Non-Linear Semantic Embedding for Organizing Large Instrument Sample Libraries

Eric J. Humphrey, Aron P. Glennon and Juan Pablo Bello
Music and Audio Research Lab (MARL), New York University
{ejh333, apg250, jpbello}@nyu.edu

***Abstract*—Though tags and metadata may provide rich indicators of relationships between high-level concepts like songs, artists or even genres, verbal descriptors lack the fine-grained detail necessary to capture acoustic nuances necessary for efficient retrieval of sounds in extremely large sample libraries. To these ends, we present a flexible approach titled Non-linear Semantic Embedding (NLSE), capable of projecting high-dimensional time-frequency representations of musical instrument samples into a low-dimensional, semantically-organized metric space. As opposed to other dimensionality reduction techniques, NLSE incorporates extrinsic semantic information in learning a projection, automatically learns salient acoustic features, and generates an intuitively meaningful output space.**

I. Introduction

Navigating the overwhelming wealth of available multimedia content is now a widely appreciated problem and there is clear motivation to develop methods of organizing and searching this data via computational means. In the information retrieval community, there is a growing opinion that metadata and social activity may be far richer indicators of relationships between items than those obtained by directly analyzing the content [1]. While some systems based on human 'signals' may out-perform those based on, for example, musical ones, there are still multimedia retrieval tasks where tags and context fail to provide sufficient information.

Searching and exploring musical instrument sample libraries is one such instance, where sparse metadata and text-based querying methods often reduce the task of navigating a sound library to that of an exhaustive, brute force search. The latent difficulty in retrieving a target sound from a potentially massive collection of items lies in the inability to capture the specific query linguistically, relying instead on metaphors; distorted guitars are described as 'crunchy' or trumpets 'bright.' At the granularity of a single sound, tag information is too coarse to prove useful for organization that captures a specific item's acoustic subtleties.

Instead of translating sound into cumbersome, semantic intermediaries for various retrieval tasks, the ideal sound retrieval representation bypasses this process entirely and leaves data points in a semantically-organized space. We additionally impose the criteria that exploring this space be intuitive, which entails low dimensionality for the purposes of visualization. Unfortunately, most previous work in musical sound organization, typically in the form of instrument classification, does exactly the opposite, creating high-dimensional feature spaces so that classes can be differentiated with high accuracy. Though the nuances differ from one system to another, musical instrument classification generally proceeds by first designing a set of features and subsequently parsing the results with a classifier of choice. We refer the interested reader to [2], [3] or [4].

Here, we present a generalized approach referred to as non-linear semantic embedding (NLSE) capable of both learning salient signal-level features and a low-dimensional projection into a semantically meaningful output space for the purposes of organizing an instrument sample library. Importantly, NLSE addresses the limitations of current statistical dimensionality reduction techniques, such as MDS, PCA or LLE. Unlike these methods, NLSE explicitly encodes semantic information in the transformation, makes minimal assumptions about salient acoustic features, generalizes well to unseen data and produces an output space where distance is meaningful. The remainder of the paper is organized as follows: Section II describes the core components of the proposed technique, Section III outlines our methodology for training and evaluation, Section IV presents and discusses the results of the experiments, and Section V offers conclusions and directions for future work.

II. Non-Linear Semantic Embedding

Building upon previous work in deep learning, we present a generalized technique to project high-dimensional audio representations into a low-dimensional geometric output space that preserves semantic relationships. The four primary elements of the method consist of automatic feature extraction, a defined metric output space, iterative training by semantic example, and an appropriate time-frequency representation.

A. Automatic Feature Extraction

There are several clear reasons why it is advantageous to automatically learn features from some minimally processed input representation. Most importantly, reducing an input to a set of designed features makes the assumption that the information captured by those features is the most significant for a given task, and that the information discarded as a result is irrelevant. For tasks like musical instrument classification, feature extraction design can be tantamount to an exercise in creative thinking, limited only in scope to what one may think might be relevant. Additionally, automatically learning a set of features allows several applications to potentially stem from the same source representation without needing to transform the data for different goals.

In an ideal scenario, salient signal-level features are learned by a trainable function. Traditional neural networks are extremely poor at robustly processing raw input representations as the spatial organization of the input vector is inconsequential to the network; if the data in an input vector were circularly shifted one position, the output of the network could change drastically. Convolutional neural networks (CNN), pioneered by Lecun et al in [5], completely resolve this issue, providing two key benefits: sensitivity to highly correlated data and varying degrees of translation invariance.

