Classification of One-Dimensional Non-Stationary Signals Using the Wigner-Ville Distribution in Convolutional Neural Networks

Johan Brynolfsson
Mathematical Statistics,
Centre for Mathematical Sciences,
Lund University, Sweden
Email: johanb@maths.lth.se

Maria Sandsten
Mathematical Statistics,
Centre for Mathematical Sciences,
Lund University, Sweden
Email: sandsten@maths.lth.se

***Abstract*—In this paper we argue that the Wigner-Ville distribution (WVD), instead of the spectrogram, should be used as basic input into convolutional neural network (CNN) based classification schemes. The WVD has superior resolution and localization as compared to other time-frequency representations. We present a method where a large-size kernel may be learned from the data, to enhance features important for classification. We back up our claims with theory, as well as application on simulated examples and show superior performance as compared to the commonly used spectrogram.**

I. Introduction

Convolutional neural networks (CNN) is rapidly growing as a tool for classification of a wide range of signals. By optimizing convolutional kernels in multiple layers, where the results from the kernels in one layer is passed to the next, extremely accurate classifiers can be generated. Instead of manually identifying features that separate different classes, features are instead learned from pre-classified data, as interpreted through the convolutional kernels [1]. Although initially introduced to be applied to image sets, the application of CNNs has spread to also be applied to the time-frequency representations of audio, EEG, or other one-dimensional non-stationary signals. The addition of temporal frequency information has been shown to generate more robust results in classification; however most papers simply calculate the spectrogram of the audio signal [2]–[4]. There are many other ways to estimate the time-frequency content of a signal, such as methods based on the evolutionary spectrum, time-scale representations or smoothing kernels applied to the Wigner-Ville distribution (WVD) [5]. The WVD is unique and unbeaten in terms of locality, for signals containing only one single component, as well as being information loss-less. From the WVD, all possible quadratic time-frequency representations can be generated using a two-dimensional smoothing kernel of the WVD. Where as the spectrogram usually is calculated using some chosen data window function (e.g. Hann or Hamming windows), and the short-time Fourier transform, it could also be computed using a smoothing kernel of the WVD. The reluctancy of using the WVD is often attributed to the cross-terms arising between every pair of components in the signal, making it visually difficult to interpret. Such cross-terms are smoothed using the spectrogram or other general kernels. It has however been argued that these cross-terms in fact hold valuable classification information [6], and hence should not be excluded. Especially when the difference between the classes are small, such that high resolution and accurate locality is needed to separate them.

There are papers that have used WVD-based methods as input to a classification CNN, [7], [8], it is however often done without proper justification or discussion on the choice of inputs, and without any comparisons to other forms of inputs. In this submission we argue that the WVD should be used as input to a classification CNN. We form an extra layer where a large size convolutional kernel is learned, to allow for any type of smoothing. Throughout the paper, $*$ will denote convolution, i the imaginary unit, t and n denotes continuous and discrete time, respectively, f and m continuous and discrete frequency, and τ denotes time-lag. Continuous functions will be denoted $y(t)$, and the discrete observations of the same function will be denoted $y[n]$. $N(\mu, \sigma)$ denotes the Gaussian distribution. All integrals are assumed to range between $-\infty$ to ∞.

II. Time-Frequency Representation

The WVD is defined, given the auto-correlation function $r_z(t,\tau) = z(t+\frac{\tau}{2})z^*(t-\frac{\tau}{2})$, as

$$W_z(t,f) = \int r_z(t,\tau)e^{-i2\pi f\tau}d\tau, \qquad (0)$$

where z is an analytic signal, i.e. all frequencies are non-negative and z^* denotes the complex conjugate. If the signal is not analytic, this is achieved by the Hilbert transform,

$$z(t) \triangleq \mathcal{H}[x(t)] = \mathcal{F}^{-1}_{f\to t}((-i\cdot \text{sign}(f))\mathcal{F}_{t\to f}x(t)), \qquad (0)$$

where $\mathcal{F}_{a\to b}$ is the Fourier transform from a to b, and $\mathcal{F}^{-1}_{b\to a}$ is the inverse Fourier transform from b to a. The WVD

This work was supported by the Swedish strategic research programme eSSENCE.

