# Spike Sorting Using Hidden Markov Models

Hailing Zhou, Shady Mohamed, Asim Bhatti, Chee Peng Lim, Nong Gu,
Sherif Haggag, and Saeid Nahavandi

Centre for Intelligent Systems Research
Deakin University, Australia
{hailing.zhou,shady.mohamed,asim.bhatti,chee.lim,nong.gu,
sherif.haggag,saeid.nahavandi}@deakin.edu.au

**Abstract.** In this paper, hidden Markov models (HMM) is studied for spike sorting. We notice that HMM state sequences have capability to represent spikes precisely and concisely. We build a HMM for spikes, where HMM states respect spike significant shape variations. Four shape variations are introduced: silence, going up, going down and peak. They constitute every spike with an underlying probabilistic dependence that is modelled by HMM. Based on this representation, spikes sorting becomes a classification problem of compact HMM state sequences. In addition, we enhance the method by defining HMM on extracted Cepstrum features, which improves the accuracy of spike sorting. Simulation results demonstrate the effectiveness of the proposed method as well as the efficiency.

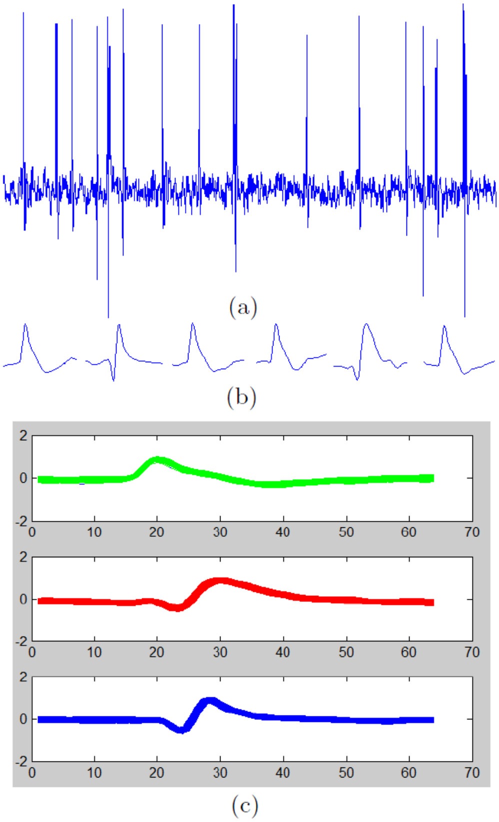
**Keywords:** Spike sorting, HMM, Cepstrum, confusion matrix.

## 1 Introduction

It is observed that complex brain processes are reflected by activities of millions of neurons. To study brains, research on understanding neuron actions is crucial. Electrodes are implanted in brains to record actions of surrounding neurons through firing potentials. The action potentials are also referred as "spikes" in neuroscience, as they appear sharp spikes in the signal waveforms. Each neuron produces spikes with a particular shape. Spike sorting is to cluster these recorded spikes into groups. In each group, spikes have similar shapes. The ultimate goal of spike sorting is to find the correspondence between spikes and neurons. Based on this research, the possibilities of new investigations on brains will be increased dramatically.

The assumptions for spike sorting are that the shapes of spikes from a specific neuron are similar and they are unique for each neuron [5]. Many spike-sorting techniques have been developed [1,2]. The challenges of spike sorting lie in that (i) the number of neurons is unknown; (ii) the spike recording is associated with physical and biological noise [4]; (iii) spikes in the local area are not easy to be distinguished [2]. Traditional spike sorting methods [10,11] usually rely on shape measurement such as comparing height, width, and peak-to-peak amplitude of

M. Lee et al. (Eds.): ICONIP 2013, Part I, LNCS 8226, pp. 553–560, 2013.



(a)

(b)

(c)

**Fig. 1.** The illustration of our method. (a)Raw data recorded from electrodes; (b)spike detection using a threshold-based method; (c) clustering the detected spikes into three groups automatically using HMM.

spikes. However, these approaches often produce inaccurate clustering because of the sensitivities to noise.

