

# USING SPARSE REPRESENTATIONS FOR EXEMPLAR BASED CONTINUOUS DIGIT RECOGNITION

*J. Gemmeke, L. ten Bosch, L. Boves, and B. Cranen*

Dept. of Linguistics, Radboud University  
P.O. Box 9103, NL-6500 HD  
Nijmegen, The Netherlands  
email: {J.Gemmeke, L.tenBosch, L.Boves, B.Cranen}@let.ru.nl

## ABSTRACT

This paper introduces a novel approach to exemplar-based connected digit recognition. The approach is tested for different sizes of the exemplar collection (from 250 to 16,000), different length of the exemplars (from 1 to 50 time frames) and state-labeled versus word-labeled decoding. In addition, we compare the novel method for selecting exemplars, based on Sparse Classification, with a conventional K-Nearest-Neighbor approach. For word-labeled decoding we developed a Viterbi search that applies minimum and maximum duration constraints. It appears that Sparse Classification outperforms KNN, while state-labeled decoding provides better performance than word-labeled decoding. In all conditions the performance increases with the size of the collection. However, the optimal window length is 10 frames for state-labeled decoding, but 35 frames for word-labeled decoding.

## 1. INTRODUCTION

For the last 30 years Automatic Speech Recognition (ASR) has been completely dominated by pattern recognition techniques based on Hidden Markov Models (HMMs) [1]. From a conceptual point of view this implies the assumption that (almost) all relevant phenomena in speech can be described in terms of a sequence of probabilistic models. This corresponds to the dominant position in Psycholinguistics that speech can be represented in the form of a relatively small number of discrete units (for example phonemes) that in one way or another abstract from the idiosyncrasies of individual tokens and that can be concatenated to create larger units such as syllables and words. More recently, a competing Psycholinguistic theory has been proposed, in which it is assumed that mental representations of speech include a record of detail of actual speech signals (called *episodes* or *exemplars*) that encode idiosyncrasies such as the speaker and possibly even the context in which an utterance was produced [2]. Interestingly, the episodic representations of speech proposed by that theory are reminiscent of the templates that formed the basis for the Dynamic Programming (or Dynamic Time Warping [DTW]) approach to speech recognition [3] that was superseded by HMMs in the late seventies of the previous century.

Compared to the DTW approach, HMMs had several decisive advantages: models that combined means with variances replaced templates, allowing more powerful distance measures and Viterbi decoding facilitated integrated search. HMMs were not only superior to DTW in conceptual terms, they also were better adapted to the limitations in memory and compute power of the digital computers and the algorithms that were available.

Recent advances in compute power, and the development of algorithms that can find structure in extremely large collections of observations, may make episodic approaches computationally feasible. At the same time it has become clear that not all speech phenomena can be covered in the form of HMMs. There is general agreement in the speech community about the need for novel approaches, not to fully replace HMMs, but certainly as an addition for handling phenomena that HMMs do not account for [1, 4]. Therefore, it is interesting to revisit pattern matching techniques such as DTW and episodic representations, to investigate if these can handle

problems that are notoriously difficult to solve in an HMM framework. One such problem is speech recognition in adverse acoustic conditions.

In [5] it was shown that an exemplar-based approach of isolated digit recognition in noise can outperform conventional model-based approaches. The approach, dubbed *sparse classification (SC)*, is based on the idea that all speech signals can be represented as a linear combination of suitably selected exemplars. The classification is based on finding the smallest number of labeled exemplars in a very large collection of exemplars that *jointly* approximate the observed speech token. Because there is no need for these exemplars to be close to each other in the original space, SC differs from the usual interpretation of episodic recognition and other exemplar-based approaches to speech recognition, which invariably search for exemplars with the smallest distance to the observed speech token.

As a step toward noise robust continuous speech recognition we extend our previous work on isolated digit recognition to continuous digit recognition. To investigate the impact of the manner in which exemplars are selected from a collection, we will compare SC to a K-Nearest-Neighbor (KNN) approach to selecting exemplars from a large collection. To keep the enterprise manageable we will limit our experiments to noise-free continuous digit sequences, leaving the extension to noise robustness for future research.

