ESANN'2006 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 26-28 April 2006, d-side publi., ISBN 2-930307-06-4.

Linking non-binned spike train kernels to several existing spike train metrics

Benjamin Schrauwen* Jan Van Campenhout
ELIS, Ghent University, Belgium
Benjamin.Schrauwen@UGent.be

Abstract. This work presents two kernels which can be applied to sets of spike times. This allows the use of state-of-the-art classification techniques to spike trains. The presented kernels are closely related to several recent and often used spike train metrics. One of the main advantages is that it does not require the spike trains to be binned. A high temporal resolution is thus preserved which is needed when temporal coding is used. As a test of the classification possibilities a jittered spike train template classification problem is solved.

1 Introduction

Spike train classification is of interest to neurobiological research where spike trains from real neurons need to be related to stimuli; and to neural network research where spiking neurons get increasing attention due to their greater computational power and their inherent ability to process temporal information [4]. Traditionally, distance metrics for spike trains use binning of the spike trains so that a so called instantaneous firing rate is obtained on which the euclidean distance is defined. However, these metrics do not capture the underlying biological variance of spike trains and thus perform poorly. Recent metrics [2, 5, 7, 9] take these biological factors into account which results in better performance. This work shows that several of these metrics are identical or closely related.

Furthermore we show that kernel functions can be constructed that are also closely related to these metrics. Kernel methods [6] perform complex premapping of the data into a higher dimensional feature space to allow classification using simple techniques like linear discrimination. The spike train kernels thus allow the use of state-of-the-art classification techniques like Support Vector Machines (SVM) or kernel Principal Component Analysis[1] (kernel-PCA) [6] on spike trains. Previous work [3, 7] already used kernel methods on spike trains, but they require binning which results in a substantial loss of temporal information. The proposed kernels do not require binning of the spike trains and thus preserve the temporal resolution. When much of the information is stored in the exact spike times (temporal coding), this is a very important feature.

As a very simple artificial benchmark application we use a jittered spike train template classification problem. Although this application may appear simple, complex tasks like speech recognition [8] are extensions of this simple benchmark.

*This research is partially funded by FWO Flanders project G.0317.05.

[1]KPCA is used to project the input and afterwards the sign of the first PCA dimension is used to classify the data.

2 Spike train kernels

A kernel is formally an inner-product operator $K : X \times X \to \mathbb{R}$ with X an arbitrary vector space (the input space). A very high or even infinitely dimensional feature space is implicitly defined by the kernel. A kernel can also be explicitly defined by the mapping to a feature space $\phi : X \to Y$ with X the initial vector space and Y the feature space. The kernel is then defined as $K(x, z) = \phi(x) \cdot \phi(z)$. The next section will compare kernels to distance metrics. This is possible because a kernel is an inner product and a distance metric can thus be written in terms of kernels.

We will now present two kernels that operate on *sets* of spike times. Define a spike train x as the set of spike times $x = \{t_1, t_2, ..., t_N\}$. We introduce two related kernel functions on two spike trains x and z, with N and M spike times respectively. We denote the i'th spike time of x as t_i and the j'th spike time of z as t'_j. The first kernel is related to the Laplacian kernel: $K_{\text{laplace}}(x, z) = \sum_i^N \sum_j^M \exp\left(-\lambda \left|t_i - t'_j\right|\right)$, the second is related to the Gaussian kernel: $K_{\text{gauss}}(x, z) = \sum_i^N \sum_j^M \exp\left(-\lambda (t_i - t'_j)^2\right)$. All spike times of the first spike train are compared to all spike times of the second spike train and a weight decreasing proportionally to the distance between the two spikes is added. The λ parameter determines the influence of the distance between the spikes of the two spike trains. A large λ leads to a kernel relating only close-by spikes, small λ results in a metric comparing all spikes to all other spikes. It is easy to see that these kernels satisfy the Mercer condition[2].

Note that this is not the classical approach to use kernels. Kernels normally operate on vectors, while in our case the kernels operate on *sets* of spike times. All spike times are compared to all other spike times. But because an exponentially decaying weight is attributed to these comparisons, for larger values of λ these kernels can still be implemented computationally fast because we can truncate the exponentials.