Statistics tools are widely used for spike sorting. In [6], Principal Component Analysis (PCA) is introduced to group spikes through analyzing spikes to get several principal components and projecting spikes into each component. Takahashi et al. apply Independent Component Analysis (ICA) [7,8] to separate spikes. As both PCA- and ICA-based methods require strong spike correlation and variances, they will not work well in case of low signal to noise ratio (SNR). In [12], Pouzat et al introduce a Markovian approach with a Monte Carlo simulation to solve spike sorting. However, high accuracy is achieved at the expense of a great computational complexity.

In this paper, we propose an effective and efficient method for spike sorting shown in Fig.1. The methods consists of two major procedures: spike detection and representation [3] and HMM based spike clustering. The main contribution

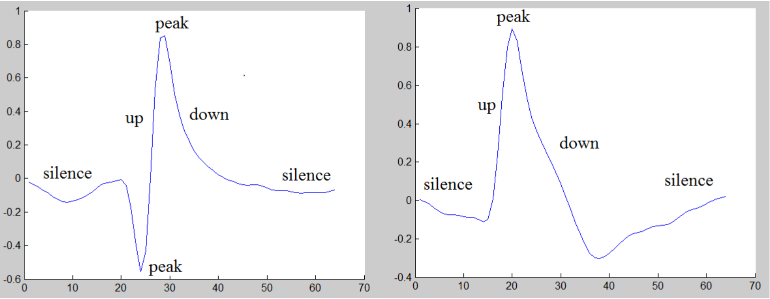


**Fig. 2.** Significant shape variations in spikes, which are assigned to HMM states

is the use of HMM to cluster the shapes of different spikes. For each spike, we observe that it contains four significant shape variations: silence, going up, peak and going down. We partition each spike into several segments and define observations of HMM as these spike segments. The states of HMM are corresponding to the four shape variations. The expectation maximization(EM) algorithm is used to compute the HMM parameters. The Viterbi algorithm is then used to find the most likely state sequence that correspond to each spike shape. After that, the sorting of spikes becomes to classify the obtained state sequences. Experiments demonstrate the effectiveness and efficiency of the proposed method.

The rest of the paper is organised as follows: Section 2 discusses the use of HMM for spike sorting. Section 3 shows the experimental results.

## 2 HMM-Based Spike Sorting Method

HMM is a statistical tool to model sequences and describe the probability distribution over a set of observations, which has been successfully used for speech recognition. There are five basic elements in a HMM: 1) the number of states $N$; 2) the number of observations $M$ ; 3) the state transition probability matrix $A$; 4) the observation probability matrix $B$ and 5) the initial state distribution $\Pi$. The triplet $\lambda = (A, B, \Pi)$ is often used to denote a HMM.

### 2.1 Spike HMM

In this paper, we utilize a HMM to model spikes in order to understand its underlying states over a sequence of observations. For a spike, the significant shape states are silence (flat), going up, peak and going down, shown in Fig. 2. Each of them is assigned to a state (from s1 to s4) of the HMM (i.e. $N = 4$). The state structure of the spike HMM and the state transitions with non-zero probability are shown in Fig. 3. It is noticed that for a general spike, the transitions between s2 (up) and s3 (down) or between s1 (silence) and s4 (peak) have zero possibilities. Moreover, most spikes start from the silence state and

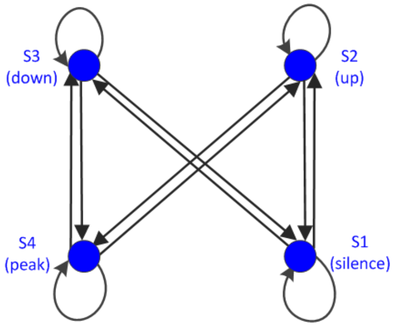


**Fig. 3.** HMM with 4 states (labeled s1 to s4) and 12 state transitions. The transition probabilities between s2 and s3 or between s1 and s4 are zero.

end in the silence state. Based on the basic knowledge about spike, the following transition probability matrix and the initial state distribution are given for the initialization of the HMM.