CNNs are an extension of classical discriminant functions in that weights are shared across an input vector as spatially shifted convolutions, typically followed by a downsampling operation and a non-linear squashing function; an example is contained within the diagram in Figure 1. The general formula for a 2-dimensional convolution of an input X and kernel W_k is given in (1), where a valid convolution is defined only on the region where both X and W_k are themselves defined.

$$(X * W_k)[m,n] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} X[i,j] W_k[m-i, n-j] \quad (1)$$

Dimensionality can be further reduced by following a valid convolution with a downsampling operation, often by taking the maximum value over a region, referred to as max-pooling, adding a bias and applying a non-linear activation function. While convolutions introduce translation invariance, downsampling provides scale invariance, where some target feature in the signal may span more or fewer points in the vector. Collectively, each layer of a CNN maintains several sets of weights, or kernels, that are convolved with its inputs, producing a set of output vectors, called feature maps. Successive convolutional layers can learn hierarchical scale and shift invariant features, and are powerful enough to be applied directly to raw input representations.

B. DrLIM

While many dimensionality reduction techniques focus primarily on statistical decompositions, Hadsell et al present a unique approach titled 'Dimensionality Reduction by Learning an Invariant Mapping' (DrLIM) [6], where a learning machine is trained by providing extrinsic similarity information about how the training data are to be organized relative to each other. DrLIM also introduces the novel aspect of imposing Euclidean distance on the output, resulting in a naturally geometric space.

Supervised training paradigms conventionally proceed by labeling a large plurality of data and iteratively presenting it to a parameterized function, computing the loss, propagating errors backwards through the function and slightly adjusting the function's parameter set until convergence. When these labels are nominal, each category must have some defined target representation such that the machine approaches this goal. The need to determine this output *a priori* is circumvented by adopting a Siamese architecture and training the parameterized function based not on a single data point, but the relationship between two.

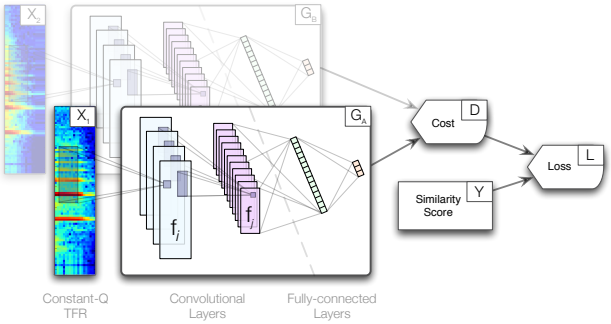


Fig. 1: Siamese Training Architecture

The DrLIM pairwise training strategy proceeds thusly, illustrated comprehensively in Figure 1: first, a parameterized function $Z_1 = G_A(X_1|W)$, with input vector X_1 and parameter set W, is designed for some target application, where the number of output units for G_A is chosen corresponding to the desired dimensionality. This function is duplicated exactly as $Z_2 = G_B(X_2|W)$, such that $W_{G_A} = W_{G_B}$. A cost function $D = C(Z_1, Z_2)$ is defined between the two outputs forming the Siamese architecture and followed by a contrastive loss function $L(D, Y)$, where Y is a semantic similarity score between the inputs X_1 and X_2 taking a value in $[0, 1]$; a diagram of this training architecture is provided in Figure 1.

The contrastive loss function $\mathcal{L}$ is defined by the weighted linear combination of two separate losses, given in (2): a similarity loss L_{Sim} and a difference loss L_{Diff}. During training, the system is always minimizing the total loss by attracting 'similar' training samples and repelling 'dissimilar' ones until stopping conditions are met. When two items are maximally similar, $Y = 1$ and the similarity loss (3) will be minimized, defined here as a square loss; when two items are maximally dissimilar, $Y = 0$ and the difference loss (4) will be minimized, defined here as a log-loss with parameters m defining a margin and Q controlling the knee of the curve.

$$L(X_1, X_2, Y|W) = Y * L_{Sim}(D) + (1-Y) * L_{Diff}(D) \quad (2)$$

$$L_{Sim} = \frac{1}{2} Y * D_W^2 \quad (3)$$

$$L_{Diff} = \frac{(1.0 - Y)}{2 * Q^2} * \log(1.0 + e^{Q*(m - D_W)})^2 \quad (4)$$

Due to the relative nature of training, where absolute information is secondary to the relationships between data, the system will attempt to resolve discrepancies when the same semantic label is associated with very different signals. And, since this training paradigm is constantly trying to maximize and minimize point-wise distances accordingly in the output space, organization of the space is intrinsically metric within the context of the semantic information provided during training.

