Multi-scale Geometric Summaries for Similarity-based Sensor Fusion

Christopher J. Tralie
Department of Mathematics
Duke University
ctralie@alumni.princeton.edu

Paul Bendich
Department of Mathematics
Duke University
Geometric Data Analytics, Inc.
bendich@math.duke.edu

John Harer
Department of Mathematics
Duke University
Geometric Data Analytics, Inc.
harer@math.duke.edu

***Abstract*—In this work, we address fusion of heterogeneous sensor data using wavelet-based summaries of fused self-similarity information from each sensor. The technique we develop is quite general, does not require domain specific knowledge or physical models, and requires no training. Nonetheless, it can perform surprisingly well at the general task of differentiating classes of time-ordered behavior sequences which are sensed by more than one modality. As a demonstration of our capabilities in the audio to video context, we focus on the differentiation of speech sequences.**

Data from two or more modalities first are represented using self-similarity matrices(SSMs) corresponding to time-ordered point clouds in feature spaces of each of these data sources; we note that these feature spaces can be of entirely different scale and dimensionality.

A fused similarity template is then derived from the modality-specific SSMs using a technique called similarity network fusion (SNF). We investigate pipelines using SNF as both an upstream (feature-level) and a downstream (ranking-level) fusion technique. Multiscale geometric features of this template are then extracted using a recently-developed technique called the scattering transform, and these features are then used to differentiate speech sequences. This method outperforms unsupervised techniques which operate directly on the raw data, and it also outperforms stovepiped methods which operate on SSMs separately derived from the distinct modalities. The benefits of this method become even more apparent as the simulated peak signal to noise ratio decreases.

TABLE OF CONTENTS

1. INTRODUCTION

This paper suggests a very general and entirely unsupervised heterogeneous sensor fusion pipeline that consists of three geometry-based techniques. We demonstrate this pipeline on a well-studied digit recognition problem (Section 2) in audio-visual fusion, but we claim that this will be broadly useful in any fusion problem where each sensor produces a time series of arbitrary dimension.

Time series from each sensor are summarized using a two-dimensional construct called a *self-similarity matrix* (SSM, Section 4). A time series $f(t_1), \ldots, f(t_n)$ of arbitrary dimension produces an SSM whose (i, j) entry encodes the similarity between $f(t_i)$ and $f(t_j)$. The two time series considered in this work (Section 3) correspond to lip pixel frames which are resized to 25x25 pixels from the video sensor (treated as a 625 dimensional Euclidean space) and MFCC time series of dimension 20 from the audio sensor, but we emphasize that the techniques are general enough to handle time series of any form from any sensor modalities. This two-dimensional representation of time series from disparate sensor streams enables useful comparisons, facilitating many applications of SSMs, including to cover song retrieval via MFCC features [22], cross-modal comparison [20] and action recognition [10].

Upstream fusion of SSMs arising from two or more modalities can then be performed (Section 5) using *similarity network fusion* (SNF, [24], [25]), a random-walk-based technique which is designed to create an SSM which combines the strengths of the individual matrices. SNF has been applied to different pre-processed modalities arising from musical audio [21] and to improving object level comparisons between 2D shapes[24], cancer phenotypes [25], [8], and image collections [8], but to our knowledge, this is the first paper that does so across audio and video modalities for individual objects. It is also possible (Section 7) to apply SNF to SSMs defined on the *object level* rather than at the feature level, leading to a downstream fusion technique.

Summary features are then extracted from the fused SSM using the *scattering transform* (Section 6). This wavelet-based technique ([13], [5]) uses the architecture of a convolution neural network, but without any supervision, to extract a hierarchy of geometric features from images (including fused and/or non-fused SSMs) in a manner that is provably robust to deformations and preserves multi-scale frequency information. The scattering transform has been applied in several venues, for example to texture discrimination [17].

We advocate combining these techniques into an extremely general pipeline (Figure 2) that is entirely unsupervised (i.e, no training data required), does not require domain-specific models, and can handle pre-processed input of any form. The benefits of this pipeline are demonstrated via several experiments with increasing levels of simulated noise (Section 8), and we also explore different combinations of the techniques (e.g, using the scattering transform directly on the modality-specific SSMs rather than fusing).