$$A = \begin{pmatrix} 0.4\ 0.3\ 0.3\ 0.0 \\ 0.3\ 0.4\ 0.0\ 0.3 \\ 0.3\ 0.0\ 0.4\ 0.3 \\ 0.0\ 0.4\ 0.4\ 0.2 \end{pmatrix}$$

$$\Pi = \begin{pmatrix} 0.6\ 0.2\ 0.2\ 0.0 \end{pmatrix}$$

## 2.2 Spike Detection and Feature Extraction

Given a sequence of neuron signals for spike sorting, the first step is the detection of the spikes. We adopt the amplitude thresholding method [3], which can quickly locate spikes. A bandpass filtering (bf) is firstly applied to the input signals $S$ and then the standard deviation $\sigma$ of the background noise is estimated using

$$median\{|S_{bf}|/0.6745\}. \tag{1}$$

The threshold is set to be $4\sigma$.

After spike detection, $L$ samples are saved for each spike. A typical value of $L$ is set to 64. We divide a spike into overlapping segments of length of $l$. The amount of overlap between consecutive segments is $o$. The number of segments is actually the number $M$ of HMM observations, which can be calculated by using $M = (L - o)/(l - o)$. The choice of parameters $l$ and $o$ significantly affects the spike clustering accuracy. With an amount of overlap between consecutive spike segments, more spike features can be captured to improve the performance. It is very delicate to choose the parameter $l$. If $l$ is too small, there is no sufficient discriminant information in the observation. If $l$ is too large, the probability of cutting across the distinct spike features are increased. Based on the knowledge of spikes, the number of samples that cover a whole peak (i.e. s2-s4-s3 or s3-s4-s2) of a spike is about 20. $l$ set to the half value (i.e. 10) is a reasonable choice, which has been proved in our experiments. Refer to Fig. 4 for illustration.

Based on the partition of a spike, we obtain $M$ observations for the HMM. For each observation vector, it has $l$ samples of the raw waveform data. The use of raw data as HMM observation vectors has a disadvantage of being sensitive to noise. To reduce the effect of noise, Cepstrum coefficients are extracted from each spike segment. We employ Cepstrum features because they have a property to capture both the amplitude property of spikes and the phase of the initial spectrum. This property makes Cepstrum features be able to separate the meaningful features from noisy signals. Cepstrum coefficients are calculated as the Inverse Fourier Transform (IFT) of the logarithm of the Fourier Transform of a spike segment, as

$$c = F^{-1} log|F(s)|, \tag{2}$$

where $s$ is the spike segment and $c$ is the Cepstrum feature vector.

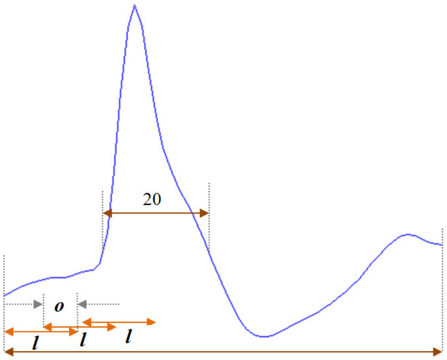


**Fig. 4.** Spike parameterization and partition. There are $L$ samples for a spike. The spike is partitioned into $M$ segments with each length of $l$ and overlap of $o$ between neighboring segments.

### 2.3 HMM Training and Clustering

A set of 300 detected spikes are used to train a HMM $\lambda = (A, B, \Pi)$, where there are $300 \times M$ observation vectors with $l$ Cepstrum coefficients for each. In the HMM, the number of states $N$ is 4, the number $M$ of observation vectors is defined in Section 2.2, the transition probability matrix $A$ and the initial state distribution $\Pi$ are given in Section 2.1. The only unknown parameter is the observation probability matrix $B$. We adopt the mixture of Gaussians with the number of 3 to initialize $B$. In order to optimize the HMM to maximize the probability of the model generating the observations, we adjust the parameters $A$, $B$, and $\Pi$ using the EM procedure [14].

In the clustering stage, the optimized HMM is used to group spikes. Every spike is represented by $M$ observation vectors, same as the description above.

