

Compression of Neural Signals using Discriminative Coding for Wireless Applications

Stefan Craciun, David Cheney, Karl Gugel, *Member IEEE*,
Justin C. Sanchez, *Member IEEE*, and Jose C. Principe, *Fellow IEEE*

Abstract—One of the most critical tasks when designing a portable wireless neural recording system is to limit power consumption. This paper proposes a new compression technique applied to neuronal recordings in real-time. The signal is compressed before transmission using a discriminative vector quantization algorithm and then it is reconstructed on the receiver side. Results show that power consumption is decreased while more efficiently using the limited bandwidth. A discriminative Linde-Buzo-Gray algorithm (DLBG) preserves action potential regions of the neuronal signal where information is contained while efficiently filtering background activity. The compression algorithm has been tested in real time on a hardware platform (PICO DSP [3]) that has a Digital Signal Processor (DSP) which performs the algorithm before sending the compressed data to a wireless transmitter. The compression ratios obtained range between 20:1 and 70:1 depending on the embedding size of the signal and the number of code-vectors used.

I. INTRODUCTION

In the design of Brain-Machine Interfaces (BMI), portability of the prosthetic system is a key factor during use in the activities of daily life (ADL). An important component in the design of these portable neural recording and decoding systems is the transmission of neural data from the brain to a device to be processed and transformed into action. This would provide many benefits such as freedom of motion, no risk of entanglement or detached wires, less risk of infection to name just a few. Engineering a wearable or implantable neural recording system for neural interfaces provides a number of challenges. An important constraint is the large bandwidth required to send the neural information. It is not uncommon to sample hundreds of electrodes, which are traditionally connected through wires to different devices. By sampling each electrode at 20 kHz and then using 16 bits to digitize the signal (A to D conversion) the total bandwidth required for only one channel is 0.31 Mbps. This means that for a 64 array of electrodes the total bandwidth is increased to 20 Mbps. This large bandwidth requirement for multiple channels is impossible to sustain on a battery-operated system with limited power consumption and bandwidth. It is therefore

critical that the information is preprocessed before it is sent through the wireless channel. With these design constraints in mind, we seek to develop a more efficient method of data transmission. Our proposed solution is to design a real-time compression algorithm that will discriminate between parts of the signal representing key information for decoding motor intent (action potentials) and the remaining parts which represent the background activity. The current proposed solutions for neural data compression is to use a threshold voltage as a boundary which separates low amplitude background activity and noise from high amplitude spikes of information. Only signals above the set threshold are detected as action potentials and sent through the wireless channel while any signal below this threshold is discarded and lost. This method uses the bandwidth more efficiently by sending only the neuron spikes that surpass the set threshold, but, choosing a threshold limit is a difficult task. There is the possibility of discarding an action potential below the threshold voltage or sending noise which momentarily surpasses the same threshold.

Our approach focuses on designing a compression scheme which avoids having to send only part of the signal and losing the lower amplitudes. The discriminative coding will favor the representation of the signal (action potentials) while under-representing the background portions but will not discard the any part of the signal. This compression algorithm has been tested in real-time on a hardware platform called Pico DSP and PICO Remote [3, 4]. The wireless transmitter is an off the shelf development board whose bandwidth is limited to sending raw data from only one electrode (0.31 Mbps). The Pico Remote is capable of sampling a total of 64-channels of neural electrodes in parallel. By using the DLBG algorithm we were able to achieve compression ratios in excess of 64:1 which means that neural signals from all 64 electrodes can be simultaneously sent through the wireless link. The action potentials will be reconstructed with little MSE while the background activity will be under-represented but no threshold value is used as a boundary between the two. Finally we demonstrate that when spike sorting the original neural signal and the reconstructed one with the same spike templates very similar results are obtained. The neural signals used in the experiments were extracted from a microwire electrode chronically implanted in layer V of a rat motor cortex. This neural signal is representative of an average chronic neural recording having a signal to noise ratio (SNR) of 23 dBs.

A. Theory

Our discriminative coding technique is a derivation of the

Manuscript received January 15, 2009. This work was supported in part by the NSF PFI 0650161 grant.