## 2. EXEMPLAR-BASED CLASSIFICATION

In ASR speech signals are represented as a spectro-temporal distribution of acoustic power, called a spectrogram, which in its turn is represented as a  $B \times T$  dimensional matrix (with  $B$  frequency bands and  $T$  time frames).

We express the spectrogram  $S$  as a single vector  $s$  of dimension  $D = B \cdot T$  by concatenating  $T$  subsequent time frames. We use a training corpus to create a collection  $A$  of exemplar spectrograms. The matrix  $A$  is formed as  $A = (\mathbf{a}_1 \ \mathbf{a}_2 \ \dots \ \mathbf{a}_{N-1} \ \mathbf{a}_N)$  with  $\mathbf{a}_n$ , ( $1 \leq n \leq N$ ) a specific token in the set of  $N$  available exemplars. With  $\mathbf{a}_n$  reshaped from a spectrogram just like  $s$ , the matrix  $A$  has dimensionality  $D \times N$ . Both  $s$  and the columns of  $A$  are normalized to unit (Euclidean) norm.

For speech recognition the reshaped exemplar spectrograms  $\mathbf{a}_n$  must have labels corresponding to linguistically meaningful classes (words, syllables, phonemes, subphonemic units etc.). Since  $\mathbf{a}_n$  typically contains multiple time frames, an exemplar may be associated to more than one class. For example, an exemplar may contain the trailing part of one word and the beginning of the next, as in a digit sequence *eight two*. We map every exemplar  $\mathbf{a}_n$  to a label vector  $\mathbf{y}_n$ . Denoting the total number of possible classes with  $Q$ ,  $\mathbf{y}_n$  is a binary vector of length  $Q$  of which the nonzero elements indicate with which classes  $\mathbf{a}_n$  is associated.

We obtain a label matrix  $Y$  of dimensions  $Q \times N$  by concatenating all exemplar labels  $\mathbf{y}_n$ :  $Y = (\mathbf{y}_1 \ \mathbf{y}_2 \ \dots \ \mathbf{y}_{N-1} \ \mathbf{y}_N)$ . Finally we denote the unknown label vector associated with the observed speech  $s$  as  $\mathbf{y}^s$ .

For recognition we first obtain a vector  $\mathbf{v}$  of length  $N$  that maps the observed speech vector  $s$  to exemplars in  $A$ . The weight vector

$\mathbf{v}$  tells us which exemplars are associated with the observed speech. We then obtain the label vector  $\mathbf{y}^s$  by calculating the score  $\mathbf{f}_s$  as:

$$\mathbf{f}_s = \mathbf{Y}\mathbf{v} \quad (1)$$

with  $\mathbf{v} \geq 0$  a sparse column vector and  $\mathbf{f}_s$  a vector of length  $Q$ . Keeping track of only the nonzero elements of  $\mathbf{f}_s$  we obtain the binary label vector  $\mathbf{y}^s$ . The procedure for creating label vectors  $\mathbf{f}^s$  is independent from the procedure for selecting the exemplars (SC or KNN).

### 2.1 K-Nearest-Neighbor classification

In exemplar-based speech recognition one can use K-Nearest-Neighbor (KNN) techniques to associate an unknown speech token  $\mathbf{s}$  with exemplars from the collection  $\mathbf{A}$  that are closest to  $\mathbf{s}$  given some distance measure. For every exemplar  $\mathbf{a}_n$  we obtain the Euclidean distance to  $\mathbf{s}$  and retain the  $K$  exemplars with the smallest distance. Then a binary vector  $\mathbf{v}$  of length  $N$  is created with  $K$  nonzero elements pertaining to the indices of the exemplars with the smallest distance.

