background

A. Sparse coding and dictionary learning

Sparse coding can be stated as attempting to reconstruct some input signal using a linear combination of the smallest possible number of basis functions taken from a finite set. That is, a sparse code $\mathbf{x}$, can be seen as a solution to

$$\arg\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{such that} \quad \mathbf{y} = \mathbf{D}\mathbf{x} \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^{N\times 1}$ is the signal we are trying to reconstruct, $\mathbf{D} \in \mathbb{R}^{N\times M}$ is the set of basis functions, packed column-wise, and $\mathbf{x} \in \mathbb{R}^{M\times 1}$ where $\|\mathbf{x}\|_0$ represents the number of nonzero values in the vector $\mathbf{x}$. In the context of sparse coding, it is understood that $M \gg N$, which makes the set of basis functions overcomplete. In the context of speech, we may consider a typical utterance to be our input signal, which we wish to code using a highly sparse selection of sub-word phonemic units, which act as basis functions.

In some cases, the dictionary of basis functions (also called *atoms* or *features*) is not known beforehand. Thus, it is necessary to obtain both the code and the dictionary simultaneously, in steps usually referred to as **sparse coding** and **dictionary learning**. Of course, if we are given only one signal, Equation (1) can be trivially optimized by creating a minimal dictionary containing just that signal. In the case of sparse coding and dictionary learning, it is therefore understood that we are given an ensemble of signals

$$\mathbf{Y} = [\mathbf{y}_0, \mathbf{y}_1, \ldots, \mathbf{y}_{K-1}], \tag{2}$$

for which we need to produce a corresponding ensemble of codes $\mathbf{X}$ and dictionary $\mathbf{D}$ satisfying

$$\arg\min_{\mathbf{X}} \|\mathbf{X}\|_0 \quad \text{such that} \quad \mathbf{Y} = \mathbf{D}\mathbf{X}. \tag{3}$$

It is also understood that $K \gg M$, i.e. the number of signals that we need to code greatly outnumber the number of allowable basis functions.

B. Shift and scale invariance

The formulation of sparse coding discussed in the previous section is not particularly appropriate for the purpose of describing speech signals. In particular, there are two main deficiencies. The first of these is that phonemic sub-units generally occur at any point in a signal, which necessitates a highly redundant dictionary to accommodate all possible time shifts of that unit. The second deficiency is that the units can be compressed or stretched in time and still retain their meaning, leading to an even more redundant set of atoms.

In order to remedy the first deficiency, several authors [2]–[5] have proposed replacing the dictionary-code product $\mathbf{Dx}$, with a dictionary-code convolution, which we define as

$$\mathbf{\Phi} * \mathbf{S} = \sum_{j=1}^{M} \boldsymbol{\phi}^j * \mathbf{s}^j, \tag{4}$$

where $\mathbf{\Phi} \in \mathbb{R}^{N_\phi \times M}$ is the convolutional dictionary, and $\mathbf{S} \in \mathbb{R}^{M \times N}$ are the coefficient sequences. The quantities $\boldsymbol{\phi}^j$ and $\mathbf{s}^j$ refer to the j^{th} column and row of $\mathbf{\Phi}$ and $\mathbf{S}$ respectively. Each atom $\boldsymbol{\phi}^j$ is now associated with a coefficient sequence $\mathbf{s}^j$ that represents not just whether an atom is being used, but also at what position in $\mathbf{y}$.

To address the second deficiency, scale invariance can be afforded to the convolutional sparse coding formulation by including each base atom at several time scales.