ISBN 978-0-9928626-7-1 © EURASIP 2017

does however suffer from cross-terms, especially if there are multiple components in the signal. Between every component in the signal an oscillating cross-term will appear in the spectrum. It can be shown that

$$W_{z_1+z_2}(t,f) = W_{z_1}(t,f) + W_{z_2}(t,f) + 2\Re\left\{W_{z_1 z_2}(t,f)\right\} \tag{0}$$

where $W_{z_1}(t,f)$ and $W_{z_2}(t,f)$ are denoted auto-terms and $2\Re\left\{W_{z_1 z_2}(t,f)\right\}$ is the oscillating cross-term centered between $W_{z_1}(t,f)$ and $W_{z_2}(t,f)$, [5]. The cross-terms may be remedied by convolution with some kernel function $\rho(t,\tau)$, giving us what is called the quadratic class

$$W_z^Q(t,f) = \iint \rho(t-s,\tau) r_z(s,\tau) e^{-i2\pi f\tau} ds\, d\tau \tag{0}$$

It is easily shown that the spectrogram, calculated with some window function $h(t)$, is a member of the quadratic class (also referred in literature as the bilinear class or Cohen's class). The spectrogram is defined

$$\begin{aligned} S_z(t,f) &= \left| \int h^*(t-s_1) z(s_1) e^{i2\pi f s_1} ds_1 \right|^2 \\ &= \left(\int h^*(t-s_1) z(s_1) e^{i2\pi f s_1} ds_1 \right) \times \\ &\quad \left(\int h(t-s_2) z^*(s_2) e^{-i2\pi f s_2} ds_2 \right) \end{aligned}$$

which, with use of the change of variables $s_1 = s + \frac{\tau}{2}$ and $s_2 = s - \frac{\tau}{2}$, gives

$$\begin{aligned} S_z(t,f) &= \iint h(t-s-\frac{\tau}{2}) h^*(t-s+\frac{\tau}{2}) \times \\ &\quad z(s+\frac{\tau}{2}) z^*(s-\frac{\tau}{2}) e^{i2\pi f\tau} ds\, d\tau \\ &= \iint \rho^h(t-s,\tau) r_z(s,\tau) e^{i2\pi f\tau} ds\, d\tau \\ &= \left(Q^h ** W_z\right)(t,f), \end{aligned}$$

where $\rho^h(t,\tau) = h(t+\frac{\tau}{2}) h^*(t-\frac{\tau}{2})$ and the time-frequency smoothing kernel $Q^h(t,f) \triangleq \mathcal{F}_{\tau\to f}\rho^h(t,\tau)$, [5].
This means that the set of spectrograms is a subset of the quadratic class, and hence a restriction on the possible time-frequency representations of the signal. When constructing the spectrogram there is trade-off in resolution between time and frequency. If a long time-frame is used to estimate the temporal frequency, good resolution in frequency is achieved but poor resolution in time. Vice-versa is true if a short time-frame is chosen. The WVD however has perfect temporal and spectral resolution simultaneously [5].

III. Neural Network Kernel Optimization

We propose that when CNNs are used to classify one-dimensional signals, e.g audio, the WVD should be used as input and the first layer in the CNN could be constructed to optimise some large size convolution kernel. A drawback of optimising the kernel applied to the WVD is the size of kernel function. If the window length is used to calculate the spectrogram is set to $N = 32$, then the corresponding smoothing kernel will be at least $N^2 = 1024$ (depending on the choice of zero-padding), and hence introduce a huge set of parameters to optimise, for a single kernel. The same problem is encountered in [9] when constructing a CNN for image noise deconvolution. The proposed solution to the problem is calculation of the singular value decomposition (svd) of the kernel. The spectrogram kernel $Q^h[n,m]$, is highly structured with a dependency between the time and frequency dimensions, where a large window function in time generates a narrow main lobe in frequency and vice-verse. We want to alleviate this restriction to possibly move away from the spectrogram to some other member of the quadratic class, opening for more information to be passed to the CNN. Denoting the svd of any kernel as $Q = USV^T$, and the j:th column in U and V as $\mathbf{u}_j$ and $\mathbf{v}_j$ respectively and the j:th diagonal element in S as s_j, the convolution of the WVD $W_z[n,m]$ with any kernel $Q[n,m]$ may be approximated using the M first singular components,