### 2.2 Sparse Classification

In our sparse classification approach we try to find exemplars that *jointly* approximate the observed speech token. As in [5], we assume  $\mathbf{s}$  can be represented as a linear combination of the exemplar spectrograms  $\mathbf{a}_n$ :

$$\mathbf{s} = \sum_{n=1}^N x_n \mathbf{a}_n = \mathbf{A}\mathbf{x} \quad (2)$$

with  $\mathbf{x}$  the  $N$ -dimensional sparse representation of  $\mathbf{s}$ . Depending on the dimensionality  $D$  and  $N$ , this system of equations is *over-* or *under-*determined. It has been shown that  $\mathbf{x}$  can be recovered in both regimes by searching the *sparsest* solution [6, 7]:

$$\min_{\mathbf{x}} \{ \|\mathbf{x}\|_0 \} \text{ subject to } \mathbf{s} = \mathbf{A}\mathbf{x} \quad (3)$$

Interpreting the weights of this linear combination as the label weights, we can use the sparse representation  $\mathbf{x}$  to form the weight vector  $\mathbf{v}$  necessary for classification. Solving Eq. 3 does not guarantee non-negativity of all elements of  $\mathbf{x}$ . This is different from KNN, where all distances are guaranteed to be non-negative. Because it is difficult to imagine how negative weight could be cognitively plausible, we decided to use  $\mathbf{v} = |\mathbf{x}|$ .

## 3. CLASSIFICATION OF CONTINUOUS SPEECH

The mathematics of sparse classification requires that all exemplars in the collection  $\mathbf{A}$  have a fixed time dimension  $T$ . However, continuous speech cannot be represented as a concatenation of fixed-length units. Therefore, we use a sliding window approach. After assigning weighted labels to individual windows, we use Viterbi decoding to obtain word-based transcriptions.

### 3.1 Classification in a sliding time window

Consider a speech utterance  $U$  represented as a spectrogram with  $B$  frequency bands and  $I$  time-frames (cf. Fig. 1). We slide a window  $S$  of length  $T$  through  $U$ , with shifts of  $\Delta$  frames. The ratio of  $\Delta$  and  $T$  determines the degree with which subsequent windows overlap. Larger step sizes  $\Delta$  reduce computational effort but can decrease recognition accuracy. In this paper we keep the shift constant at  $\Delta = 1$  frame. The total number of windows we process is  $W = I - T + 1$ . Window length  $T$  is varied between 1 and 50.

Using the procedure described in Section 2 we obtain a score vector  $\mathbf{f}_w$  for every window ( $w \in W$ ). Rather than converting these to binary labels  $\mathbf{y}^w$  we use the weights as numerical scores. We collect all score vectors in a matrix  $\mathbf{F}$  of size  $Q \times W$ . Recognition then proceeds by finding the path with the best score. An example of the score matrix is shown in Fig. 2.

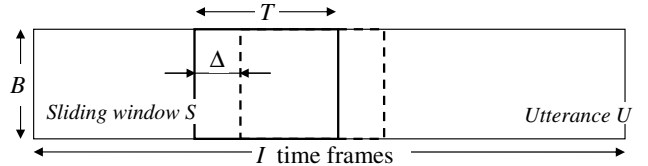


Figure 1: Schematic diagram of time-continuous classification using overlapping windows.

### 3.2 Viterbi decoding

So far, we have made no assumptions about the kind of labels that are associated with exemplars. The eventual goal of classification is a word-based transcription, but that leaves room for labels corresponding to smaller units. However, the choice of unit implies different constraints on the decoding strategy.

In conventional HMM-based speech recognition the basic units are *states*, and every word or phoneme is composed of sequences of such states. In that case, the search for the best state sequence through the matrix  $\mathbf{F}$  is constrained by the state sequences in the acoustic word models. In our current exemplar-based approach, however, we are free to associate exemplars with either states, phonemes, entire words or any other unit.

In this paper, we consider two different labels types: state-based labels and word-based labels.

#### 3.2.1 State-based labeling

In the case of *state-labeled* decoding, we used a conventional Viterbi algorithm that was available as back-end of the HMM-based speech decoder described in [8]. In this case, state transitions are constrained by the state-sequences underlying the acoustic word models, while word transitions are controlled by word entrance penalties and language models.

#### 3.2.2 Word-based labeling

In the case of *word-labeled* decoding the optimal path through the matrix  $\mathbf{F}$  must avoid frequent transitions between word hypotheses, which would lead to many insertion errors. In conventional Viterbi decoders insertions are limited by adjustment of the word entrance penalty, but it is well known that this does not necessarily prevent the emergence of very short words along the best path. We used an implementation of the Viterbi algorithm that allows incorporating minimum and maximum duration constraints to control the lengths of same-label sequences in the best path.