C. Time-Frequency Representation

While translation invariance is one of the primary benefits of using a CNN, musical pitch is naturally logarithmic. As a sound increases in frequency, the separation between its harmonic partials does as well. To take full advantage of a CNNs strengths, the time-frequency representation (TFR) of the input audio must be crafted such that pitch changes linearly as a function of frequency. This can be achieved with the Constant-Q transform [7], which can be conceptualized as a base-2 logarithmic downsampling operation and efficiently implemented as a complex matrix multiply with the short-time Fourier transform (STFT).

D. Implementation Details

The entire dataset is transformed by the STFT with a frame size of 4096 and 50% overlap and a sampling rate of 44.1kHz, resulting in a frame rate of 21.53Hz. The modified Constant-Q kernel is generated with 84 filters spanning 50–11025Hz at 12 bins per octave. Hamming windows are raised to the 8^{th} power to considerably narrow the filters and the magnitude responses are normalized to $0dB$. Whereas normal Hamming windows result in a stopband attenuation of about $-40dB$ and overlap with several adjacent filters on either side, this narrowing process results in filters with a $-60dB$ cutoff point approximately aligned with the center frequency of either adjacent filter, producing a much sharper spectral representation.

We define the input to the CNN to be a 'tile' of 20 frames (roughly one second) by 84 constant-Q coefficients and construct a network of 2 convolutional layers and 3 fully-connected layers. The first layer has 30 kernels with a (5×13) shape and a downsampling field of $(2, 4)$; similarly, the second layer has 50 kernels with a (5×7) shape and a downsampling field of (2×2). The fully connected layers consist of 200, 50, and 3 units. In total, the architecture has 134,183 parameters (weights and biases) and two model hyper-parameters ($m = 1.25$ and $Q = 10.0$). The machine is implemented using the Theano package [8] developed by the LISA Machine Learning Laboratory at the University of Montreal.

Training proceeded by presenting batches of 600 randomly selected tile pairs from the training set and performing mini-batch stochastic gradient descent. The learning rates were set according to each experiment, and all training sessions were run for a maximum of 5000 iterations.

III. Methodology

A. Data

We restrict our immediate work to solo instrument samples from the Vienna Sound Library, a substantial collection of high-quality instrumental content intended for professional music applications. Of the 32 general instrument categories, we trim the dataset down to the 12 largest solo instruments to maintain sufficient training data for each, the details of which are given in Table I. The set of content is further reduced by using only samples from the 'Einzelnoten', 'Dynamics' and 'RS' subsets in an effort to minimize noisy training data, as these three subsets primarily consist of single pitch sounds played within the standard technique of the instrument.

TABLE I: Instrument Sample Distribution

Family	Instrument	Symbol	Files	Time (min)
Brass	Trumpet	Tp	5935	345.432
	French Horn	Fr	5340	301.181
	Trombone	To	6072	326.404
	Tuba	Tu	4915	275.471
Double Reed	Oboe	Ob	2961	159.896
	English Horn	En	2986	197.736
	Bassoon	Ba	4717	303.102
Single Reed	Clarinet	Cl	5644	325.343
	Tenor Saxophone	Sx	2292	107.188
String	Violin	Vi	4788	273.509
	Cello	Ce	3557	214.488
Aerophone	Flute	Fl	4108	228.679
		Total	53315	3058.431

TABLE II: Instrument Configurations

Index	Set	Count
1	Tp, Fr, To, Tu	4
2	Tu, Ob, Cl, Ce, Fl	5
3	Ob, En, Ba, Cl, Sx	5
4	Tp, Fr, To, Tu, Cl, Sx	6
5	Tp, Fr, To, Tu, Ob, En, Ba	7
6	Tp, Fr, Ob, En, Cl, Sx, Vi, Ce	8
7	Tp, Fr, To, Tu, Ob, En, Ba, Cl, Sx	9
8	Tp, Fr, To, Tu, Ob, En, Ba, Cl, Sx, Fl	10
9	Tp, Fr, To, Tu, Ob, En, Ba, Cl, Sx, Vi, Ce, Fl	12