C. Optimisation problem

The exact formulation of the sparse coding problem as given in Equations (1) and (3) has so far remained intractable. Moreover, the pursuit of an exact recovery of the input signal may not be useful. For example, there may be some inherent source of unwanted variability in the data (such as additive noise). Also, an exact reconstruction could well be achieved at the expense of greatly reduced sparsity. Most authors therefore choose to cast the problem into an optimization framework [4]–[8]. The cost function which is commonly used is

$$C(\mathbf{\Phi}, \{\mathbf{S}_k\}) = \sum_{k=1}^{K} \|\mathbf{y}_k - \mathbf{\Phi} * \mathbf{S}_k\|_2^2 + \beta\tau(\mathbf{S}_k), \tag{5}$$

where $\mathbf{S}_k$ refers to the coefficient sequences used to code the k^{th} input signal $\mathbf{y}_k$. The cost function can be seen as a weighted sum of the reconstruction error and a code diversity measure $\tau(\mathbf{S})$. The latter term yields small values when the code is sparse, and large values when it is not. The l_0 pseudo-norm used in Equation (1) is one possible diversity measure, but others that are differentiable have been proposed [7], [9].

D. Applicability to speech signals

If one examines the phonetic transcriptions of speech corpora such as TIMIT, it becomes apparent that they represent a high-level shift and scale invariant sparse coding of speech. It therefore seems plausible that a phoneme-like transcription could arise naturally from a convolutional sparse coding pursuit on the acoustic data.

E. Previous work

1) Sub-word unit discovery: Singh et al [10] attempt to obtain a set of sub-word acoustic models and associated transcriptions using a maximum likelihood approach, where they attempt to maximise the likelihood of the acoustic data, conditioned on the orthographic transcriptions, acoustic models and resulting pronunciations. The model used by [10] makes extensive use of the assumption that there exists a strong correlation between the spelling of words and their pronunciation.

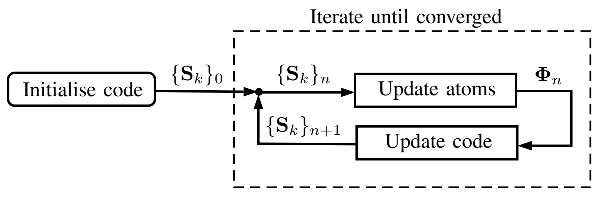


Fig. 1. Overview of sparse code and dictionary learning with an initial code.

Previous work by Goussard and Niesler [1] used the two-stage approach described earlier, where the audio was first segmented into phoneme-like fragments and then clustered. The segmentation was performed by inserting segmentation boundaries whenever the log energy weighted cosine distance between successive MFCC vectors exceeded a predetermined threshold [11]. The resulting set of acoustic fragments was clustered into a compact set of units using agglomerative hierarchical clustering with a dynamic time warping (DTW) alignment cost acting as distance metric.

2) Sparse coding applied to speech: There has also been some research on using sparse coding with the aim of improving automatic speech recognition. In [3], Smit and Barnard developed a continuous speech recogniser using statistical models of sparse code sequences conditioned on individual words. Smit also investigated the impact of changing the parameters of the sparse coding and dictionary learning pursuit, and demonstrated the emergence of a dictionary containing phoneme-like units [7].

III. Implementation

The development of a set of sparse codes and an associated dictionary of atoms proceeds in an iterative fashion (shown in Fig. 1). Starting with some initial set of codes $\{\mathbf{S}_k\}_0$, we calculate the optimal set of atoms $\mathbf{\Phi}_0$ corresponding to those codes. These atoms are then used to calculate a new set of codes $\{\mathbf{S}_k\}_1$, which are then used to recalculate a new set of atoms $\mathbf{\Phi}_1$. This is repeated until convergence of $C(\mathbf{\Phi}_n, \{\mathbf{S}_k\}_n)$.

A. Initialisation

There are many possible strategies for obtaining an initial code for each utterance. In this study, we make the somewhat arbitrary choice of filling each $\mathbf{S}_k$ matrix with ones according to the following rules:

1) Each time frame in the utterance is given a chance to contain a single non-zero coefficient according to a prespecified probability P_θ, and
2) for those time instants that may contain a non-zero coefficient, the corresponding feature index is randomly chosen from a uniform distribution.

The role of P_θ is to encourage an initial l_0 code sparsity. It is chosen to lead to an average "phoneme" rate roughly similar to that of a real phoneme transcription.