Above, we used indexes  $w$  for the columns (windows) and  $q$  for the rows (labels) in the matrix  $\mathbf{F}$ . However, since the modified Viterbi algorithm operates on any kind of rectangular matrix, in this subsection we will use the default notation of  $i$  and  $j$  to index the columns and rows.

The aim is to find the best path starting at *any* point in the first column ( $\{(1, j), 1 \leq j \leq Q\}$ ) to *any* point in the last column ( $\{(W, j), 1 \leq j \leq Q\}$ ) of the matrix, in such a way that the horizontal stretches of this path (i.e. parts of the path with the same  $j$ ) obey minimum and maximum length constraints. To this end we modified the conventional Viterbi algorithm as follows:

- *The set of predecessors.* In the conventional Viterbi implementation, usually the best path is sought between the lower left point  $(1, 1)$  and upper right point  $(W, Q)$  of the matrix  $\mathbf{F}$ , allowing only transitions from neighboring points  $\{(i-1, j), (i-1, j-1), (i, j-1)\}$ . In the modified Viterbi algorithm, however, we allow the set of predecessors of the point  $(i, j)$  to be the set  $\{(i-1, k)\}_{1 \leq k \leq Q}$ .
- *Transition penalties.* We modify the penalties that are associated with each transition step for two reasons: (a) the set of predecessors is larger than in conventional Viterbi and (b) a penalty  $\alpha$  must be imposed when duration constraints are violated.

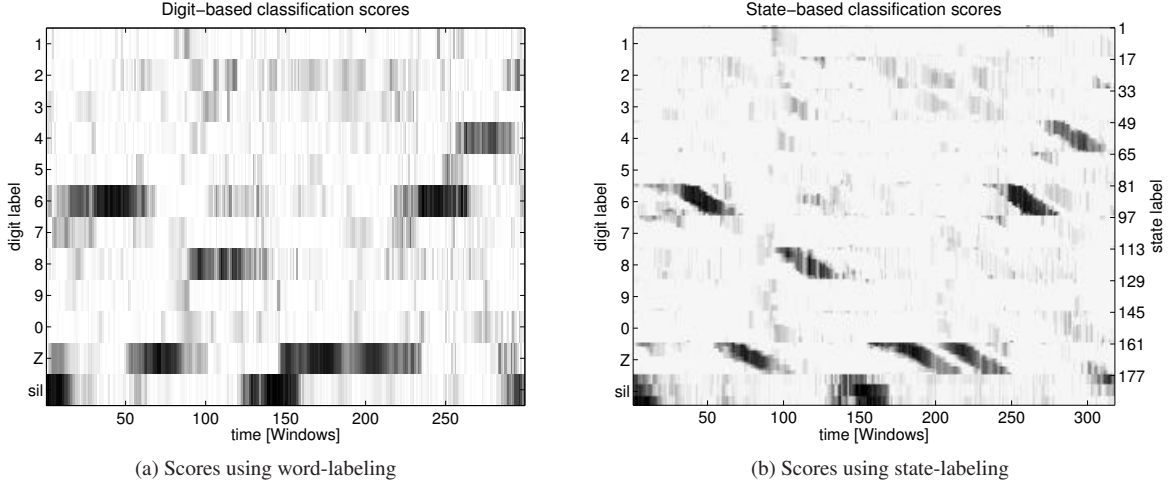


Figure 2: Example of time-continuous classification scores obtained using Sparse Classification, a window length of 20 frames and a collection of 16000 exemplars. Fig. 2a shows scores obtained using word-based class labels and the scores displayed in Fig. 2b represent HMM-state based labels. The corresponding digit labels are also shown. The utterance spoken is ‘6Z8ZZ64’, with ‘Z’ representing ‘zero’. The label ‘sil’ represents silence.

We introduce the following notation. We introduce matrix of local costs  $L = -F$ . Additionally we define two auxiliary matrices of the same size as  $L$ . The first auxiliary matrix  $G$  (which is also used in conventional Viterbi) will eventually contain the *global* costs. The second auxiliary matrix  $D$  will be used during the search such that the integer  $D(i, j)$  eventually specifies the duration of the most recent label hypothesis along the best path from the first column to  $(i, j)$ . For each  $j$ ,  $1 \leq j \leq Q$  the minimum and maximum duration is specified by means of two user-specified arrays  $\min(j)$  and  $\max(j)$ . And possibly there is a language model  $M(q_i, q_j)$  that specifies the cost of the bigram  $q_i \rightarrow q_j$  for all label pairs  $q_i, q_j$ . The term  $M$  specifies the language model costs of a transition between two labels.