$$\left(Q ** W_z\right)[n,m] \approx \sum_{j=1}^{M} s_j \cdot \mathbf{u}_j[n] * \left(\mathbf{v}_j[m]^T * W_z[n,m]\right), \tag{-5}$$

where equality is achieved if $M \geq \text{rank}(Q)$.
By optimizing this window function we open the door to multiple sub-classes in the quadratic class, including the spectrogram, but without the performance restrictions induced by the spectrogram, or any other member of the quadratic class, with a set of data windows.
The proposed layer structure may be seen, in figure 1, under kernel-tuning. Note that this section only contains pure convolution. No bias is added, nor is any activation function or pooling applied. Note also that it is a one-to-one mapping layer, i.e. one image in, one image (of same size) out, meaning the kernel will only allow for smoothing out noise or enhancing existing information that may be interpreted by the convolutional feature extracting part of the neural network. One could easily structure the first layer to create multiple output images, where different kernel-functions are generated. In this paper we use only one to keep the comparison with the spectrogram fair.

IV. Simulated Examples

To show the advantages of using the WVD instead of the spectrogram we created eight different classes and simulated a total of 2000 observations to be used for training. This training set was further split into 1800 training samples and 200 validation samples. The validation samples were used to analyze over-fitting during training. An additional 2000 observations were simulated to be used for evaluation. For each realization the WVD and the spectrogram is calculated, assuring that both sets contain the same realizations of the classes. The spectrogram is calculated using a Hann window of length 32 [5]. The spectrogram and WVD are separately normalized to be zero-mean and unit-variance, to increase efficiency [12].

ISBN 978-0-9928626-7-1 © EURASIP 2017

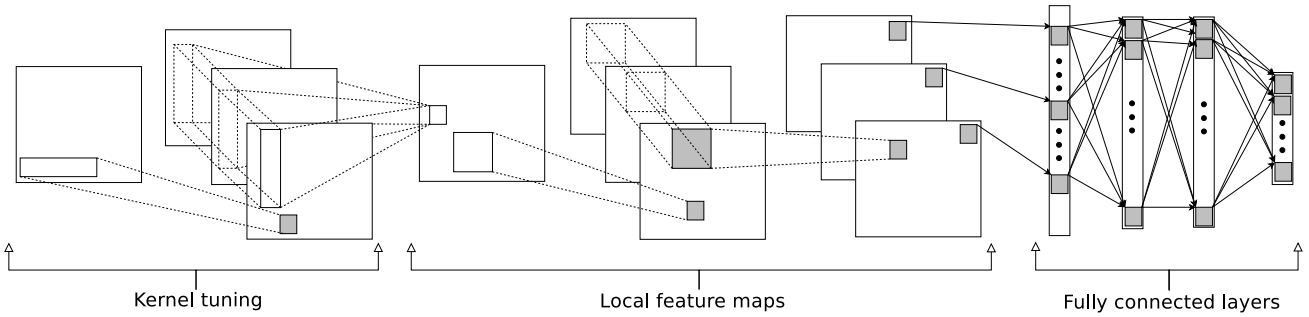

Figure 1: The proposed neural network structure, divided into three parts. The first part consists of a smoothing, where the vectorized components of the smoothing kernel is optimised. The second part consists of two standard setup convolutional layers, both including a ReLu-layer [10] and a 2x2 max-pooling, [11], The third part consists of three fully connected layers, and finally a softmax operator is applied to the last layer to perform classification.

