Predicting Transformed Audio Descriptors: A System Design and Evaluation

Graham Coleman
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain
graham.coleman@upf.edu

Fernando Villavicencio
Music Technology Group
Universitat Pompeu Fabra
Barcelona, Spain
fernando.villavicencio@upf.edu

ABSTRACT

We propose and present an example system design for predicting *changes* in perceptually relevant audio properties under the effects of common musical and sonic transformations. By building these predictive models, we may facilitate descriptor-driven control of effects while avoiding queries to the transformation itself. In this study we model spectral descriptors of a limited class of sounds under the resampling transformation with Support Vector Regression (SVR) and report on the accuracy of the predictions, with an emphasis on performance as a function of model parameters. On a test set of resampled inputs, the statistical model predicts an output filter bank within 3-4 times the mean absolute error of a comparable analytical model.

Categories and Subject Descriptors

I.5 [**Pattern Recognition**]: Applications— *signal processing*; I.5 [**Pattern Recognition**]: Models—*deterministic, statistical, SVM, SVR*

General Terms

Design, Experimentation, Performance

Keywords

descriptor-driven transformation, digital audio effects, spectral models, support vector machines

1. INTRODUCTION

Descriptor-driven control is a strategy that can be applied to sound synthesis as well as to transformations / sound effects, wherein *control parameters* for the process are selected that steer the output towards desired *target descriptors*.

Several works [7, 13] in control of parametric synthesis by descriptors have focused on the optimization of parameters, using a trial and error approach. This involves an audio synthesis/transformation followed by analysis of the result directly in the search loop, which may require considerable computational resources.

By modeling relationships between parameters and descriptors, the need to synthesize/transform candidate parameters can be avoided. A type of model that we refer to as "predictive" predicts output descriptors given by certain parameters. Work described in [2] uses evolutionary computing to find a path of spectral envelopes that is smooth in terms of spectral moments, and can be seen as employing predictive models. A second type of "selective" model maps target descriptors directly to control parameters. For example in [6] target descriptors fundamental frequency and loudness are mapped to control parameters of an additive synthesizer using SVR machines. This work deals with predictive rather than selective models.

As shown in [4], given a predictive model of a transformation, numerical programming techniques can be used to find acceptable parameters for descriptor based control. If an analytical predictive model can be derived for our transformation, then great. In the absence of a derived model, a model can be learned statistically from examples, which will be the focus of this work.

We propose therefore to predict output descriptors in audio transformations as a function of their input descriptors and control parameters, or $f(\vec{d_{in}}, \vec{p}) = \vec{d_{tr}}$. In this work we present a study modeling resampling as a statistical black box from input and output descriptors and the transformation parameter. We will generate a database of test signals with certain characteristics (noisy, stationary, can be accurately described and predicted by broad spectral features) and model them with Support Vector Regression, a statistical technique, then measure the accuracy of the predictions against a test set.

This article is structured as follows; in Section 2, we will present the objects being modeled and the modeling tools. In Section 3, we describe our specific experimental procedure. In Section 4, we will report and discuss the results, examining the effect of model parameters upon the accuracy of the predictions. In Section 5, we will conclude.

2. SVR PREDICTION OF DESCRIPTORS

2.1 Average Magnitude of Bands (AMB)

As descriptors, we use a rough measure of the spectrum obtained by averaging the magnitude values from the set of DFT bins K_j that fall into a particular filter band j, where $\sharp K_j$ is the number of bins spanned by band j.

$$\text{AvgMagBand}(j) = \frac{1}{\sharp K_j} \sum_{k \in K_j} |X_k| \qquad (1)$$

for frequency bands $j = 1 \ldots B$. These descriptors are linear with respect to gain and summation of the input signal[1] and

[1] as such, facilitate the analytic model of mean descriptors

MML'10, October 25, 2010, Firenze, Italy.

should be adequate for describing noisy signals. We report them in units of nAMB, normalized by the maximum band value (7.30×10^{-3} AMB) in the input database.

2.2 Resampling Transformation

Resampling is a frequently used audio transformation in which the signal speed (adjusted by a factor L^2, relative to a fixed sampling rate) and frequency scale (subsequently pitch) are adjusted by interpolating audio samples in time. Changes brought about by resampling are deterministic and known [10, 9] yet nonlinear with respect to a filter bank representation.