The balancing between the local costs in the  $L$  matrix, the penalties in  $M$ , and penalty  $\alpha$  determines the path that is considered optimal. The higher the value of  $\alpha$ , the more expensive transitions between labels become if such a transition incurs a violation of the duration constraints; in the limit ( $\alpha \rightarrow \infty$ ), only durations of labels  $k$  are allowed between  $\min(k)$  and  $\max(k)$ .

As can be inferred from Algorithm 1, the novel Viterbi algorithm recursively updates elements of global cost matrix  $G$  while keeping track of the class index  $\hat{k}$  that minimized the global score. After the recursion, the backtrace based on the values of  $\hat{k}_i$  for  $i = W, W - 1, \dots, 1$  provides the best path through the matrix  $F$  taking into account the imposed duration constraints.

## 4. EXPERIMENTS

### 4.1 Experimental setup

For our experiments we used the clean versions of the training data and test set ‘A’ of the AURORA-2 corpus [9]. Acoustic feature vectors consisted of mel frequency log power spectra:  $B = 23$  bands with center frequencies starting at 100 Hz (frame shift = 10ms).

As a reference, a conventional HMM-based speech decoder (described in [8]) achieves 99.5% accuracy on this test set using PROSPECT features [8]. HMM-state based labels of the exemplars were obtained via a forced alignment with the orthographic transcription using the HMM-based recognizer. Digits were described by 16 states with an additional 3-state silence word. From the frame-by-frame HMM-state labels we extracted digit labels. Minimum and maximum digit durations were also extracted from the state-based transcription.

We created collections of exemplars by randomly selecting

---

### Algorithm 1: Modified Viterbi algorithm

---

**Initialization step:**

**for**  $j = 1$  to  $Q$  **do**  
 $G(1, j) = L(1, j); D(1, j) = 1;$   
**end for**

All other entries in  $G$  and  $D$  will be defined later in the search.

**Recursion step:**

**for**  $i = 2$  to  $W$  **do**  
**for**  $j = 1$  to  $Q$  **do**  
 $G(i, j) = \min_{k \in \{1, \dots, Q\}} \{G(i-1, k) + M(k, j) + C_{dur} + L(i, j)\}$   
 $\hat{k}_i =$  the minimizing value for  $k$   
 $D(i, j) = \begin{cases} D(i-1, \hat{k}_i) + 1 & \text{if } \hat{k}_i = j \\ 1 & \text{otherwise} \end{cases}$   
**end for**  
**end for**

In this scheme,  $C_{dur}$  denotes the duration violation cost which is dependent on  $i, j, k, D(i-1, k), \min(k)$ , and  $\max(k)$  and is specified as:

$$C_{dur} = \begin{cases} \alpha & \text{if } (k \neq j) \wedge \neg(\min(k) \leq D(i-1, k) \leq \max(k)) \\ 0 & \text{if } (k \neq j) \wedge (\min(k) \leq D(i-1, k) \leq \max(k)) \\ \alpha & \text{if } (k = j) \wedge (D(i-1, k) > \max(k)) \\ 0 & \text{if } (k = j) \wedge (D(i-1, k) \leq \max(k)) \end{cases}$$


---

16000 windows from the speech in the training set. We repeated the random selection for 6 window lengths, from  $T = 1$  to  $T = 50$  frames. Contrary to the experiments with isolated digits in [5] no time normalization was applied to the windows. The spectrograms of the windows were reshaped to vectors and subsequently added as the columns of the collection. For experiments with collection sizes smaller than 16000 we used the first  $N$  columns of the original collection to form the final collection  $A$ .

The speech decoding system was implemented in MATLAB. The duration penalty  $\alpha$  appeared to be not critical, and was set to  $\alpha = 10$ . We did not use a language model. The minimization in Eq.3 was approximated by the SolveLasso solver implemented

Table 1: Word recognition accuracy for several window lengths and collection sizes. The results shown here pertain to SC classification.