3 Related spike train metrics

In this section we will present several metrics on spike trains and show that they all are related or even equal to each other and to the kernels presented in the previous section.

3.1 Spike train algebra

In [2] a vector space spanned by a base of spike trains is defined. This vector space is closely related to the kernels presented in the previous section because the kernels are actually inner products in this vector space. Let's define a vector space of 'weighted' spike trains[3]. A weighted spike train will be denoted as

[2]This is a necessary condition in order to apply the so-called 'kernel trick' which underlies all modern applications of kernels.

[3]These weights need to be defined in order to construct a vector space but will be set to one in the remainder of this work.

$[w_1, w_2, ..., w_N; t_1, t_2, ..., t_N] \equiv \sum_{i=1}^{N} w_i \delta_{t_i}$ with $\delta_{t_i}(t) \equiv \delta(t - t_i)$ the Dirac delta function and $w_i \in \mathbb{R}$. Note that weighted spike trains are still a function of time. To define a vector space we need a scalar product, a vector sum, and an inner-product. The scalar product is defined as $c[w_1, ..., w_N; t_1, ..., t_N] = [cw_1, ..., cw_N; t_1, t_2, ..., t_N]$ and the vector sum as

$$[w_1,...,w_N;t_1,...,t_N]+[w'_1,...,w'_M;t'_1,...,t'_M]=[w_1,...,w_N,w'_1,...,w'_M;t_1,...,t_N,t'_1,...,t'_M].$$

The inner product can be defined on two unit spikes (single spikes with unit weight) $\langle [1;t_1], [1;t_2] \rangle = \langle \delta_{t_1}, \delta_{t_2} \rangle = \exp(-\lambda |t_1 - t_2|)$ which can then be generalized to

$$\langle [w_1,...,w_N;t_1,...,t_N], [w'_1,...,w'_M;t'_1,...,t'_M] \rangle = \sum_{i=1}^{N} \sum_{j=1}^{M} w_i w'_j \exp(-\lambda |t_i - t'_j|)$$

with λ a scaling factor. It has been proved in [2] that this is a vector space. Note that a Gaussian version of the inner product is also possible though not considered in [2]. For spike trains with unit weight, the inner product in this vector space is equal to our spike train kernels. The distance in this vector space can be defined as: $D_\lambda^{\mathrm{VS}}(x, z) = |x - z| = \sqrt{\langle x - z, x - z \rangle}$ which gives for spike trains with unit weights (these are the ones we are interested in)

$$D_\lambda^{\mathrm{VS}}(x,z)=\sqrt{\sum_{i,j=1}^{N} \exp(-\lambda|t_i-t_j|)+\sum_{i,j=1}^{M} \exp(-\lambda|t'_i-t'_j|)-2\sum_{i=1}^{N}\sum_{j=1}^{M} \exp(-\lambda|t_i-t'_j|)}.$$

The inner product, the distance metric and the presented kernels are thus closely linked.

3.2 L^2-norm of Gaussian filtered spike trains

In [5] the distance between two spike trains was defined by the L^2-norm of the difference between the Gaussian filtered spike trains. The Gaussian filtering of spike train $s(t)$ is equal to $\sum_i \exp(-\lambda(t - t_i)^2)$. A Laplace variant is of course also possible. The L^2-norm of the difference between the filtered versions of the two spike trains is equal to: $D_\lambda^{L^2}(x, z) = |x - z| = \sqrt{\int_{-\infty}^{+\infty} |x(t) - z(t)|^2 \, \mathrm{d}t}$. This distance metric can be made equal to the distance defined for the spike train vector space of Section 3.1: $\frac{2\sqrt{\lambda}}{\sqrt{\pi}} D_{2\lambda}^{L^2}(x, z) = D_\lambda^{\mathrm{VS}}(x, z)$. The analytical derivation of this relationship is left out for brevity. We can conclude that both metrics are directly related to each other by a constant scaling factor for the distance and the time constant.