A typical descriptor extraction process divides an input signal into uniform-size analysis frames (overlapping or non-overlapping). Resampling changes the length of its input, and as such, either the number or spacing of frames in analysis must change, i.e. under a fixed analysis frame rate, there will be different numbers of input and output frames. Hence, input and output frames can no longer be directly compared.

One approach to predicting descriptors under varying length would be to restrict the input to single centered frames, and the resampling factor L from 0.5 to 2. Another approach is to predict descriptors of frames in aggregate (e.g. mean) rather than for each frame. This is the approach chosen for this study; input sounds are stationary and mean output descriptors are predicted from mean input descriptors.

2.2.1 Analytic Model

An analytic model for resampling (for comparing with learned models) can be derived based on the observation that resampling is equivalent to spectral interpolation [10]:

$$f(\vec{fb_{in}}, L) = L^{\alpha} \cdot \text{INTERP}(\vec{fb_{in}}, L) \quad (2)$$

and gains per band are scaled according to the change in length, and α is a constant that minimizes error on the training database ($\alpha \approx 0.5$). However, the ratio of mean output to mean input seems to vary quite a bit by L and instance (between 0.8 and 1.4), so this could be improved.

2.3 SVRs for Descriptor Prediction

Support Vector Machines are a supervised learning technique frequently used for classification and regression [12] (referred to as Support Vector Regression, or SVR). These techniques rely on a formulation of linear regression generalized by a mapping into a higher dimensional space (the kernel trick) and solved as a quadratic programming problem. *Support vectors* refer to a subset of input patterns selected and weighted by a training process. As suggested in a practical guide to SVMs [8], we use the Radial Basis Function (RBF) kernel, which maps examples to a space corresponding to their distance from the support vectors.

Predicting our vector of descriptors entails predicting output descriptors for all filter bands. There exist formulations of SVRs for the vector-valued case [1], but implementations are not widespread. Fortunately, aggregating single SVRs for each output are equivalent under some criteria and have been reported to give similar performance [1].

Finding an optimal model with SVRs typically consists in training and evaluating models in search for the best model parameters, usually using a grid search. Besides the training size, which is an implicit (and important) parameter, there are three parameters that strongly affect the model performance: C, the complexity parameter, is a regularization parameter of classic support machines; it determines the tradeoff between models that fit the data closely and flatter models (hyperplanes closer to zero). Some regression machines also have an additional parameter, ϵ (or precision), that determines the width of the insensitive-tube in the error function, informing the model which scales of details are considered significant. The optimal value of this parameter should be related to the inherent noise level in the data [11] (and thus sensitive to the scale of the labels). Finally, we must optimize any kernel parameters, in this case γ, the radius of the RBF kernel (a spherical gaussian), which determines the selectivity of the model; how many of the support vectors are used to determine the label of the query instance.

3. EXPERIMENTAL PROCEDURE

First, we generated a database of 400 input sounds by taking uniform white noise samples of 0.5 seconds each (44.1k) and filtering them by a spectral distribution described by a Hann window surrounded by stop regions a) covering a certain width of frequency from 5% to 90% (20 values) b) with the non-stop section shifted to different positions, from hard left to hard right (20 values). Once generated, input sounds were randomly partitioned into training and test with an 80% training split.

3.1 Learning Database

Next, we resampled each input sound under different resampling factors L; a uniformly-spaced grid of parameters with 41 values of $log_2 L$ i.e. $(-1, -.95, .., 0, .., .95, 1)$ or between 0.5 and 2 in terms of L. From each of the input and output sounds we averaged the magnitude spectrum of all frames, dropping silent frames at the edges, using an Hann window of 2048 points, an overlap factor of 2, and a zero-padding factor of 4. Then, from each time-averaged spectrum we took a magnitude average over 16 bands uniformly divided by frequency. We export these features as training and test databases to the Weka ARFF format [5] for the learning tasks and build and evaluate LibSVM models [3].

To summarize, we have two databases, a train and test, which consist of 320 and 80 input sounds. Each input sound has 41 corresponding output sounds, each of which is described by a transformation parameter $log_2 L$, an input envelope of 16 bands, and an output envelope of 16 bands.