**Table 1.** Statistics of the number of spikes in each generated clusters using the proposed method and the method of [3]

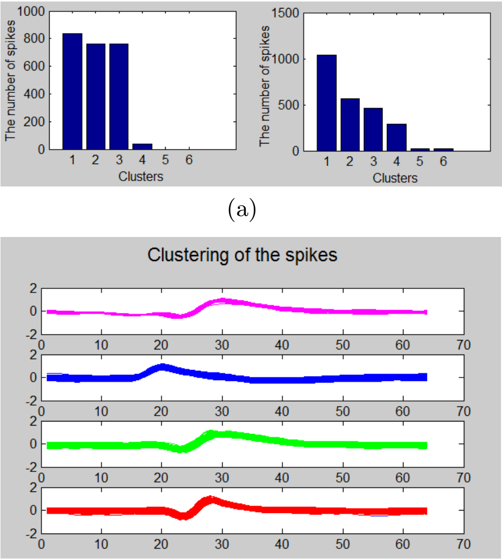
| Methods | The proposed method | | | The method [3] | | |
|---|---|---|---|---|---|---|
| Spike sets | 0.05 | 0.10 | 0.15 | 0.05 | 0.10 | 0.15 |
| Cluster 1 | 807 | 837 | 878 | 810 | 952 | 962 |
| Cluster 2 | 756 | 761 | 724 | 753 | 562 | 466 |
| Cluster 3 | 756 | 711 | 629 | 741 | 477 | 456 |
| Cluster 4 | - | 85 | 97 | 15 | 370 | 398 |
| Cluster 5 | - | - | 20 | - | 19 | 40 |
| Cluster 6 | - | - | - | - | 14 | 23 |
| Total cluster Num. | 3 | 4 | 5 | 4 | 6 | 6 |
| Dominant cluster Num. | 3 | 3 | 3 | 3 | 1 or 4 | 1 or 4 |

The Viterbi algorithm [15] is employed to find the most likely state sequence in the model that produces the observations. The output state sequence consists of $M$ elements received values from 1 to 4 that are corresponding to s1 to s4 respectively. As the order of states respects primary spike geometry that is dominant to distinguish different spikes, state repetitions are removed from output state sequences by using the $diff$ and $find$ operations in MATLAB. Take the spike in Fig. 4 for an instance. The output state sequence $(1, 1, 2, 2, 4, 3, 3, 3, 1, 1)$ is reduced to be $(1, 2, 4, 3, 1)$. After that, spike sorting becomes the clustering of the unique state sequences, which can be easily implemented by using the *unique* operation in MATLAB.

## 3 Experimental Results

In this part, we show our experimental results on simulation data given in [3]. In the simulation data, it is known that there are 3 spike types. In addition, the spikes with different noise levels are provided as well as the true classes for each spike. In the experiment, there are 3 sets of spikes with 3 noise levels: 0.05, 0.1 and 0.15 respectively. We implement our method using MATLAB on Intel Core i7-2600K. To sort 2319 spikes from 3 classes, the proposed classification method siginificantly improves the computational complexity. The classification process takes about 10 seconds to complete. We set the parameters as $l = 10$, $o = 4$ and $M = 10$. Other parameters are described in the Section 2. To evaluate the performance of our proposed method, we compare our results with the method in [3] regarding the number of generated clusters and the spikes in each cluster. The clustering accuracies are also compared via the confusion matrix that is introduced in [13] specifically for evaluating clustering techniques.

Table 1 summarizes the results, where the number of spikes in each cluster is listed. It should be noted that the number of total clusters and the number of dominant clusters, referring to Fig. 5(a) for illustration. Based on the results, it



(a)

(b)

**Fig. 5.** Experimental results. (a) The illustration of the number of spikes in each generated clusters using the proposed method (left) and the method [3] (right). (b) The spikes are grouped into 3 major clusters (blue, green and red) using the proposed method.