3.3 Spike train metric spaces

A frequently used class of biologically inspired spike train metrics was introduced in [9] and is based on spike train alignment. The metric is defined by the minimisation of a cost function. There are three basic operations with their related costs: creating a spike (cost 1), deleting a spike (cost 1) and moving a spike in time (cost $q|\Delta t|$ with $q > 0$ a parameter to set the cost of moving a spike). The distance between two spike trains is defined by the minimal cost to produce

the latter spike train starting from the first. This metric is called D^{spike} and emphasizes the exact location of single spikes[4]. It is calculated using dynamic programming techniques. Note that the cost of moving a spike can be generalised and that for example the Laplacian like cost function $2\left(1-\exp(q\left|\Delta t\right|)\right)$ can be used.

Initially it might not be directly clear how D^{spike} relates to the kernels defined in the previous section. This is why we conducted an experiment. As shown in Section 3.1, spike train vector spaces are directly related to the proposed spike train kernels. To be able to compare the D^{spike} metric to the vector space metric, we use $2\left(1-\exp(q\left|\Delta t\right|)\right)$ as a cost function for moving spikes. We generated 100 000 pairs of spike trains, each containing a number of spikes uniformly distributed in the interval $[0, 50]$ and spike times uniformly distributed in the interval $[0, 500]$. The distance between two spike trains of each pair are calculated using D^{spike} and $|x-z|^2$. The q parameter is set to 10. The correlation coefficient is equal to 0.9989 which indicates a near-identity relation.

The original D^{spike} metric with $q\left|\Delta t\right|$ as a cost function for moving a spike can also be compared to the vector space metric if we use the inner product $\langle [1;t_1], [1;t_2]\rangle = \langle \delta_{t_1}, \delta_{t_2}\rangle = \max(0, 1-\frac{q}{2}\left|t_1-t_2\right|)$. The same experiment now gives a correlation coefficient of 0.9992 which is even better.

We can conclude that the D^{spike} metric is very similar to the other metrics but not identical. But note that it is computationally more demanding (it takes approximately twice as long to compute).

3.4 Spikernel and the alignment score kernel

The 'spikernel' [7] (a string kernel that is applied to the instantaneous firing rate) and the alignment score kernel [3] (a kernel built by an explicit feature space mapping based on an alignment score such as the one presented in Section 3.3, but then on binned spike trains) are also kernel based methods that operate on spike trains such as the kernels presented in this work; except they operate on *binned* spike trains. With respect to our spike train kernels this introduces an extra parameter to tune (bin size) and most importantly, throws away much of the temporal resolution of the spike train.

The discussion of whether a spike train should be binned or not is closely related to the question of rate versus temporal coding used for coding inside the brain. The current view on this topic is that both are used. Some parts of the brain mainly communicate through the average firing rate of its neurons (like the motor neurons), while other parts are very sensitive to the exact temporal position of the spikes (higher brain regions). A kernel can be viewed as a similarity measure that can capture much of the domain 's prior knowledge. Our kernels preserve temporal resolution and are sensitive to the exact temporal position of the spikes. It can thus be expected that our spike train kernels will perform better on classification tasks where temporal coding is dominant.

[4] More complex metrics are defined (like D^{interval} emphasizing spike interval timing and D^{motif} emphasizing temporal patterns) but they can all be related to D^{spike}.

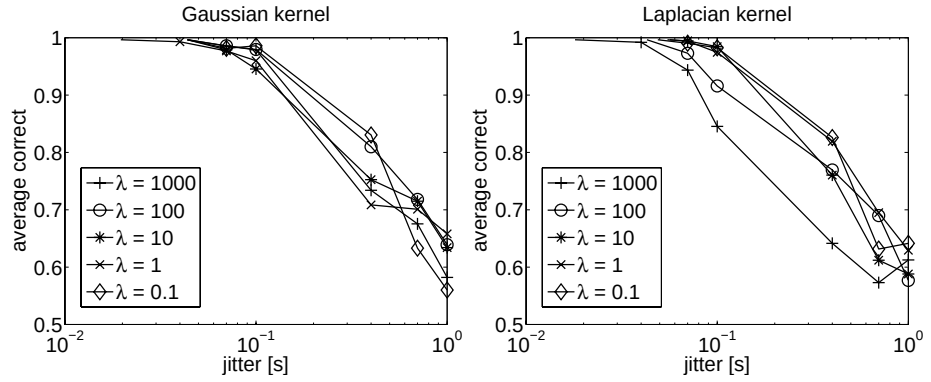

Fig. 1: Classification results using SVM with Gaussian (left) and Laplacian (right) kernels. Performance is expressed in average correct classification and it is plotted against the added jitter. The different plots in each figure are for different values of the λ parameter.