Stefan Craciun, David Cheney, Karl Gugel, and Jose C. Principe are with the Department of Electrical and Computer Engineering at the University of Florida, Gainesville, FL 32611 USA (email: [craciuns, gugel, djcheney]@ufl.edu, [principe]@cnel.ufl.edu).

Justin C. Sanchez is with the Department of Pediatrics, Division of Neurology, University of Florida, Gainesville, FL 32611 USA (email: jcs77@ufl.edu).

well-established vector quantization methods [1]. It is an improved lossy approximation technique where the input data (the neural signal) is approximated by code-vectors. All the code-vectors form what is known as a code-book which is used to encode and then decode the compressed signal. Normally a LBG algorithm [1] would distribute the code-vectors evenly among the input space treating all points with equal importance. In our case we discriminate between the background activity and action potentials when assigning the code vectors.

When mapping the neural signal to a two dimensional feature space (2 D embedding) we obtain the point distribution shown in figure 1.

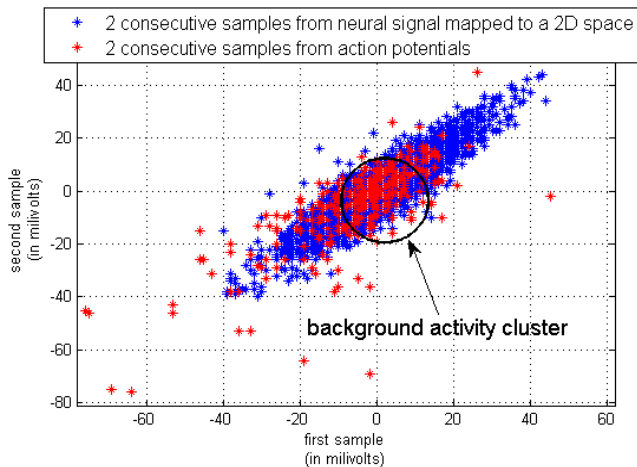


Fig 1 Overlap of action potential samples (in red) and background activity samples (black circle) for low embedding size (2D)

Each point in this figure represents two consecutive samples from the neural signal. The voltage amplitude (in millivolts) of the first sample is the x coordinate and the voltage level of the second sample is the y coordinate of this point. The total number of samples used from the neural signal is 40000. When mapped to a 2 D space this signal will generate 20000 points shown as blue crosses in fig 1. The action potentials within this same 20 second neural recording are extracted and again plotted in the same feature space (red crosses in fig 1). It is observed that for low dimensional embedding (2D or 3D) there is significant overlap between points extracted from the action potentials and the background activity cluster centered around the origin. For spike sorting it is essential that the depolarization and hyperpolarization phases of the action potential be easily discernable because they provide features for classification. In 2-D space however, these sections are mapped to the same region as background activity (lower voltage levels). The solution to this problem is to map the neural signal to a higher dimensional feature space (larger embedding size 9, 10). Fig. 2 shows a histogram of normalized Euclidean distances measured from each action potential point to the origin. Notice how for a 2D embedding the majority of Euclidean distances lie close to the origin (0, 0). For a higher embedding size (10D) the normalized Euclidean distances increase and shift away from the origin. This means that action potential samples are now separated from the background activity which will always lie in close proximity of the origin.

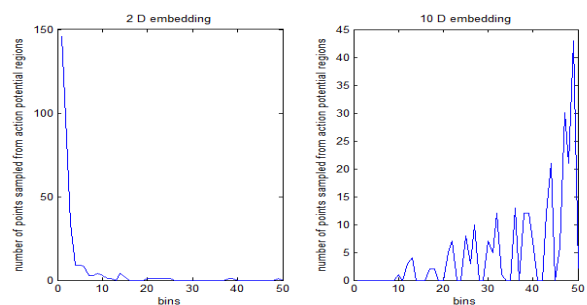


Fig 2 The normalized Euclidean distance measured from AP samples to the origin increases as a function of dimensionality

The DLBG algorithm has two approaches in generating the code-vectors of the compressor. The first method which is also more computationally efficient, assigns weights to the feature space as a function of entropy. First the Euclidean distance to the origin is measured for every point in the feature space. For a n dimensional feature space the Euclidean distance between two points $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ is calculated

$$\text{by: } \sqrt{\sum_{i=1}^n (a_i - b_i)^2} \quad (\text{in our case } B = (0, 0, \dots, 0) \text{ is the origin})$$

Since action potentials have higher voltage amplitude, samples from these regions will have a larger Euclidean distance while background activity samples will have a smaller Euclidean distance. A histogram of the Euclidean distances is computed by binning the distances into 1000 regions. Figure 3 shows the histogram of a 2D feature space. As expected the majority of points lie in the first few bins (background activity samples occur frequently) with the bin size decreasing as the distance from the center is increased (this is where action potentials reside).