3.2 Selection of Learning Parameters

Model parameters are evaluated by exhaustive search over a coarse grid. For simplicity, all features are used as inputs, and a global set of model parameters is used, though these should be updated (i.e. per band model parameters and feature selection) in future work.

A grid of parameters (such as in Figure 1) for training size, C, ϵ, and γ is iterated over. For each set of model parameters we store corresponding predictions on the test examples and a subset of the training examples. Once generated, we can compare the predictions with the actual transformed descriptors and generate error statistics. We describe performance as well as select the best performing model with the mean of absolute residual ($\overline{|r|}$) over all bands.

4. RESULTS

For the model with best performance, we predict the transformed descriptors of the test set with a mean residual per

[2] At times as $\log_2 L$, negative transposition in octaves

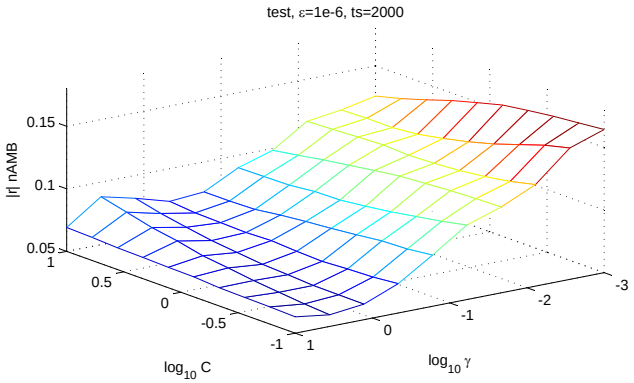

Figure 1: Coarse grid search for acceptable ranges of γ and C.

band of 0.0326 nAMB which is around 16.2% of the mean output value (0.206 nAMB). We achieve this performance using a training set of 12k input-param-output tuples, complexity C of 3.162, kernel γ of 4.642, and a model precision ϵ of 1×10^{-6}. By contrast, the analytic model in Section 2.2.1 achieves an average residual per band of 0.0103 nAMB, which is around 5.14% of the mean output value.

At first, we overlooked the importance of the ϵ parameter causing our model to fail to attend to the finer details of the regression. Using typical values of the other parameters we found a more suitable value that improved the performance by an order of magnitude (effect shown in Figure 4).

Figure 1 depicts the major trends in the coarse grid search. We see a roughly paraboloid shape, as well as a trend pointing towards greater γ and lesser C in on the test set. For this training size, it appears that the best solutions could be found in $\log_{10}\gamma$: [0, 2], $log_{10}C$: [-3, 0], or overlapping with the corner facing us in the figure (region partially shown).

Along an axis of constant C, we see individual minima with respect to γ that shift slightly according to C. In our formulation of the RBF kernel, γ is inverse to the radial distance of the kernel ($K(x,y) = e^{-\gamma\cdot<x-y,x-y>^2}$) so that smaller γ means a less exclusive kernel that covers many of the training examples and that larger γ means a more exclusive kernel that covers only a few close training examples.

By contrast, we see the trend with respect to C to be more subtle and more sensitive to the other parameters such as γ. Nonetheless, when C is less than the optimal value it would correspond to underfitting the data, and when C is greater than the optimal value it would correspond to overfitting the data, C seems to be sensitive to the training size, while the optimal value of γ seems less so.

From subsequently larger training sets we can show that the optimal value of γ doesn't change much with training size, and that optimal value of C increases with the increase in training size (more data allows a more complex and less smooth model).

trainsize	best ($\overline{\lvert r\rvert}$ nAMB)	best γ	best C
12000	0.0326	4.6	3.16
8000	0.0369	4.6	1
4000	0.0460	4.6	0.316
2000	0.0594	3.59	-1*

Table 1: Optimal values of γ and C for different training sizes. * minimum for coarse grid was on edge, hence may not be optimum.

By examining the local behavior of the performance at the largest training size, we can show that the behavior is parabolic with respect to γ and C.

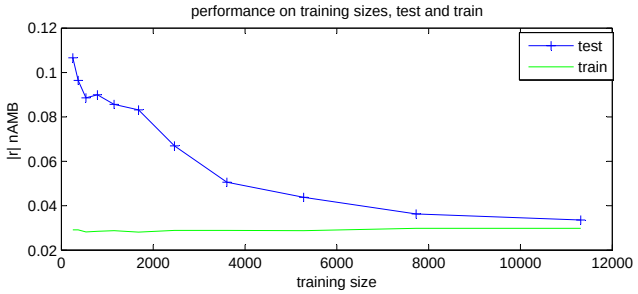

Figure 2: System performance with respect to trianing size. As training data increases, test performance approaches training performance.