Layer	# of maps & neurons	kernel
input	1 map of 128x64	
kernel1	$1 \rightarrow 8$ maps of 128x64	32x1
kernel2	$8 \rightarrow 1$ map of 128x64	1x32
conv1	$16 \rightarrow 16$ maps of 128x64	5x5
maxpool1	$16 \rightarrow 16$ maps of 64x32	
conv2	$16 \rightarrow 16$ maps of 64x32	5x5
maxpool2	$16 \rightarrow 16$ maps of 32x16	
fullyconn1	32	
fullyconn2	32	
softmax	8	

Table I: Neural network parameters

A. Data

The classes are created to be pairwise similar, with small differences making them inherently difficult to separate from each other. The first two classes are linear chirps, defined with a small difference in chirp rate. Class 3 and 4 both consist of two sinusoids with the same parameters but where class 3 has a third sinusoid in between the first two, with a much smaller amplitude. Class 5 consists of a single sinusoid, where class 6 includes two sinusoids with a very small frequency difference. Class 7 is a Gaussian component and class 8 is the same Gaussian component with another Gaussian component situated close the first one, but with smaller amplitude. The signals are built using the following components:

Sinusoids:

$$y_k[n] = a_k \exp\left(i2\pi f_k n + \phi\right) \qquad \text{(-5)}$$

Linear chirps:

$$y_k[n] = \exp\left(i2\pi\left(f_k n + \frac{df_k}{2}n^2\right) + \phi\right) \qquad \text{(-5)}$$

Gaussian:

$$y_k[n] = a_k \exp\left(-\frac{(n-t_k)^2}{2b_k^2}\right)\exp\left(i2\pi f_k n + \phi\right) \qquad \text{(-5)}$$

For all components, the phase is randomized as uniformly distributed $\phi \in U(0, 2\pi)$.
The signals of each simulated class c, is defined

$$y^c[n] = \sum_{k=1}^{K} y_k[n] + \epsilon_\alpha[n], \quad n = 0, \ldots, 128 \qquad \text{(-5)}$$

where $\epsilon_\alpha[n] \in N(0, \alpha)$, is an uncorrelated noise sequence. The respective parameters of the classes are presented in table II, where each ϵ_σ denotes a new random observation of $N(0, \sigma)$. Note that randomness is assigned to all parameters in each class, additional to the random noise added to each realization.

B. Neural network structure

The neural network structure used to classify the simulated signals is presented in figure 1. The sizes and number of kernels are presented in table I. We have here chosen to set the kernel size of the first large-size kernel to $N = 32$, making the comparison with the spectrogram fair. Further, the number of vector components used was set to $M = 8$, which was empirically found. Note that no orthogonality is imposed on the vectors. Even though they are then not true singular vectors, they still add upp to a valid kernel. Note also that the convolutional and fully connected layers are followed by a ReLu function as activation function [10], unlike the kernel tuning section which does not.
This neural network structure will henceforth be denoted as "kernel tuning Convolutional Neural Network" or ktCNN. The relatively few layers and parameters, and the small number of simulations used to train the network, as compared to current state-of-the-art CNNs are motivated to show the performance gains of using WVD for this quite simple toy example. The main goal here is to highlight the performance difference depending on the input.