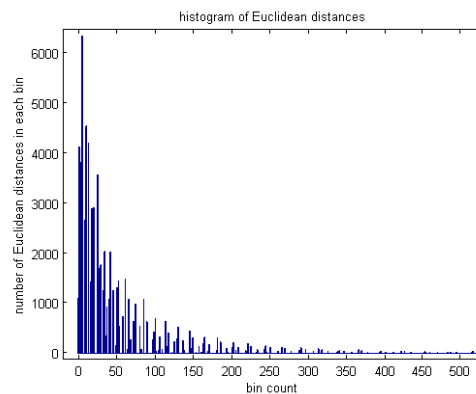


Fig 3 Histogram of Euclidean distances from 2D feature space

Let $p(i)$ be the probability that a point from the feature space resides in bin i . $p(i)$ is the ratio of the number of points in bin i to the total number of points in the feature space. The entropy of bin i is calculated as: $H(i) = -p(i) * \log_{10}(p(i))$

$H(1)$ through $H(1000)$ will represent the weights of the regions so that all 1000 bins will be assigned a particular weight based on the entropy of that region. Once each region of space has been assigned a particular weight the DLBG algorithm is used to generate the code-vectors [1, 2]. Even though this method is computationally efficient and takes advantage of the structure of the data it does not take into

account any prior information of the action potentials. A better solution would be to extract the action potentials from the neural signal and assign weights as a function of the mutual information calculated between the neural signal and the extracted action potentials. This is the second DLBG method which insures that regions of the feature space where action potentials exist have much higher weights than regions where no information is stored (background activity). However, this method requires more computation and has the disadvantage of having to locate the action potentials in the training data set. Once the action potentials have been extracted we compile all of them into a new signal and project it to a multi-dimensional feature space. Again the Euclidean distances to the origin are binned into 1000 regions and the feature space is partitioned just as before into 1000 regions (bins).

Let $p_x(i)$ be the probability that a point in the feature space resides in bin i , then:

$$H(X_i) = -p_x(i) * \log_{10}(p_x(i))$$

is the entropy of bin i .

Similarly let $p_y(i)$ be the probability that an action potential point resides in bin i of the feature space, then:

$$H(Y_i) = -p_y(i) * \log_{10}(p_y(i))$$

is the entropy of bin i for action potentials.

Let p_{x_i, y_i} (joint probability) be the probability that a point in the feature space resides in bin i and is sampled from an action potential, then: $H(x_i, y_i) = -p_{x_i, y_i} * \log(p_{x_i, y_i})$ is the joint entropy of bin i . The mutual information of bin i calculated between action potentials and the entire neural signal is: $I(X_i; Y_i) = H(X_i) + H(Y_i) - H(X_i, Y_i)$

$I(X_i; Y_i)$ represents the weight given to bin i . Once the 1000 regions have been assigned a weight by the mutual information calculations we proceed again to using the DLBG algorithm [1], [2] to distribute the code-vectors of the compressor over the feature space.

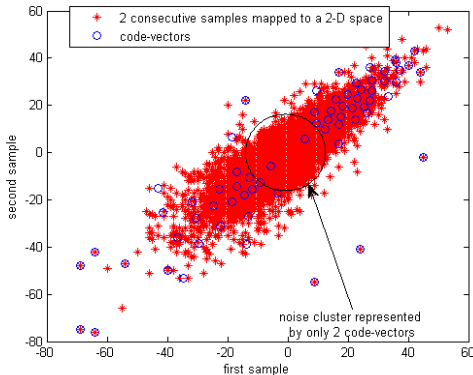


Fig 4: Code vector distribution over the feature space

As can be observed from fig. 4 the code-vectors of the compressor will have a much higher density in the regions where action potentials reside. This means that action potentials will be accurately reconstructed while background activity will be underrepresented but not completely discarded.