To argue that we have sufficient data for learning to converge, we can examine the performance on the test set with respect to the amount of training data. While we haven't reached a point where the performance flattens out, we come quite close to the performance on the training set, (as seen in Figure 2) with the error on the test set flattening out as training size increases.

4.1 On the Learning Parameters

In the following section, we illustrate the effect of each model parameters on the vector-valued prediction result, showing how changes in those parameters affect the predictions produced. We illustrate this with triads of plots: the true values after transformation, a predictive model with suboptimal parameters (we attend to this to see the effect of the parameters), and a model with good parameters. In each plot the colored lines show the descriptor vector (filter bank) at different values of the resampling parameter between 0.5 and 2. The bold line is the transformation with resampling factor L=1, also equivalent to the input.

Training Size For an insufficient training size, it seems that test examples don't converge, i.e. there are insufficient training patterns within the neighborhood of the RBF kernel for a particular query point (not shown).

γ As stated earlier, small γ means large promiscuous kernel (many training patterns are active for a query pattern) and large γ means a small and selective kernel (few training patterns are active for a query pattern). As such, overly small gamma will result in overly smooth approximations. By contrast, an overly large and selective γ will result in not enough training patterns for a query, so some examples will fail to converge. See Figure 3.

ϵ (Precision) controls the width of insensitive-tube for the error function. Thus, if it is too large, the regression will ignore the smaller details in the target function. By contrast, if it is too small, it will expend lots of computational effort overfitting to noise. See Figure 4

4.2 Distribution of Test Error

By Input Sample In the discussions of model parameters above (Section 4.1), we develop the intuition that there were model parameters under which the approximation did not "converge". By averaging performance for each of the test input sounds, we can show how different inputs have vastly different average performance. Examining the sorted performance curve shows two subsets of test examples with respect to performance divided by an inflection point: a broad section of about 70% of examples with relatively uni-

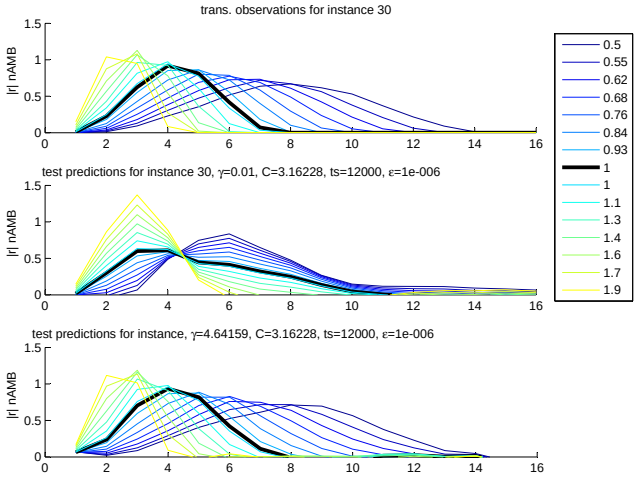

Figure 3: With smaller γ (middle) we produce an overly smooth approximation.

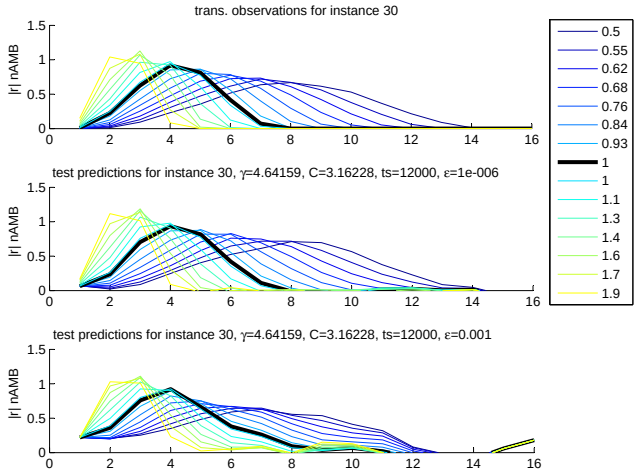

Figure 4: With overly large ϵ (bottom) we fail to attend to the details of the target function.

form (low) error, and the other 30% of examples with a sharper performance difference. These examples seem not to converge (not yet known why), and contribute a proportionally higher amount of error to the approximation (that perhaps merits further investigation).