B. Training

For the purposes of training, similarity is defined at the instrument level on the assumption that taxonomical boundaries roughly coincide with acoustic homogeneity. This is a coarse assumption, but it further serves to underscore the challenge in verbally describing a large number of sounds. All sounds associated with the same instrument are considered similar ($Y = 1$), and all others are dissimilar ($Y = 0$). Additionally, the dataset was divided into 11 disjoint sets (9 training, 1 validation, 1 test) so that performance could be cross-validated and averaged.

We observed that a presentation ratio of 50% positive-to-negative similarity reinforcement facilitated training, regardless of the number of classes. Without forcing this ratio, randomly selecting pairs would only produce a $^1/_{|\mathcal{C}|}$ ratio of positive examples, where $|\ \mathcal{C}\ |$ is the number of unique classes. When the number of classes is large, there is more average repulsion during training than attraction, and convergence is impeded as a form of training bias.

C. Experiments

To evaluate the applicability of NLSE for musical instrument organization, we outline three experiments each aimed at

quantifying specific performance criteria. The first experiment investigates the impact of different instrument configurations on local structure and boundaries in the output space as compared to other dimensionality reduction algorithms. Next, we quantify global organization and distance with a cross-validated ranked retrieval task, again comparing to other methods. Finally, we measure the impact of training data size on the NLSE performance at the same ranked retrieval task.

1) Semantic Organization: Using the training data as anchor points in the output space, we perform k-Nearest Neighbor classification with the test set over 8 different instrument configurations, shown in Table II. These configurations are selected specifically to explore instrument family and capacity effects. For example, (1), (3), (4), (5), (7) are based around mixtures of instrument families. (2) only has one of each for maximum distinction and (6) has two from each family for increased confusion. (8) consists of all wind instruments, and (9) is all 12 instruments.

We compute classification accuracy specifically using an naïve classifier (k-nearest neighbor) to quantify the inherent configuration of the three output spaces. As a classifier, kNN focuses on neighborhoods and provides information about local semantic organization. Simply put, for the purposes of sound library exploration, most points near a given coordinate in the output space should be semantically similar. Additionally kNN performs very well in the event of non-linear boundaries, the occurrence of which is typically lost in other probabilistic classifiers like maximum-likelihood estimation (MLE).

2) Ranked Retrieval: As a second experiment, we conduct a ranked document retrieval task to address two questions. As a counterpart to the previous experiment, ranked retrieval provides genuine insight into both the global organization and the order of an entire set of relevant items. More importantly, this serves as a quantitative way to assess performance on the target application – semantic sound organization and retrieval.

To perform ranked retrieval over all sound files, models are trained 10-fold in the last instrument configuration (9) and applied to each TFR in 50% overlapping tiles. The output for each input tile is averaged across the file and associated with the original metadata. Evaluation proceeds by calculating the averaged Recall-Precision (RP) curve and the Mean Average Precision (MAP) across that specific fold's corresponding test set.

3) Training Data Dependence: In addition to characterizing performance achieved by the approach presented here, it is common caveat that trainable functions are typically data hungry machines and require large amounts of training data to generalize. As a final test, we explore the relationship between the size of the dataset for training as a percentage of all of the data used in the previous experiments. With a dataset of roughly 48k sound files for training, a set of four NLSE models were trained at three different percentages of the total dataset $[50\%, 25\%, 10\%]$, with similarly folded data as before. After training, the evaluation from Experiment 2 was repeated.

D. Comparison Algorithms

It is a curious consequence of the NLSE approach that no comparable dimensionality reduction methods are designed to operate directly on a raw TFR like that used for NLSE. In fact, directly applying methods like PCA or LLE to something like a constant-Q representation fails outright on instrument and timbre identification tasks. As the primary variance is explained by loudness (energy) and pitch (fundamental frequency), PCA emphasizes those attributes that are not instrument specific. On the other hand, LLE makes no attempt to resolve pitch translations and cannot help but associate pitch neighbors across instruments more so than timbral ones. The intuition is analogous to the direct application of LLE to images of faces, where two different faces in the same absolute position are typically more similar locally than the same face appearing in two different positions.