II. SIMULATIONS AND RESULTS

In order to test the performance of the DLBG algorithm we used a 20 seconds training data set to generate the code-vectors. The neural signal is digitized and recorded on a TI MSP430 microprocessor [3]. The microprocessor will clock the 12 bit samples into a CPLD at a 20 MHz. The CPLD will then buffer the signal temporarily. Finally the Pico DSP [3, 4] will read the neural recordings from the CPLD chip and perform the compression. It is important to mention that the set of code-vectors generated for one neural channel can only be used to compress that same channel. Another neural channel will have action potentials that are mapped to a different region of the feature space and so another set of code-vectors specific to those action potentials should be used.

A table of 32 code-vectors is first generated by applying the DLBG algorithm to the training data. The DSP then executes the compression algorithm by finding the smallest Euclidean distance from the sampled data to every code-vector and selects this code-vector to approximate the data. Table 1 shows the codebook of the compressor. Each code-vector in Table 1 is represented by a unique binary number. The most frequently used code-vector (1st one) is represented by one bit in the codebook. As the code-vectors become less probable to occur more bits can be used to encode them. This technique maximizes the compression ratio.

TABLE 1
15-DIMENSIONAL CODE VECTORS AND BINARY REPRESENTATION

15 Dimensional Codevector	Binary representation	Probability of occurrence
1	0	93.4%
2	10	0.09%
3	110	0.084%
...	...	
32	11111110	0.01%

The same codebook is used to decode the information and reconstruct the original neural recording. The compression ratio is calculated as the ratio of the raw 16 bit neural recordings to the number of bits transmitted through the wireless channel. To demonstrate the range of compression ratios achievable with this technique we begin with a 2-D feature space and increase it to 15 D. Fig 5 shows a graph of the compression ratio as a function of embedding size when the number of code-vectors is held constant (64).

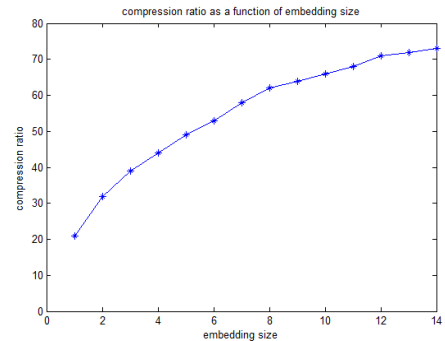


Fig 5 Compression ratio as a function of embedding size

The two parameters that influence the overall performance of the DLBG algorithm are the number of code-vectors of the compressor and the embedding size (dimensionality of the

feature space). Choosing the optimal values for these parameters will depend on the SNR of the neural signal and how well the action potentials will cluster in the feature space.

Figure 6 shows the reconstruction of the same action potential for two different embedding sizes (2 and 10) using the same number of code-vectors in both cases (64).

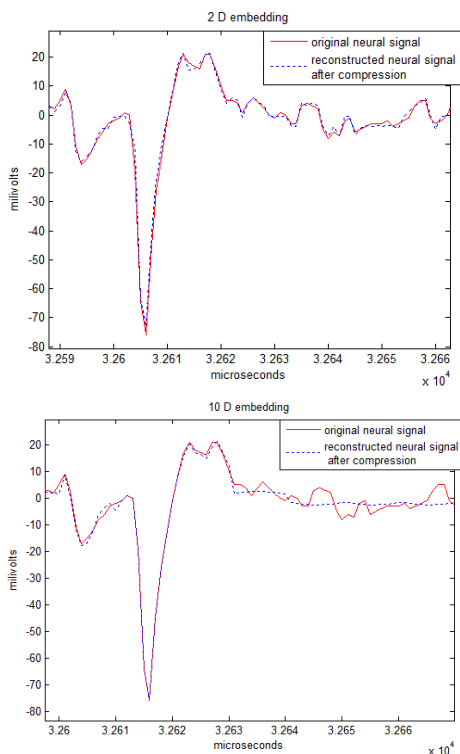


Fig. 6 Reconstruction of an action potential for 2D vs. 10D embedding

It is observed that the reconstruction for this particular action potential is identical in both scenarios but the higher embedding reconstruction filters the noise more efficiently.