By Transformation Parameter (L) We examine average performance by resampling parameter value, but we see that it is relatively flat (varies between 0.027 and 0.041 nAMB) especially compared to the difference among test input sounds as above. This could be because the training database had an equal amount of examples for each parameter value (test distribution equal to training distribution).

5. CONCLUSIONS

Using a model only generated from data produced by the transformation itself, we are able to achieve performance almost to factor 3 of an analytic model, informed by knowledge of the transformation. As such, it opens up the possibility of building accurate models of transformations for which we don't have such knowledge or insight. In addition, we have in some way characterized the failure modes of SVRs for vector prediction of transformed audio descriptors.

We thank Esteban Maestre and Ferdinand Fuhrmann for their helpful suggestions on the paper.

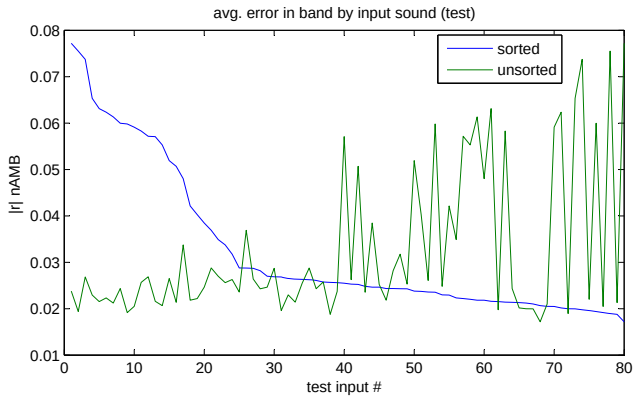

Figure 5: Sorted examples 1-25 vary more in terms of average error and contribute more error proportionally than the majority of examples.

6. REFERENCES

[1] M. Brudnak. Vector-Valued support vector regression. In *Neural Networks, 2006. IJCNN'06. International Joint Conference on*, pages 1562–1569. International Neural Network Society, July 2006.

[2] M. Caetano and X. Rodet. Evolutionary spectral envelope morphing by spectral shape descriptors. In *Proceedings of ICMC2009*, August 2009.

[3] C. C. Chang and C. J. Lin. *LIBSVM: a library for support vector machines*. 2001.

[4] G. Coleman and J. Bonada. Sound transformation by descriptor using an analytic domain. In *Proceedings of DAFx 2008*, pages 341–347. DAFx Conferences, September 2008.

[5] S. R. Garner. Weka: The waikato environment for knowledge analysis. In *In Proc. of the New Zealand Computer Science Research Students Conference*, pages 57–64, 1995.

[6] S. L. Groux and P. F. Verschure. Perceptsynth: mapping perceptual musical features to sound synthesis parameters. In *IEEE International Conference on Acoustics, Speech and Signal Processing, 2008.*, pages 125–128, March 2008.

[7] M. Hoffman and P. Cook. The Featsynth framework for feature-based synthesis: Design and applications. In *Proc. International Computer Music Conference (ICMC-07)*, pages 184–187, August 2007.

[8] C. W. Hsu, C. C. Chang, and C. J. Lin. *A Practical Guide to Support Vector Classification*. http://www.csie.ntu.edu.tw/~cjlin/libsvm/, 2003.

[9] J. O. Smith. *Digital Audio Resampling Home Page*. http://www-ccrma.stanford.edu/~jos/resample/, January 28, 2002.

[10] J. O. Smith. *Mathematics of the Discrete Fourier Transform (DFT)*. http://ccrma.stanford.edu/~jos/mdft/, 2007.

[11] A. Smola, N. Murata, B. Schölkopf, and K. R. Muller. Asymptotically optimal choice of ε-loss for support vector machines. In *Proceedings of the International Conference on Artificial Neural Networks, Perspectives in Neural Computing*, pages 105–110, September 1998.

[12] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, September 2003.

[13] M. Yee-King and M. Roth. Synthbot: An unsupervised software synthesizer programmer. In *Proc. International Computer Music Conference (ICMC-08)*, pages 184–187, August 2008.