ISBN 978-0-9928626-7-1 © EURASIP 2017

Class	Components	Parameters
1	1 linear chirp	$f_1 = 0.2 + \epsilon_{0.01}$ $df_1 = 0.001 + \epsilon_{0.001}$
2	1 linear chirp	$f_1 = 0.2 + \epsilon_{0.01}$ $df_1 = 0.0012 + \epsilon_{0.001}$
3	3 sinusoids	$a_1 = 3 + \epsilon_{0.005}$ $f_1 = 0.06 + \epsilon_{0.005}$ $a_2 = 0.5 + \epsilon_{0.005}$ $f_2 = 2f_1$ $a_3 = 2 + \epsilon_{0.05}$ $f_3 = 3f_1$
4	2 sinusoids	$a_1 = 3 + \epsilon_{0.005}$ $f_1 = 0.06 + \epsilon_{0.005}$ $a_2 = 2 + \epsilon_{0.05}$ $f_2 = 3f_1$
5	2 sinusoids	$a_1 = a_2 = 2 + \epsilon_{0.01}$ $f_1 = 0.05 + \epsilon_{0.01}$ $f_2 = 0.051 + \epsilon_{0.01}$
6	1 sinusoid	$a_1 = 4 + \epsilon_{0.01}$ $f_1 = 0.0505 + \epsilon_{0.01}$
7	1 Gaussian	$a_1 = 3 + \epsilon_{0.005}$ $f_1 = 0.15 + \epsilon_{0.005}$ $b_1 = 12 + \epsilon_{1.0}$ $t_1 = 77 + \epsilon_{3.0}$
8	2 Gaussians	$a_1 = 3 + \epsilon_{0.005}$ $f_1 = 0.15 + \epsilon_{0.005}$ $b_1 = 12 + \epsilon_{1.0}$ $t_1 = 77 + \epsilon_{3.0}$ $a_2 = 0.2 + \epsilon_{0.005}$ $f_2 = f_1$ $b_2 = 7 + \epsilon_{1.0}$ $t_2 = 27 + \epsilon_{3.0}$

Table II: Table of the 8 different classes. Each ϵ_σ is a new independent observation of $N(0, \sigma)$ for each parameter and signal realization. The parameters are those defined in eq. (IV-A)-(IV-A)

We optimize this using WVD (denoted WVD-ktCNN) and spectrogram (denoted spect-ktCNN), separately.

Further we train a CNN where the first kernel tuning section is removed, i.e. the WVD and spectrogram are used as input directly to the convolutional section. All other settings, i.e. number of maps and neurons, will be identical to those of ktCNN. These will be denoted WVD-CNN and spect-CNN, respectively. This is done to show that kernel tuning is beneficial for the WVD, while not increasing the performance of spectrogram.

The neural network was implemented, and optimized, using tensorflow [13]. Optimization was performed using the Adam method, minimizing the cross-entropy [14]. The minimization was performed with a learning rate of 10^{-4}, first and second momentum decay rates were set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and the zero-division avoiding smoothing term was set to $\epsilon = 10^{-8}$. A mini-batch size of 200 observations was used.

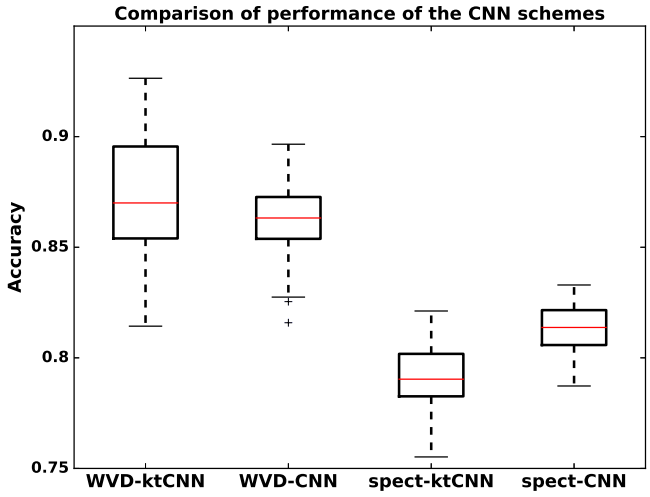

Figure 2: Box plot showing the results of the 50 different optimizations.

C. Results

Each neural network structure was re-optimized 50 times, with the weights in the neural network initialized randomly each time, to show that the results are consistent. The neural networks were then used to classify the additional 2000 class realizations in the test set. The results are presented in figure 2, where a box-plot depicts the accuracy of the 50 versions of each network structure, as evaluated on the test-set. The two schemes with WVD as input performs significantly better than the two with spectrogram as input. The resulting confusion matrix for the best WVD network (WVD-ktCNN) is presented in table III and the confusion matrix for the best performing spectrogram classifier (spect-CNN) is found in table IV. Comparing the two confusion matrices, the WVD input performs better on 7 out of 8 classes, with class 4 and 5 being the only exceptions. The largest difference is seen in the 7:th and 8:th classes. The second smaller Gaussian component in class 8 is heavily smoothed by the spectrogram, making it difficult to detect. A large difference in accuracy may also be noted between classes 1 and 2, where the perfect locality of the WVD makes a difference as compared to the spectrogram. Note that the spect-ktCNN actually performs worse than spect-CNN, meaning that no extra information may be enhanced, and hence the kernel tuning is only inserting additional randomness, making the network structure perform worse on average with kernel tuning than without.