Ultimately what should determine the embedding size and the number of code-vectors used is how well a spike sorting software (such as spike 2) can match the two waveforms. We quantified this by first spike sorting the original neural signal. We obtained a template for the different neurons and applied the same template to the reconstructed neural signal. Fig. 7 below shows the two templates and the template limits. The main difference in the two templates is the amplitude while the regions of depolarization and hyperpolarization look almost identical (worst case scenario).

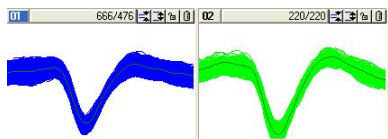


Fig 7 Two action potential templates and their limits (above)

In order to quantify how successful our compression scheme is, we generated several reconstructions with different normalized MSEs (for action potential regions). Figure 8 shows how MSE influences the percentage of action potentials correctly classified.

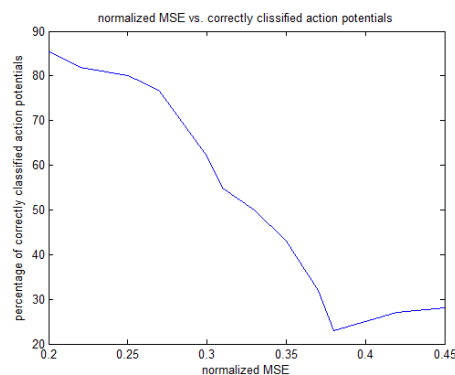


Fig 8 Dependence of correct classification on MSE

Even if the templates represent the worst case scenario we managed to obtain a maximum of 85.4% correct classification of the 2 action potentials for a 120 seconds time period. The MSE is heavily dependent on the number of code-vectors used while the compression ratio depends almost entirely on the embedding. Choosing the embedding size and the number of code-vectors thus becomes an easier task. The embedding size should be increased to improve the compression ratio while the number of code vectors should be increased to lower the MSE and increase the percentage of correct classification.

III. CONCLUSION

The discriminate LBG algorithm was successfully used to compress neural signals extracted from a microwire electrode chronically implanted in layer V of a rat motor cortex. By using the DLBG compression technique the bandwidth was decreased by more than 64 times. This means that 64 more neural channels can be transmitted through the wireless link. Higher compression ratios can be achieved by mapping the input signal to a high enough dimensional space where separation between action potentials and noise becomes large. The MSE of reconstruction can be decreased by using a larger number of code-vectors. When adding code-vectors to the feature space the density of code-vectors in the action potential regions will increase while the density of code-vectors in the noise region will remain constant. This means that the compression ratio slightly increases while action potentials can be correctly classified by spike sorting software. The number of code-vectors of the compressor and the embedding size are two parameters that can be used to fine tune the DLBG algorithm to the desired performance.

REFERENCES

- [1] Y Linde, Buzo A., and Gray R.M., "An algorithm for vector quantizer design," *IEEE Trans. on Communications*, vol. 28, pp. 84-95, Jan. 1980.
- [2] Rao, S.; Paiva, A.R.C.; Principe, J.C., "A Novel Weighted LBG Algorithm for Neural Spike Compression," *International Joint Conference on Neural Networks*, pp.1883-1887, 12-17 Aug. 2007.
- [3] G. Cieslewski, D. Cheney, K. Gugel, J. Sanchez, and J Principe, "Neural Signal Sampling via the Low Power Wireless Pico System", *Intl. Conf. of Engineering in Medicine and Biology Society*, pp. 5904-5907, Aug. 2006.
- [4] Cheney, D.; Aik Goh; Jie Xu; Gugel, K.; Harris, J.G.; Sanchez, J.C.; Principe, J.C., "Wireless, In Vivo Neural Recording using a Custom Integrated Bioamplifier and the Pico System," *Neural Engineering, 2007. CNE '07. 3rd International IEEE/EMBS Conference on Neural Engineering*, vol., no., pp.19-22, 2-5 May 2007.