B. Obtaining sparse codes

In order to generate sparse codes, we make use of the Coordinate Descent algorithm [8], which uses the l_1 norm

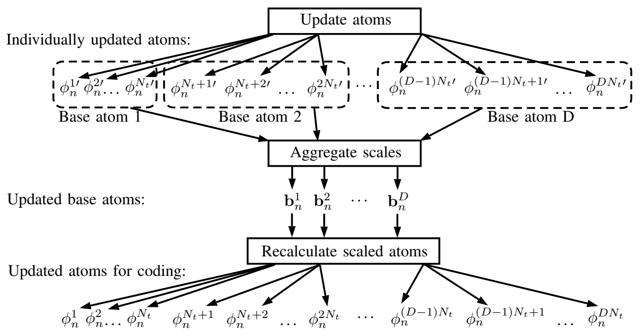

Update atoms
Individually updated atoms:
$\phi_n^{1\prime}\ \phi_n^{2\prime} \dots \phi_n^{N_t\prime}$
$\phi_n^{N_t+1\prime}\ \phi_n^{N_t+2\prime} \dots \phi_n^{2N_t\prime}$
$\phi_n^{(D-1)N_t\prime}\ \phi_n^{(D-1)N_t+1\prime} \dots \phi_n^{DN_t\prime}$
Base atom 1
Base atom 2
Base atom D
Aggregate scales
Updated base atoms:
$\mathbf{b}_n^1\ \mathbf{b}_n^2 \cdots \mathbf{b}_n^D$
Recalculate scaled atoms
Updated atoms for coding:
$\phi_n^1\ \phi_n^2 \dots \phi_n^{N_t}\ \phi_n^{N_t+1}\ \phi_n^{N_t+2} \dots \phi_n^{2N_t} \cdots \phi_n^{(D-1)N_t}\ \phi_n^{(D-1)N_t+1} \dots \phi_n^{DN_t}$

Fig. 2. Overview of the dictionary update process.

as a diversity function. Coordinate Descent greedily optimises the cost function by iteratively choosing the code coordinate (i.e. atom and point in time) whose optimal value would yield the largest reduction in cost. The algorithm terminates once the relative reduction achieved by the next optimal choice of coordinate falls below some threshold.

C. Atom updates

During the atom update stage, we would like to use the updated codes to produce a new dictionary that further reduces the cost function. However, the only term in $C(\mathbf{\Phi}, \{\mathbf{S}_k\})$ that we can optimise is the reconstruction error

$$R = \sum_{k=1}^{K} \|\mathbf{y}_k - \mathbf{\Phi} * \mathbf{S}_k\|_2^2 = \sum_{k=1}^{K} \|\mathbf{r}_k\|_2^2. \tag{6}$$

The atoms are updated using a frequency-domain approach inspired by Grosse et al [4], which recognises that R remains invariant (up to a scaling constant), when the DFT is applied to $\mathbf{r}_k$. This makes it possible to write the atom-code convolutions as elementwise multiplications, yielding a tractable way to solve for all ϕ^j simultaneously in a way that minimises the reconstruction error.

In this study, each of the N_t possible time-scales at which an atom can occur becomes a distinct atom for the purposes of coding, and the atom update approach alluded to above also neglects the relationship between scaled versions of the same atom. Thus, after the scaled atoms have been updated individually, yielding the preliminary updated atoms $\phi_n^{j\prime}$, it becomes necessary to reassert the relationship between those that belong to a particular class. This is done without much rigour by scaling the atoms to a common length (using spline interpolation), and then taking a weighted average according to the relative frequency of the scales. These prototypical *base atoms* $\{\mathbf{b}^d\}$ are then rescaled to all relevant lengths and made available for coding, as shown in Fig. 2.

IV. Experiments and results

A. Data selection and preprocessing

The experiments performed in this study use subsets of the TIMIT speech corpus. The 6300 utterances are divided into three categories of sentences, which are designed to expose different aspects of speech. These categories are:

SA **Dialect sentences**, which are meant to expose dialectical variation among 8 regions across the United States. The category consists of 2 sentences spoken by all 630 speakers.

SX **Phonetically compact** sentences designed to provide good coverage of phone pairs, with some emphasis on interesting or difficult contexts. The corpus contains 450 such sentences and each sentence is spoken by 7 speakers.