Window [T]	Collection size [N]						
	250	500	1000	2000	4000	8000	16000
1	5.0	10.5	22.7	31.2	38.1	43.8	49.2
5	20.9	32.9	41.0	48.5	55.3	62.1	65.0
10	37.7	44.6	58.2	63.7	68.3	70.5	73.7
20	46.7	55.7	67.5	72.7	78.5	81.1	83.2
35	49.5	63.2	71.3	78.0	82.3	84.6	85.5
50	41.8	52.0	58.4	65.5	70.4	73.7	75.6

(a) Using word-labeled decoding

Window [T]	Collection size [N]						
	250	500	1000	2000	4000	8000	16000
1	36.4	40.3	50.8	61.0	69.0	76.3	80.4
5	64.7	78.7	87.8	93.1	95.6	96.9	97.7
10	64.2	80.4	90.1	93.8	96.6	97.4	98.2
20	67.4	84.7	91.6	93.8	95.4	96.3	96.7
35	61.7	72.7	78.5	82.0	84.2	85.3	86.1
50	45.7	56.7	62.6	67.0	69.6	71.0	71.5

(b) Using state-labeled decoding

Table 2: Word recognition accuracy for several window lengths and collection sizes. The results shown here pertain to KNN classification.

Window [T]	Collection size [N]						
	250	500	1000	2000	4000	8000	16000
1	9.0	17.1	25.7	32.3	35.6	39.5	38.6
5	8.5	16.7	16.4	13.2	21.1	28.9	29.0
10	4.9	18.2	41.9	44.3	41.2	43.0	46.0
20	2.7	5.6	33.6	44.8	56.5	62.0	64.9
35	7.5	17.1	31.0	50.2	61.5	69.8	75.1
50	3.2	14.9	27.1	37.8	51.0	59.4	65.9

(a) Using word-labeled decoding

Window [T]	Collection size [N]						
	250	500	1000	2000	4000	8000	16000
1	26.3	43.6	60.4	73.1	84.9	89.1	91.9
5	29.1	62.1	83.9	91.2	92.4	92.9	93.4
10	21.1	44.9	79.0	89.8	93.1	93.9	94.5
20	12.6	31.9	64.3	81.5	89.1	91.3	92.9
35	10.1	21.4	40.3	59.6	72.0	78.0	81.6
50	11.6	14.6	30.6	43.0	55.5	62.4	67.3

(b) Using state-labeled decoding

as part of the `SparseLab` toolbox.<sup>1</sup> This iterative method was terminated after 30 iterations, resulting in  $\mathbf{x}$  having (at most) 30 nonzero coefficients. Correspondingly we use the  $K = 30$  nearest neighbors when doing KNN classification.

#### 4.2 Window length and collection size

Episodic speech recognition introduces several new parameters, such as the number of exemplars and their duration (number of frames). Because the parameters might well show significant interactions, we investigated the recognition accuracy as a function of the number of exemplars and the duration.

We carried out recognition experiments with a number of collection sizes  $N$  and window length  $T$ . We consider window lengths of 1, 5, 10, 20, 35 and 50 frames and collection sizes of 250, 500, 1000, 2000, 4000, 8000, 16000 exemplars. For every window, we first determine the associated exemplars, using KNN and SC classification. For both methods we then do decoding twice: once using word-labeling (Tables 2a and 3a) and once using the low level state-labeling (Tables 2b and 3b).

### 5. RESULTS AND DISCUSSION

#### 5.1 Word-labeling vs. state-labeling

From the recognition accuracies in Tables 2a and 3a (word-labeled decoding) and Tables 2b and 3b (state-labeled decoding) it can be seen that state-labeled decoding outperforms word-labeled decoding. This holds both for Sparse Classification and KNN Classification.

Both for word-based and state-labeled decoding the best performance is obtained with a collection size of 16000 exemplars. With marginal exceptions for KNN word-labeled decoding performance increase is monotonic with collection size. The optimum with respect to window size is different: For word-labeled decoding the best recognition accuracies are obtained with a window length of 35 frames. That number happens to be equal to the mean number of frames per digit. For state-labeled decoding the best accuracies are obtained with a window length of 10 frames. Here too, the observation holds for SC and KNN.

