On a time-frequency blurring operator with applications in data augmentation

NTNU
Norwegian University of Science and Technology

Simon Halvdansson

Introduction

If we want to spread out a signal in time and frequency, a reasonable approach is to convolve the STFT of the signal with some kernel and then synthesize a new signal:

$$B_\mu^\varphi \psi = V_\varphi^*(\mu * V_\varphi \psi) = \int_{\mathbb{R}^{2d}} \mu * V_\varphi \psi(z) \pi(z) \varphi \, dz.$$

Famously, time-frequency localization operators work the same way with a multiplication instead of a convolution. In this work, we study the analytical properties of this *time-frequency blurring operator* and its possible utility as a tool for data augmentation.

THEORY:

Motivation

The action of modifying the phase-space representation of a function and then synthesizing a signal back is well-known in time-frequency analysis, this is essentially what a localization operator $A_m^\varphi : \psi \mapsto V_\varphi^*(m \cdot V_\varphi \psi)$ does. Apart from multiplication, convolution is a standard way to combine two functions which has a lot of structure so we choose to put this operation in between the analysis and synthesis. Investigating the analytic properties of this operator, as one does for localization operators, is interesting and we make the case that the operator also has value from an applied point of view also (see the right column).

Example

Original: **Modified:**

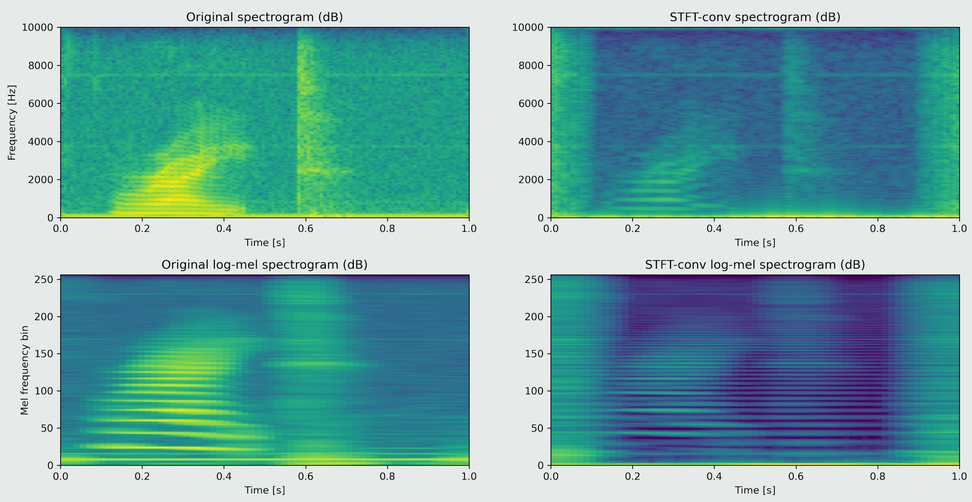

Figure 1. Spectrograms in the "modified" column have had a discrete time-frequency blurring operator with Gaussian kernel applied to them.

Deep filtering/position-dependent kernel

Several traditional machine learning noise reduction techniques have used the pipeline

$$\psi_{\text{noisy}} \xrightarrow{\text{STFT}} V_\varphi \psi_{\text{noisy}} \xrightarrow{\text{Neural network}} m \xrightarrow{\text{Apply}} A_m^\varphi \psi_{\text{noisy}}$$

with the neural network trained with a loss functions like

$$\big\| |V_\varphi A_m^\varphi \psi_{\text{noisy}}| - |V_\varphi \psi_{\text{clean}}| \big\|.$$

Recently, this approach was generalized under the name Deep Filtering where the network instead learned a filter for each time-frequency bin. The continuous version of this can be written as the convolution

$$F(z) = (\mu_z * V_\varphi \psi)(z)$$

where the kernel $\mu_z \in L^1(\mathbb{R}^{2d})$ depends on z. Consequently, the full version of their filtering can be written as

$$B_{\boldsymbol{\mu}}^\varphi \psi = \int_{\mathbb{R}^{2d}} \mu_z * V_\varphi \psi(z) \pi(z) \varphi \, dz$$

where we use a bold $\boldsymbol{\mu}$ to indicate that it is a function on double phase space $\boldsymbol{\mu}(z, w) = \mu_z(w)$. In the Deep Filtering papers, the neural network learns the mapping $V_\varphi \psi_{\text{noisy}} \mapsto \boldsymbol{\mu}$.