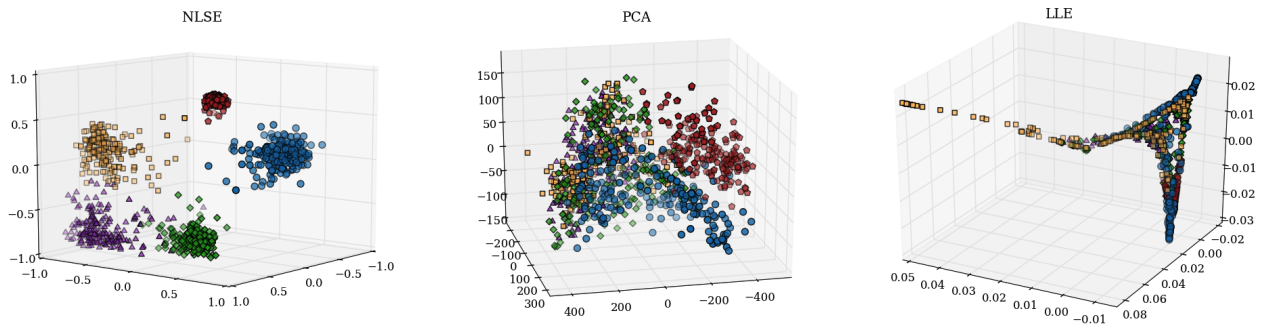
MFCCs, alternatively, are widely believed to capture a pitch-invariant representation well-suited for timbral analysis and commonly used in research on musical instrument classification and other acoustic similarity work. Though it is admittedly less than ideal to compare methods on different input representations, we aim to provide an input representation that is more conducive to how PCA and LLE function. To these ends, we create an equivalent data set of MFCC tiles from the sound files detailed in Section III-A. We form 20×12 tiles (time by MFCC), using coefficients 2 through 13 in keeping consistent with the literature [9]. We omit the first cepstral coefficient, as it corresponds to average power, or loudness.

In our experiments, we use the Scikits.Learn PCA [10] and MDP-Toolkit LLE [11] as comparative implementations . Because LLE must calculate a square distance matrix between all input vectors to compute an embedding, the implementation requires an inordinate amount of computer memory, is entirely unable to scale and simply cannot incorporate as much information during training as NLSE or PCA. As a result, the training capacity is pushed to the limits of the system (about 12,000 tiles from disjoint files for training, requiring approximately 4GB) and averaged across several iterations. Both PCA and NLSE are able to synthesize an order of magnitude more training samples than this, although it is timely to note that NLSE has no theoretical limit on the amount of training data it can incorporate in the tuning of parameters.

IV. Results and Discussion

After evaluating the models across the three experiments, NLSE performs exceptionally well at organizing musical instrument sounds in a very low dimensional space. As shown in Figure 2, it is visually striking to what degree NLSE is able to discern between classes and cluster similar data points accordingly. It consistently exhibits both local and global structure, and is surprisingly rather insensitive to reductions in the size of the training set.

The mean, min and max accuracy from the kNN-classifier across instrument configurations are shown in Figure 3, where the value of k was varied across the set $[5, 8, 12, 20, 32, 50]$.



(a) Non-linear Semantic Embedding (b) Principal Components Analysis (c) Locally Linear Embedding

Fig. 2: Output spaces generated by the three methods, shown for configuration (1): Tuba (red pentagons), Oboe (green diamonds), Clarinet (purple triangles), Cello (blue circles) and Flute (yellow squares).

For the first three cases, NLSE averages an overall classification accuracy over 98%, whereas both PCA and LLE are just shy of 70%. NLSE remains above 90% until the 7^{th} configuration, while PCA and LLE are both between 30%–40% lower than NLSE for all cases. As is somewhat expected, classification accuracy decreases as a function of class count. However, the difference from best to worst is about 15% (99% to 84%), 25% (69%-44%) and 24% (68%-44%) for NLSE, PCA, and LLE respectively. This is numerically indicative of the fact that NLSE is forming tighter and better separated clusters.

The RP-curves for the comparative ranked retrieval task were max-smoothed following the TREC standard and interpolated to 11 points on the range $[0, 1]$, inclusive. Query-averaged RP-curves from each modeled were then themselves averaged to produce the cumulative RP-plot is shown in Figure 4. As may be expected from the first experiment and images of the corresponding embeddings, NLSE produces an embedding significantly more conducive to the task of ranked item retrieval than PCA or LLE. The MAP scores for NLSE, PCA and LLE are 0.734, 0.212, and 0.13 respectively, showing a significant drop between classification and ranked retrieval performance for PCA and LLE.