SI **Phonetically diverse** sentences are chosen to include diversity in sentence types and phonetic contexts and to maximise the variety of allophonic contexts found in a large corpus of text. The corpus contains 1890 SI sentences and each sentence is spoken only once.

In this study, we elect to use only the phonetically diverse (SI) sentences for training. The other categories contain so much repetition that their inclusion might bias the development of acoustic units that favour very specific contexts.

Before we apply sparse coding and dictionary learning to the selected training set, the utterances are converted to 12-coefficient MFCC feature vectors using the HMM Toolkit (HTK). The MFCCs are generated at a rate of one every 20 ms, with a window size of 32 ms, resulting in a 12 ms overlap between frames. For each speaker, the utterances are pooled for the application of a cepstral mean and variance normalisation.

B. Training overview

Using the selected data, we performed a few experiments to identify the impact of varying the parameters available for tuning (see Table I). For consistency, the same random initialisation was used for each set of parameters, and a fixed number of 20 training iterations was carried out. Each run took approximately 2 hours to complete using 64 AMD Opteron processor cores clocked at 2.1 GHz.

For the parameters used in this study, the training was well-behaved and showed fast convergence. Fig. 3 shows how the sparse coding and feature learning iteratively improves the cost-function until it converges to some local optimum.

Fig. 4 and 5 show the effect of local variation of the diversity penalty and number of base atoms. Unsurprisingly, reducing the diversity penalty leads to codes that are less sparse and therefore a more accurate reconstruction. In most cases, increasing the number of base atoms yielded an improvement in the reconstruction error, with the exception of ($\beta = 8, D = 70$). This may be symptomatic of numerical problems with the exact solver used for the feature updates, since much more severe instability was encountered in the cost function for larger values of N_ϕ and D. Curiously, increasing the number of base atoms also yielded an increase in the number of non-zero code coefficients that was being used by the converged solution.

Fig. 6 shows two of the atom dictionaries learned by our sparse coding and dictionary learning system. It is apparent that the atoms encode quite a bit of context to the left and right

TABLE I. Training parameters

Parameter	Description
β	Diversity penalty
N_ϕ	Maximum atom scale (number of frames)
D	Number of base atoms

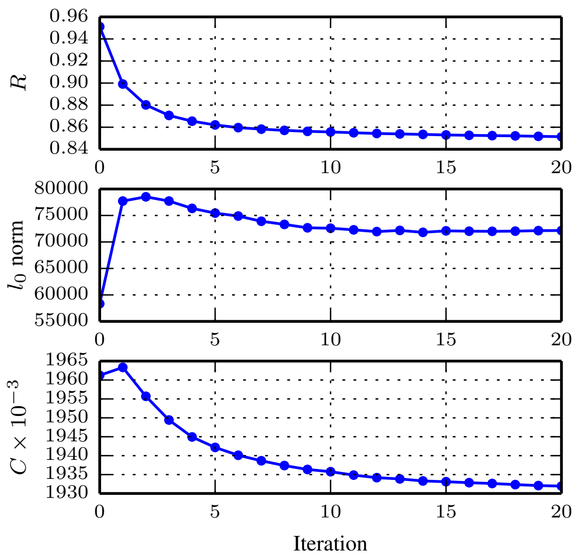


Fig. 3. Development of reconstruction error, code sparsity and cost function for $\beta = 9$, $N_\phi = 20$ and $D = 50$.

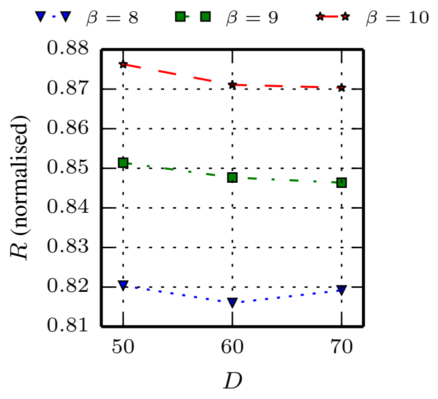


Fig. 4. Terminal normalised reconstruction error vs. number of base atoms for various values of β. N_ϕ is set to 20.