Analytical properties

Boundedness:
We can prove boundedness of the operator between a few standard spaces, mainly using Young's inequality and standard results from time-frequency analysis:

- $\|B_\mu^\varphi \psi\|_{L^2} \leq \|\mu\|_{L^1} \|\varphi\|_{L^2}^2 \|\psi\|_{L^2}$,
- $\|B_\mu^\varphi \psi\|_{L^\infty} \lesssim \|\mu\|_{L^1} \|\varphi\|_{L^\infty} \|\psi\|_{M^1}$,
- $\|B_\mu^\varphi \psi\|_{M^r} \lesssim \|\mu\|_{L^p} \|\psi\|_{M^q}, \qquad \frac{1}{p} + \frac{1}{q} = 1 + \frac{1}{r}$,
- $\|B_\mu^\varphi \psi\|_{L^p} \lesssim \|\mu\|_{L^1} \|\psi\|_{M^p}, \qquad 1 \leq p \leq 2$,
- $\mu \in \mathcal{S}(\mathbb{R}^{2d})$, $\varphi \in \mathcal{S}(\mathbb{R}^d)$, $\psi \in \mathcal{S}(\mathbb{R}^d) \implies B_\mu^\varphi \psi \in \mathcal{S}(\mathbb{R}^d)$.

Non-compactness:
If B_μ^φ is not the zero operator, it is non-compact. This also means that a convolution operator on the Gabor space $V_\varphi(L^2)$ is non-compact

Positivity:
If $\hat{\mu} \geq 0$, B_μ^φ is a positive operator, however the implication does not go both ways.

Zeroness:
There exists non-zero μ, φ such that B_μ^φ is the zero operator. This is achieved by choosing μ, φ such that supports are disjoint in the Fourier domain.

APPLICATION:

Data augmentation for machine learning

If we have a limited data set and want to train a machine learning system on the data, we often employ data augmentation to create augmented versions of our data to give the system more examples to learn from.
Many systems that work on signals use the spectrogram $|V_\varphi \psi|^2$ as input and then use computer vision systems to e.g. classify a signal, transcribe audio, output a mask for noise reduction or source separation. Standard methods to augment signals include:

- Adding white noise
- Mixup: Linearly combine two signals $\alpha\psi + (1-\alpha)\phi$ and modify the label accordingly
- CutMix: Cut up and mix two signals $\chi_\Omega \psi + (1 - \chi_\Omega)\phi$ and modify the label accordingly
- SpecAugment: Probably most used method, a form of structured dropout which randomly drops frequency bands and mutes audio for selected intervals
- Room Impulse Response: Convolve waveform with a predefined impulse response to simulate the effect of being in an echoey room

We propose the blurring operator B_μ^φ as an additional technique.

Spectrogram blurring

We can achieve a similar type of blurring by ignoring the phase component and just blurring the spectrogram. Generally, the phase of the noisier parts of a signal has less structure meaning that they are suppressed when taking an average which is not the case for spectrogram blurring. We include this simpler (albeit less amenable to time-frequency analysis methods) operation in our comparison below.

Experimental setup

We choose a simple task to test the effectiveness of the augmentation. 1 second audio recordings are to be classified as one of 35 classes and we limit the amount of training data to 100/300/600/1000 examples per class. We use two standard architectures for this; one convolutional neural network (CNN) and one vision transformer (ViT), which both take a log-mel spectrogram as input as this is a standard way to solve this problem. Networks are trained many times to average out the randomness in the test accuracy. We repeat the training procedure for different augmentation setups and number of training examples thousands of times in total, contributing non-trivially to my home electricity bill.