<sup>1</sup>This toolbox is publicly available from <http://www.sparselab.stanford.edu>

A detailed analysis (not shown here) revealed that word-labeled decoding yielded many more insertion and (to a lesser extent) deletion errors than state-labeled decoding. This is caused by the fact that in word-labeled decoding it is difficult to distinguish between the start and end of a word in the label sequence. The HMM-state models, on the other hand, are composed of a sequence of 16-states each of which must be visited in a prescribed order. As can be observed in Fig. 2a, with word-labeled decoding any spurious digit activation that exceeds the (necessarily small) minimum duration may cause an insertion error. In contrast, Fig. 2a illustrates that with the state-based decoding spurious state activations do not lead to insertions. The state sequence that underlies individual digits forces the Viterbi search to find solutions that have a clear diagonal structure.

Deletions can occur when encountering repeated words. For example, in Fig. 2 the repeated digit ‘6’ will result in a deletion error if the combined length of the two digits does not exceed the maximum duration for this digit. When doing HMM-based state decoding the distinction between the two subsequent digits is trivial.

The optimum window length seems to be one that covers a number of labels that is just large enough for the Viterbi search to harness its constraints. In word-labeled decoding window length should be larger than the minimum duration of a word, but not so long that many windows cover more than two words. For state-labeled decoding windows should be long enough to cover a state sequence that is characteristic for a word, but not so long that many windows cover more than one word. Another possibly interesting issue here is that a 10-frame window covers roughly the same number of frames that are taken into account when using delta and delta-delta features in traditional HMM-based decoding.

A decoding approach not studied in this work is using a phone-based representation (cf. [4]). It is conceivable that a phone-based representation allows for a more natural extension to large vocabulary tasks, while still providing robustness against errors like repeated digits and insertions.

#### 5.2 Window length

SC outperforms KNN for all window lengths except for single-frame windows. Apparently, SC is better able to exploit time context information than KNN. Somewhat unexpectedly, for single-frame windows, KNN classification with state-labeled decoding performs better than SC: 91.9% at  $N = 16000$  (cf. Table 3b) as opposed to 80.4% at  $N = 16000$  (cf. Table 2b). The situation is

reversed when doing word-labeled decoding but a detailed analysis showed this is again due to insertion errors.

Our current experiments do not allow to formulate a definitive explanation of why SC is worse in single-frame classification than KNN. A post mortem analysis shows that quite frequently a single example is sufficient to approximate one frame in the utterance. Most likely, this is due to the fact that the minimization in Eq. 3 is too underdetermined. If the single label returned by the solver happens to be wrong, it may be impossible for the Viterbi decoding to recover the correct path.

### 5.3 Collection size

When comparing the results with respect to collection size it is clear that we obtain higher accuracies with larger collection sizes. The increase in recognition accuracy when increasing the collection sizes beyond 4,000 exemplars is sub-linear. At the same time, the computational complexity grows faster than linear in the collection size. The sub-linear performance improvement with growing collection size is most likely due to the fact that the additional exemplars are often very similar to exemplars already present in the collection and thus contribute little, if any, additional information. This raises the question whether there are better procedures for creating a collection than just by making random selections. The answer is most probably yes, but it is far from evident how a more intelligent selection of exemplars should proceed. For SC an obvious option is some kind of greedy search, in which candidates that are too close to exemplars that are already in the collection are discarded.

It is interesting to see that SC outperforms KNN for small (< 1000 exemplars) collection sizes. This is most likely due to the fact that with KNN it may happen that all 30 exemplars that are closest to the unknown token happen to be associated to a ‘wrong’ label. This is the more likely since all chosen exemplars will be close to each other. SC on the other hand can combine exemplars ‘from all over the place’ to jointly approximate the observed speech token. This increases the possibility that the correct labels are included. Of course, SC may also select remote labels if the correct one is included among the exemplars with the highest weights. Apparently, this inclusion of ‘wrong’ labels does not have a large impact on the Viterbi decoder.