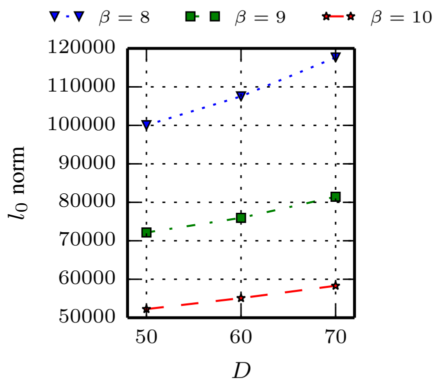


Fig. 5. Number of coefficients used vs. number of base atoms for various values of β. N_ϕ is set to 20. For comparison, the dataset contains 56377 reference phoneme instances.

of the recognisable steady state areas. This seems to suggest that our system is more suitable to the discovery of context-dependent diphones or triphones.

C. Coincidence with reference phonemes

In this section, we examine the relationship of our trained dictionary with the reference set of phonemes and transcriptions provided by TIMIT. Since we did not attempt to infer an optimal alignment between our sparse codes and the reference phoneme transcriptions, we simply count the number of times a non-zero coefficient corresponding to any given base atom occurs within a reference phonemic boundary. Those counts are then recorded in a 2D histogram which we call the *coincidence matrix*. In order to make it more graphically interpretable, the histogram is normalised so that the bins corresponding to each reference phoneme sums to one.

A disadvantage of this approach is that the coincidence matrix inevitably gets muddied by phonemic context—we are not just seeing which atoms are most frequently used to code certain phonemes, but also which atoms are often used just before or after a certain phoneme.

Fig. 7 shows one such coincidence matrix. There are noticeable *hotspots* where one phoneme is strongly coded by a small number of atoms. There are also many phonemes that are broadly coded using many different atoms. In each case no particular atoms have developed that cater well for all occurrences of the phoneme. Overall, it is reasonable to conclude that at least some of the atoms learned are phonetically relevant.

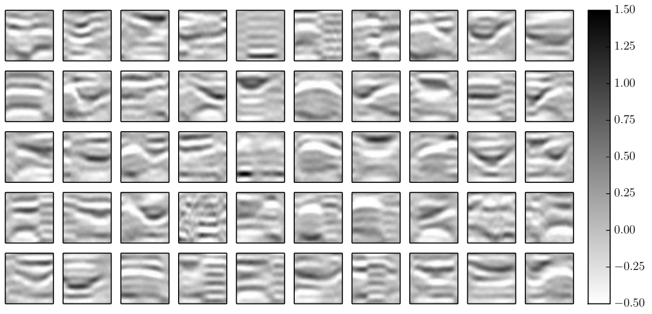
D. Usefulness to the generation of pronunciation dictionaries

In order to use the resultant sparse codes and corresponding atom dictionaries to generate pronunciation dictionaries using the atoms as sub-word units, it is necessary that the sparse codes represent a reasonable segmentation of the acoustic data. At the very least, the sub-word units should not overlap in time.

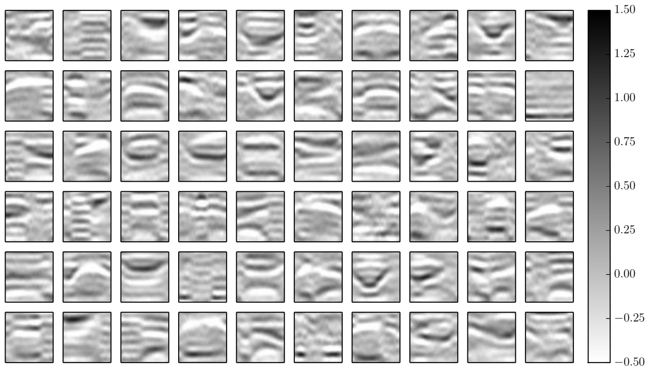
We use a straightforward method to measure overlap: for each unique pair of non-zero code coefficients, we count the number of frames that overlap and add that to a running total. In order to compare the level of overlap between utterances of different lengths, the overlap score is normalised by the length of the utterance.