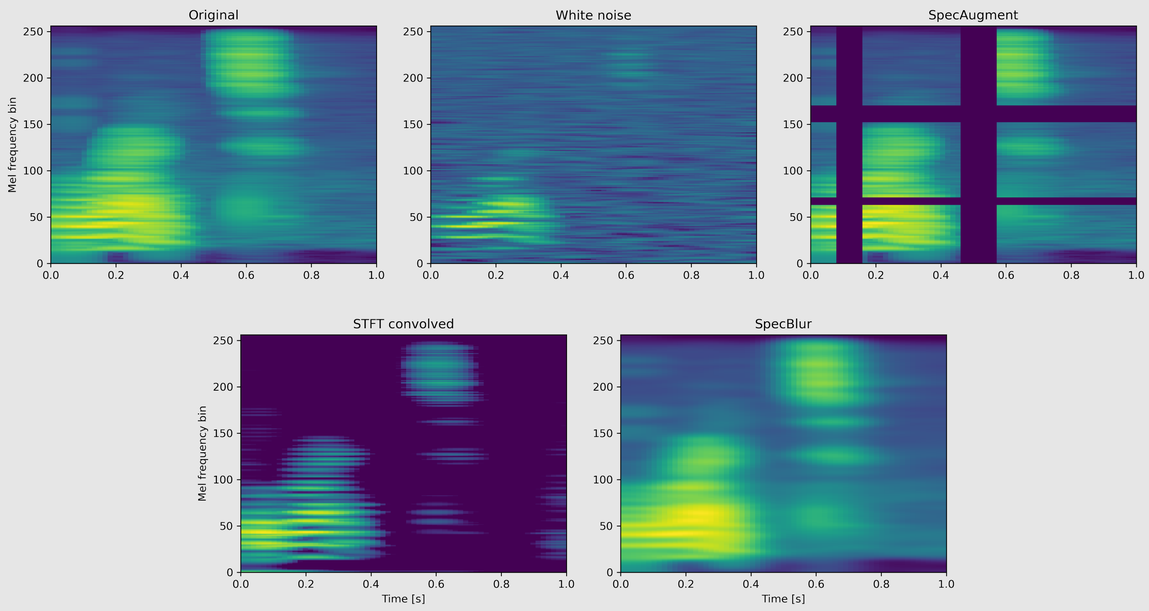

Figure 2. Log-mel spectrograms of an audio recording from the SpeechCommands V2 dataset with different augmentation techniques applied to it.

Results (vision transformer)

Table 1. Average accuracy on the test set along with standard errors.

Augmentation	Acc-100	Acc-300	Acc-600	Acc-1000
None	25.85 ± 0.29	71.15 ± 0.46	84.32 ± 0.23	89.17 ± 0.20
White noise	41.64 ± 0.32	80.84 ± 0.22	87.94 ± 0.15	90.72 ± 0.09
SpecAugment	46.97 ± 0.33	81.26 ± 0.22	87.55 ± 0.08	90.61 ± 0.14
STFT-blur	50.46 ± 0.28	81.00 ± 0.24	87.56 ± 0.18	90.40 ± 0.15
SpecBlur	52.67 ± 0.30	84.08 ± 0.14	89.00 ± 0.12	91.29 ± 0.13
White noise + SpecAug	56.61 ± 0.33	84.46 ± 0.15	89.61 ± 0.15	91.80 ± 0.15
STFT-blur + SpecBlur	67.54 ± 0.29	85.65 ± 0.16	89.22 ± 0.17	91.72 ± 0.12
All	73.38 ± 0.19	86.89 ± 0.14	90.60 ± 0.13	92.70 ± 0.08

References

Halvdansson, S. (2024). *On a time-frequency blurring operator with applications in data augmentation*. arXiv: 2405.12899 [math.FA].