### 5.4 Noise robust ASR

The recognition accuracies obtained with SC hold promise for improving noise robustness using a Missing Data Technique (MDT) [10]. At the heart of MDT is the assumption that it is possible to estimate –prior to decoding– which spectro-temporal elements of the acoustic representations are reliable (i.e., dominated by speech) and which are unreliable (i.e., dominated by background noise). One way of noise robust speech decoding is to base recognition only on features which are labeled reliable.

The Compressive Sensing theory underlying the SC method asserts that a sparse representation of a signal can be recovered from a very limited number of measurements (features). In [5] we showed that SC can successfully recognize isolated digits in noise using very few reliable features, provided that a sufficiently accurate missing data mask is available. The current paper shows that also good recognition accuracies can be obtained with SC on a *connected* noise-free digit task. This warrants further research to see whether SC can also be made to work on connected speech in noisy conditions.

It remains to be investigated whether the KNN approach to selecting examples can also be extended to noisy speech.

## 6. CONCLUSIONS

We have extended our previous work on isolated digit recognition by applying Sparse Classification (SC) to continuous digit recognition. SC is based on the idea that arbitrary speech signals can be represented as a sparse linear combination of suitably selected exemplars. The technique works by finding the smallest number of labeled exemplars that jointly approximate the observed speech.

We compared this non-parametric technique with a conventional K-Nearest-Neighbor (KNN) approach.

For the purpose of continuous digit recognition we applied a sliding time window approach, applying SC and KNN to every window individually. The classification in individual windows is based on finding labeled exemplars that identify the observed speech token. Next, Viterbi decoding is used to find the best path through the label matrix. Connected digit recognition experiments on the clean speech test set of AURORA-2 show SC compares favorably compared to a K-nearest-neighbor (KNN) approach, achieving a recognition accuracy of 98.2% vs 94.5% for KNN.

For word-labeled decoding we developed a novel implementation of the Viterbi search that imposes minimum and maximum duration constraints on the units on the best path. Nevertheless, state-labeled decoding clearly outperformed word-labeled decoding. This is due to the fact that the local constraints in state-labeled decoding are more effective, irrespective of the duration constraints. We have also investigated the influence of window length and collection size and showed that the best recognition accuracies are obtained using as large a collection as possible and a window length of 10 frames for state-labeled decoding.

The promising results of our experiments open new directions of research in exemplar-based speech recognition and makes the way free for future research on the noise robustness properties reported on in [5].

### Acknowledgments

The research of Jort Gemmeke was carried out in the MIDAS project, granted under the Dutch-Flemish STEVIN program. Research by Louis ten Bosch is funded by the European Commission under contract FP6-034362 (ACORNS).

## REFERENCES

- [1] H. Bourlard, H. Hermansky, and N. Morgan, “Towards increasing speech recognition error rates,” *Speech Communication*, vol. 18, p. 205231, 1996.
- [2] J. Goldinger, “Echoes of echoes? an episodic theory of lexical access,” *Psychological Review*, vol. 105, pp. 251–279, 1998.
- [3] C. Myers and L. Rabiner, “Connected digit recognition using a level-building dtw algorithm,” *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 29, pp. 351 – 363, 1981.
- [4] M. D. Wachter, M. Matton, K. Demuyne, P. Wambacq, R. Cools, and D. Van Compernelle, “Template based continuous speech recognition,” *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, pp. 1377–1390, 2007.
- [5] J. Gemmeke and B. Cranen, “Noise robust digit recognition using sparse representations,” in *Proceedings of ISCA 2008 ITRW “Speech Analysis and Processing for knowledge discovery”*, 2008.
- [6] E. J. Candès and M. Wakin, “An introduction to compressive sampling,” *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, March 2008.
- [7] E. J. Candès and R. P., “Highly robust error correction by convex programming,” *IEEE Trans. Inform. Theory*, vol. 54, pp. 2829–2840, 2006.
- [8] H. Van hamme, “Prospect features and their application to missing data techniques for robust speech recognition,” in *Proc. of INTERSPEECH-2004*, 2004, pp. 101–104.
- [9] H. Hirsch and D. Pearce, “The aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions,” in *Proc. of ISCA ASR2000 Workshop, Paris, France*, 2000, pp. 181–188.
- [10] B. Raj and R. M. Stern, “Missing-feature approaches in speech recognition,” *Signal Processing Magazine*, vol. 22, no. 5, pp. 101–116, 2005.