**Table 2.** Performance comparison using confusion matrices

| Methods | The proposed method | | | The method [3] | | |
|---|---|---|---|---|---|---|
| | Class 1 | Class 2 | Class 3 | Class 1 | Class 2 | Class 3 |
| Class 1 | 235 | 11 | 17 | 239 | 29 | 21 |
| Class 2 | 5 | 292 | 15 | 7 | 272 | 12 |
| Class 3 | 11 | 18 | 280 | 5 | 20 | 279 |
| Accuracy | 91.2% | | | 88.9% | | |

is obvious that the number of dominant clusters using our proposed method is more distinct than the one using [3] and our proposed method can produce 3 (the actual number of spike classes) dominant clusters for all the data with different SNRs. The spikes in each group are shown in Fig. 5(b), where we can see that the shape variations of spikes in each dominant cluster are small. Moreover, the clustering accuracy is measured using a confusion matrix given in Table 2, where we divide the spikes into two parts - one for building prototypes of clusters and the other for testing the accuracy using the Euclidean distance. The comparison indicates that our method produces more accurate spike sorting.

## 4 Conclusion

We presented an effective and efficient spike sorting method. A threshold-based method is applied to quickly locate spikes. Cepstrum coefficients are calculated to robustly represent spike segments. The main contribution is the introduction of HMM for spike sorting, where four HMM state are defined and searched to respect the spike geometry. Experimental results demonstrate the superiorities of our proposed method.

## References

1. Brown, E.N., Kass, R.E., Mitra, P.P.: Multiple Neural Spike Train Data Analysis: State-of-the-art and Future Challenges. Nature Neuroscience 7, 456–461 (2004)
2. Lewicki, M.S.: A Review of Methods for Spike Sorting: The Detection and Classification of Neural Action Potentials. Network: Computation in Neural Systems 9(4), R53–R78 (1998)
3. Quiroga, R.Q., Nadasdy, Z., Ben-Shaul, Y.: Unsupervised Spike Detection and Sorting with Wavelets and Superparamagnetic Clustering. Neural Computation 16, 1661–1687 (2004)
4. FMitra, P., Bokil, H.: Observed Brain Dynamics. Oxford university Press (2007)
5. Samiee, S., Shamsollahi, M.B., Vigneron, V.: A New Markovian Approach Towards Neural Spike Sorting. In: 8th International Conference on Information, Communications and Signal Processing, pp. 1–5 (2011)
6. Lagerlund, T.D., Sharbrough, F.W., Busacker, N.E.: Spatial Filtering of Multichannel Electroencephalographic Recordings through Principal Component Analysis by Singular Value Decomposition. Journal of Clinical Neurophysiology 14(1), 73–82 (1997)
7. Takahashi, S., Sakurai, Y., Tsukada, M.: Classification of Neuronal Activities from Tetrode Recordings using Independent Component Analysis. Neurocomputing 19(1-4), 289–298 (2002)
8. Takahashi, S., Anzai, Y., Sakurai, Y.: Automatic Sorting for Multineuronal Activity Recorded with Tetrodes in the Presence of Overlapping Spikes. Journal of Neurophysiology 89(4), 2245–2258 (2002)
9. Takahashi, S., Anzai, Y., Sakurai, Y.: Automatic Sorting for Multineuronal Activity Recorded with Tetrodes in the Presence of Overlapping Spikes. Journal of Neurophysiology 89(4), 2245–2258 (2002)
10. Dinning, G.J.: Real-time Classification of Multiunit Neural Signals using Reduced Feature Sets. IEEE Transactions on Biomededical Engineering 28, 804–812 (1981)
11. Schmidt, E.M.: Computer Separation of Multi-unit Neuroelectric Data: A Review. Jouranl of Neuroscience Methods 12, 95–111 (1984)
12. Pouzat, C., Delescluse, M., Viot, P.: Improved Spike-Sorting By Modeling Firing Statistics and Burst-Dependent Spike Amplitude Attenuation: A Markov Chain Monte Carlo Approach. Journal of Neurophysiology 91, 2910–2928 (2004)
13. Stehman, S.V.: Selecting and Interpreting Measures of Thematic Classification Accuracy. Remote Sensing of Environment 62(1), 77–89 (1997)
14. Baum, L.E., Petrie, T., Soules, G., Weiss, N.: A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. The Annals of Mathematical Statistics 41(1), 164–171 (1970)
15. Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. Proceedings of the IEEE 77(2), 257–286 (1989)