Lastly, the RP-curves for reduced training set size with NLSE are shown in Figure 5 and the MAP scores were calculated as 0.834, 0.814, and 0.797. The three percentages of 50%, 25% and 10% corresponded to approximately 24k, 12k, and 4.8k files used for testing, respectively. As a rough estimate, the lowest case (4.8k files) is equivalent to approximately 270 minutes of audio across all 12 instruments, or about 22.5 minutes each. Even with the substantial reduction in data, the learning rate was increased to 0.05 from 0.02 and every model's performance increased over Experiment 2. Not only is this encouraging, but suggests that the early stopping criteria used was not patient enough and performance results for the first two experiments could probably be higher.

It is intuitively satisfying that LLE outperforms PCA on a kNN classification task, but the embedding shown in Figure 2-c lacks any kind of intuitive interpretation. As demonstrated by the Cello points (blue circles), there are areas where data points are well localized, but from a semantic context, the taxonomy is scattered in multiple clusters, explaining the discrepancy between classification accuracy and MAP in retrieval. The impact of this is apparent in the RP-curves shown in Figure 4, where LLE performs the worst at ranked retrieval. While locally coherent, LLE is helpless to maintain global class structure in the output space.

Almost inversely, PCA does a better job of keeping an entire taxonomy together, referring again to Figure 2, and this is reflected quantitatively in the MAP from the ranked retrieval task. While it is clearly organized in some manner, it is certainly not along instrument boundaries. Without performing a qualitative analysis to ascertain whether or not the organization coincides with some salient dimensions, it is impossible to know the usefulness of this space beyond the metrics outlined here.

Overall, NLSE performance is encouraging on its own merits. Examining any of the RP-plots corresponding to an NLSE model, there is noticeable negative concavity to the curve once recall reaches roughly 60%–70%. This occurs when precision begins to decrease much faster than recall, suggesting the edge of a cluster or dense area has been reached. This is meaningful insofar as introducing early-stopping criteria for ranked retrieval of relevant documents, or alternatively identifying areas of confusion that may be interesting for sound exploration or manually attributing metadata. Opportunities for graceful degradation in anything that can be construed as an expert system – which this could arguably be – are extremely useful in the context of building robust tools that could be deployed 'in the wild,' as it were.

V. Conclusions

The immediate observation to make is that, given sufficient patience, NLSE performs most impressively at the task of organizing this specific dataset. The system is able to reasonably place 12 non-impulsive instruments into a 3 dimensional

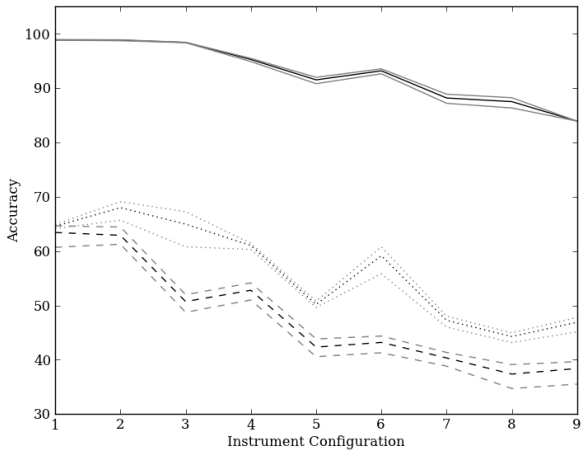


Fig. 3: kNN Classification across instrument class configurations for each algorithm, NLSE, PCA and LLE respectively. Lighter lines indicate minimum and maximum classification accuracy per method.