4 Experiments

As a benchmark test for validating the two spike train kernels we use the artificial jittered spike train classification benchmark (also used in [5] and [1]), and perform SVM classification on it. We generate two random 'spike train templates' with a duration of 1 s, an average Poisson firing rate of 10 Hz and a refractory time of 3 ms. From these templates, 500 spike trains for training and 200 spike trains for testing are generated, randomly drawn from the two templates; and each spike jittered with Gaussian noise with zero mean and several variance values. We also tested for various settings of the λ parameter. All results are averages of 10 independent runs.

The results can be seen in Figure 1. We trained a soft-margin SVM (which allows some misclassifications) for three values of $C = \{1, 10, 100\}$ (which is a kind of cost attributed to misclassifications) but the results were very similar. The figures show the results for $C = 10$ which was the best performing setting. The figures clearly show that we get very good classification results even with high levels of jitter. Compared to the results in [5] and [1] we get the same levels of recognition with almost an order of magnitude more jitter. For example the jitter level in [1] (equivalent to a jitter of 66 ms with our setup) attained a correct classification of 0.89. We reach a correct recognition of 0.89 at about 200 ms jitter which is three times more jitter.

Note that for comparing the results from the kernels we should compare λ values for both kernels that result in similar 'widths'. These widths are approximately equal if $\lambda_{\text{gauss}} \approx \frac{\lambda^2_{\text{laplace}}}{2}$. Taking this into account, the kernels perform very similarly. Smaller values of λ tend to perform better. This is expected because small λ values result in broad spheres of influence around each spike

which gives a more global scope to the comparisons, but will lead to more computations. A biologically plausible value for λ would be 100 for the Laplacian case (being equivalent to a membrane time constant of 10 ms).

We also tested kernel-PCA on this benchmark. The first principle component was used to classify the data and we got a 0.89 classification score for jitter of up to 100ms which is only slightly worse than in the SVM case.

5 Conclusions and future work

In this publication we presented two kernels that can be applied to spike times, allowing kernel methods like SVM or kernel-PCA to be applied to the classification of spike trains, but without binning the spike trains and a loss of temporal information. We have shown that the presented kernels are related to several existing metrics for spike trains (and that several of these metrics are themselves related). The performance of the presented kernels was demonstrated on a jittered spike train template classification benchmark. Both SVM and kernel-PCA give very good recognition rates, even with high levels of jitter. No clear evidence is present that shows that one of both kernels outperforms the other.

As future work we plan to use this technique on a physiological dataset to evaluate its performance on real life data [7]. Another more practical application is to demonstrate its use as a Liquid State Machine post-processing spike train classifier for solving for example a speech recognition problem [8].

References

[1] O. Booij and H. T. Nguyen. A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters*, 95(6):552–558, 2005.

[2] A. Carnell and D. Richardson. Linear algebra for time series of spikes. In M. Verleysen, editor, *Proc. of ESANN*, pages 363–368, 2005.

[3] J. Eichhorn, A. Tolias, A. Zien, M. Kuss, C. E. Rasmussen, J. Weston, N. Logothetis, and B. Schölkopf. Prediction on spike data using kernel algorithms. In *Proc. of NIPS*, pages 1367–1374, 2004.

[4] W. Maass and C. Bishop. *Pulsed Neural Networks*. Bradford Books/MIT Press, Cambridge, MA, 2001.

[5] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Computation*, 14(11):2531–2560, 2002.

[6] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, 2002.

[7] L. Shpigelman, Y. Singer, R. Paz, and E. Vaadia. Spikernels: Predicting arm movements by embedding population spike rate patterns in inner-product spaces. *Neural Computation*, 17(3):671–690, 2005.

[8] D. Verstraeten, B. Schrauwen, D. Stroobandt, and J. Van Campenhout. Isolated word recognition with the liquid state machine: a case study. *Information Processing Letters*, 95(6):521–528, 2005.

[9] J. D. Victor and K. P. Purpura. Metric-space analysis of spike trains: theory, algorithms and application. *Network: Comput. Neural Syst.*, 8:127–164, 1997.