V. Conclusions

We present evidence, both theoretically and practical, that the time-frequency representation WVD should be used as input, as opposed to an ad-hoc chosen spectrogram, when classifying one-dimensional signals using CNN. The WVD has higher temporal and frequency localization making it superior for separation of similar classes. A toy example was created to

ISBN 978-0-9928626-7-1 © EURASIP 2017

True class \ Predicted class	1	2	3	4	5	6	7	8
1	96	4	-	-	-	-	-	-
2	7	93	-	-	-	-	-	-
3	-	-	100	-	-	-	-	-
4	-	-	6	94	-	-	-	-
5	-	-	-	-	86	14	-	-
6	-	-	-	-	1	99	-	-
7	-	-	-	-	-	-	74	26
8	-	-	-	-	-	-	2	98

Table III: Confusion matrix for WVD-ktCNN rounded to nearest integer percent.

True class \ Predicted class	1	2	3	4	5	6	7	8
1	89	10	-	-	-	-	-	-
2	20	80	-	-	-	-	-	-
3	-	-	99	1	-	-	-	-
4	-	-	3	97	-	-	-	-
5	-	-	-	-	88	12	-	-
6	-	-	-	-	6	94	-	-
7	-	-	-	-	-	-	49	51
8	-	-	-	-	-	-	28	71

Table IV: Confusion matrix for spect-CNN rounded to nearest integer percent.

highlight the performance differences. Further investigations may be done, inserting some restrictions on the kernel tuning section to make it more stable.

REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[2] H. Lee, Y. Largman, P. Pham, and A. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1096–1104, 2009.

[3] M. Espi, M. Fujimoto, K. Kinoshita, and T. Nakatani, "Exploiting spectro-temporal locality in deep learning acoustic event detection," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 26, 2015.

[4] Y. Costa, L. Oliveira, and C. S. Jr., "An evaluation of convolutional neural networks for music classification using spectrograms," *Applied soft computing*, vol. 52, pp. 28–38, 2017.

[5] L. Cohen, *Time-Frequency Analysis*. Prentice-Hall, 1995.

[6] B. Gillespie and L. Atlas, "Optimizing time-frequency kernels for classification," *IEEE Transactions on Signal Processing*, vol. 49, no. 3, pp. 485–496, 2001.

[7] S. Dash and G. Rao, "Arrhythmia detection using Wigner-Ville distribution based neural network," *Procedia Computer Science*, vol. 85, pp. 806–811, 2016.

[8] A. Mjahad, A. Rosado-Munoz, M. Bataller-Mompeán, J. Francés-Víllora, and J. Guerrero-Martínez, "Ventricular fibrillation and tachycardia detection from surface ecg using time-frequency representation images as input dataset for machine learning," *Computer Methods and Programs in Biomedicine*, pp. 119–127, 2017.

[9] L. Xu, J. Ren, C. Liu, and J. Jia, "Deep convolution neural network for image deconvolution," *Advances in Neural Information Processing Systems*, pp. 1790–1798, 2014.

[10] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *The 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[11] D. Scherer, A. Müller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *20th International Conference on Artificial Neural Networks (ICANN)*, (Thessaloniki, Greece), 2010.

[12] T. Jayalakshmi and A. Santhakumaran, "Statistical normalization and back propagation for classification," *International Journal of Computer Theory and Engineering*, vol. 3, no. 1, pp. 89–93, 2011.

[13] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[14] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, 2015.

ISBN 978-0-9928626-7-1 © EURASIP 2017