Fig. 8 shows the average overlap scores for each set of parameters used in this study. In all cases, the overlap scores are much larger than unity, implying that the equivalent of each frame in the utterances is being coded (on average) by several atoms. Clearly, the sparsity constraints we impose are not sufficient for a coding to arise that is localised enough in time to be useful for segmentation. It may be necessary to include a penalty term in the cost function to achieve a reduction in overlap.

The second requirement we place on our sparse codes and atoms for them to be fit for the generation of pronunciation



(a) $\beta = 10$, $N_\phi = 20$ and $D = 50$



(b) $\beta = 9$, $N_\phi = 20$ and $D = 60$

Fig. 6. Mel spectrogram representation of base atom dictionaries at the end of training. Lower frequencies are at the bottom.

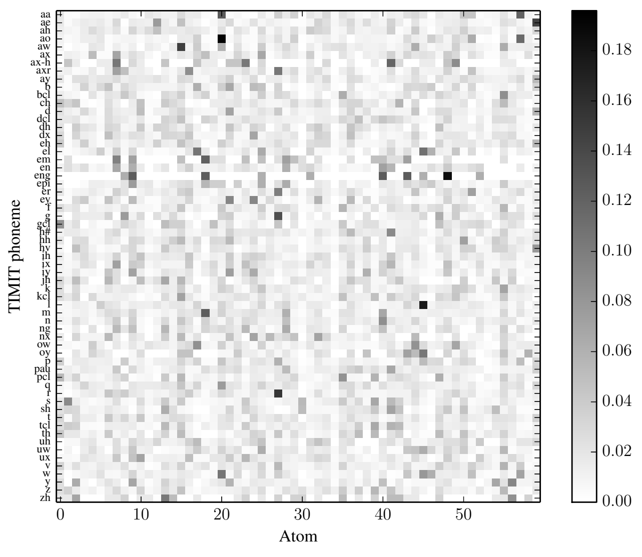


Fig. 7. Coincidence matrix for $\beta = 10$, $N_\phi = 20$ and $D = 60$.

dictionaries, is that utterances that sound the same should be transcribed in the same way using our atoms. If that is not possible, we could end up with an impractically large set of competing pronunciations. Of course, not even the reference phonetic transcriptions for TIMIT are perfectly consistent for utterances with identical orthography, and multiple pronunciations for words often exist. Hence, it is important to establish a baseline consistency score.

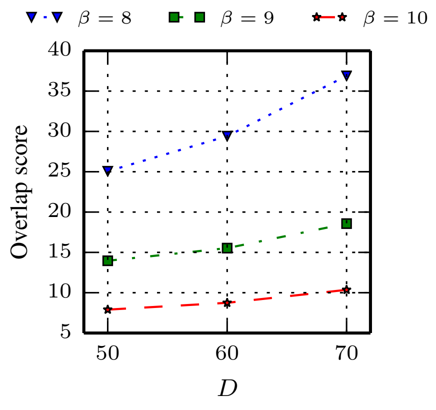


Fig. 8. Normalised overlap scores for various training parameters.

TABLE II. BASELINE CONSISTENCY SCORES USING REFERENCE PHONEME TRANSCRIPTIONS

	SA1	SA2
Overall	0.221	0.232
New England	0.240	0.230
Northern	0.204	0.208
North Midland	0.204	0.222
South Midland	0.216	0.242
Southern	0.231	0.249
New York City	0.209	0.229
Western	0.218	0.237
Army Brat	0.205	0.207

A consistency score between two sparse codes can be obtained by calculating the Levenshtein (edit) distance between the corresponding sequences of base atoms, after they have been sorted according to their midpoints. We normalise the obtained edit distance by the length of the longer sequence, so that we have a score between 0 (when the sequences are exactly the same) and 1 (when there is no commonality at all). When we have multiple sequences to compare, we obtain a *within-class* consistency score by calculating the mean edit distance across all pairs of sequences corresponding to the same orthographic transcription.