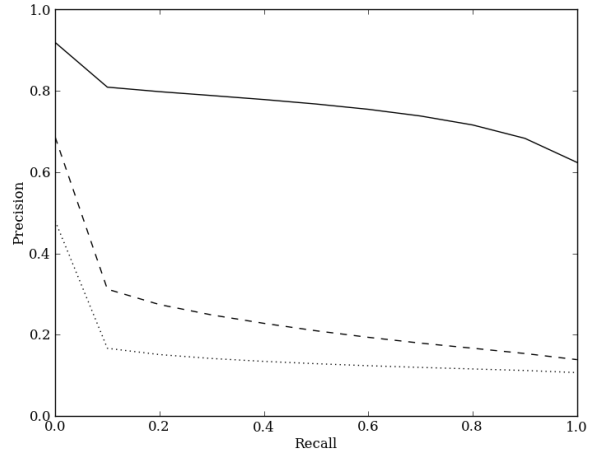


Fig. 4: Recall-Precision for NLSE (solid), PCA (dashed) and LLE (dotted).

space. It cannot be stressed enough that any gains realized via come at the expense of a substantial training phase. However, as the model is itself a function, it is easy to apply to new data once adequately trained. To this point, future work includes quantifying how a given model generalizes to other data in this dataset, the same instruments in other datasets, or possibly even entirely new instruments.

It is surprising to note that the critical minimum amount of data required for NLSE to work reasonably well on the Vienna Sound Library is below 22 minutes of data per instrument. This is certainly something to determine proceeding forward, but there is also a chance that the quality of the audio content used makes the task easier on the machine. Knowing now that the approach works on at least one dataset, comparisons with

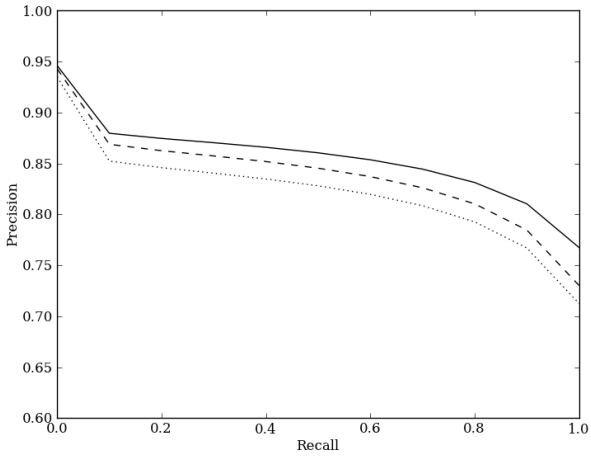


Fig. 5: Recall-Precision for 50% (solid), 25% (dashed) and 10% (dotted) of all training data.

others and sensitivity to noise (mislabeled data, reduced audio quality, etc.) can be systematically evaluated.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation under grant IIS-0844654.

REFERENCES

[1] M. Slaney. "Web-Scale Multimedia Analysis: Does Content Matter?" *IEEE MultiMedia*. 18(2): 12–15 (2011)

[2] S. Essid, G. Richard, B. David. "Musical Instrument Recognition by Pairwise Classification Strategies." *Audio, Speech, and Language Processing, IEEE Transactions on* (2006) 14 (4): 1401–1412

[3] A. Livshin, X. Rodet. "Musical Instrument Identification In Continuous Recordings." *Proc. Int. Conf. Digital Audio Effects (DAFX)*. 2004. 222–227

[4] B. Sturm, M. Morvidone, L. Daudet. "Musical Instrument Identification using Multiscale Mel-frequency Cepstral Coefficients." *18th European Signal Processing Conference (EUSIPCO-2010).*

[5] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. "Gradient Based Learning Applied to Document Recognition." *Proceedings of IEEE,*. 86(11):2278–2324, 1998.

[6] R. Hadsell, S. Chopra, Y. LeCun. "Dimensionality Reduction by Learning an Invariant Mapping." *Conference on Computer Vision and Pattern Recognition. (CVPR)*: 2006.

[7] J. Brown, M. Puckette. "An efficient algorithm for the calculation of a constant Q transform." *J. Acoust. Soc. Am.* 92(5):2698–2701, 1992.

[8] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley and Y. Bengio. "Theano: A CPU and GPU Math Expression Compiler." *Proceedings of the Python for Scientific Computing Conference (SciPy) 2010.* June 30 - July 3, Austin, TX

[9] H. Terasawa, M. Slaney,J.Berger. "The Thirteen Colors of Timbre." *2005 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*.

[10] "Scikits.Learn: Machine Learning in Python." version 0.8.1. http://scikits-learn.sourceforge.net

[11] T. Zito, N. Wilbert, L. Wiskott, P. Berkes. "Modular toolkit for Data Processing (MDP): a Python data processing frame work." *Front. Neuroinform.* 2008. 2:8.