We perform our consistency experiment on utterances from the SA dataset. This set contains only two orthographically unique sentences, and is spoken by all speakers in the TIMIT corpus. Table II contains the overall consistency scores of the reference transcriptions for these two utterances, as well as a breakdown per dialectical region. We then calculate sparse codes for the SA utterances using the base atoms learned in our previous experiments, and score them for consistency. Fig. 9 plots these scores, which range from 0.778 to 0.816. These figures indicate a consistency that is far too poor to be used to generate a compact pronunciation dictionary. Clearly there is a big margin for improvement.

The excessive level of overlap seen across all experiments may be partly to blame for this. We hypothesise that allowing the system the freedom to develop codes with highly overlapping atom use may preclude the development of atoms that are sufficiently able to represent temporally isolated acoustic events, which may in turn lead to a poorer consistency score.

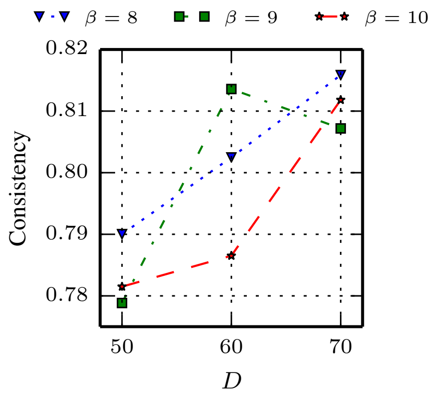


Fig. 9. Average consistency scores on the SA dataset with atoms learned in previous experiments.

V. Conclusion

We found that it is possible to use sparse coding and dictionary learning to discover phonetically relevant units in acoustic data. However, the usefulness of these units for generating pronunciation dictionaries is limited by the development of a segmentation that is not sufficiently localised in time and a transcription that is too inconsistent for utterances that are orthographically identical. In order to improve this, our ongoing work is considering ways to encourage codes that use atoms in a non-overlapping fashion.

VI. Acknowledgements

The presented study was sponsored in part by the National Research Foundation of the Republic of South Africa and by Telkom South Africa. Computations were performed using the University of Stellenbosch's Rhasatsha HPC.

References

[1] G. Goussard and T. Niesler, "Automatic discovery of subword units and pronunciations for automatic speech recognition using TIMIT," in *Proceedings of PRASA*, 2010.

[2] B. A. Olshausen, "Sparse codes and spikes," in *Probabilistic models of the brain: Perception and neural function*. MIT Press, 2001, pp. 257–272.

[3] W. Smit and E. Barnard, "Continuous speech recognition with sparse coding," *Computer Speech & Language*, no. 2, p. 200219.

[4] R. B. Grosse, R. Raina, H. Kwong, and A. Y. Ng, "Shift-invariance sparse coding for audio classification," *CoRR*, vol. abs/1206.5241, 2012.

[5] M. Mørup, M. N. Schmidt, and L. K. Hansen, "Shift invariant sparse coding of image and music data," Tech. Rep., 2008. [Online]. Available: http://www2.imm.dtu.dk/pubdb/p.php?4659

[6] K. Kavukcuoglu, P. Sermanet, Y. lan Boureau, K. Gregor, M. Mathieu, and Y. Lecun, "Learning convolutional feature hierarchies for visual recognition."

[7] W. Smit, "Sparse coding of single spoken digits," in *PRASA 2013*, 2013.

[8] Y. Li and S. Osher, "Coordinate descent optimization for l_1 minimization with application to compressed sensing; a greedy algorithm," *Inverse Probl. Imaging*, vol. 3, no. 3, pp. 487–503, 2009.

[9] B. A. Olshausen and D. J. Fieldt, "Sparse coding with an overcomplete basis set: a strategy employed by v1," *Vision Research*, vol. 37, pp. 3311–3325, 1997.

[10] R. Singh, B. Raj, and R. Stern, "Automatic generation of subword units for speech recognition systems," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 2, pp. 89–99, Feb 2002.

[11] L. ten Bosch and B. Cranen, "A computational model for unsupervised word discovery." in *Proceedings of Interspeech*, 2007, pp. 1481–1484.