978-1-5386-2014-4/18/$31.00

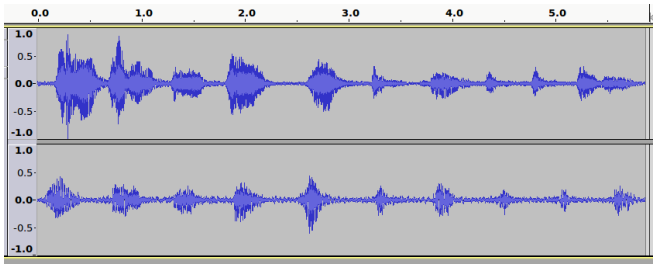


Figure 1. Even after uniformly scaling sequences from two speakers, the raw audio signals do not align perfectly.

There have been countless papers ([2] is a good survey) advocating different approaches to audio-visual fusion, and some which tackle the specific problem of digit recognition. The vast majority of these techniques are supervised, requiring labeled examples in order to build a model. For instance, in a widely-used and excellent method [16] which combines Canonical Correlation Analysis (CCA) with Hidden Markov Models (HMMs), the CCA subspace in which the modalities are most correlated needs to be learned on a training set before applying the method to new examples, and a different HMM must also be trained on each class. Many of the most recent and very successful methods [9] are based on deep learning and thus require very large numbers of labeled examples. Our proposed approach, by contrast, is completely unsupervised; that is, we rely on labeled examples only to *evaluate* our models, rather than using them in *training* to generate the features used by our models.

ACKNOWLEDGMENTS

All three authors were partially supported by the Air Force Office of Scientific Research under grant AFOSR FA8750-17-C-0054, as well as the National Science Foundation under the BIGDATA grant, # DMS 144749. We are very grateful to Dr. Erik Blasch (AFOSR) and Drs. Peter Zulch and Jeffrey Hudack (AFRL) for motivating discussions.

2. RECOGNITION PROBLEM

We demonstrate our methods via a recently-released but already well-studied audiovisual database [1], which has speakers making utterances while recorded simultaneously by several cameras and by audio. We focus on the "Digits" portion of the database, which contains 51 distinct speakers uttering digit strings of length ten (e.g, 1735162667). There are ten distinct strings, with each one uttered three distinct times by each speaker; hence there are $51 \cdot 3 = 153$ examples of each string.

This dataset is particularly interesting because the two modalities capture different aspects of speech (e.g, 'p' sounds cannot be distinguished from 'b' sounds by video). There are also no obvious correlations between the two modalities, unlike that between, say, nadir-measured speed and horizontally-measured doppler as in some of our earlier work [20]. Furthermore, there is quite a bit of variation for a single digit string when uttered by different speakers (see Figure 1), and indeed sometimes even by the same speaker.

Evaluation Methods

Here we rigorously define our problem, as well as the metrics used to evaluate success of the different proposed pipelines. Let $\mathcal{D}$ be the collection of all utterances of all 10-digit strings, decomposed as the disjoint union of sets D_i, each of which contains the 153 distinct utterances of the ith string, for a total of 1530 strings in the entire database. Given any notion of distance μ on $\mathcal{D}$ and a specific string s, the strings in $\mathcal{D}$ can be ranked in increasing order of distance from s. The goal is to create a μ where, for each such s, if $s \in D_i$, then the other strings in D_i will be ranked higher than the strings not in D_i. Success in achieving this goal for a specific s is measured by a *precision recall curve*. More precisely, s is pulled out of the database, and the remaining 1529 strings are put in ranked order by μ. The ranks of the remaining 152 digits in the digit class of s are used to construct the precision recall curve as follows; there are 152 points in the graph, with "recall" r_i on the x-axis equal to $r_1 = 1/152$, $r_2 = 2/152$, ..., $r_{152} = 1.0$, and the corresponding "precision" values p_i on the y-axis equal to the proportion of items in the correct class to all items we've gone through by the time we reach the i^{th} correct item. For instance, if the tenth item occurs at rank 100 in the list, then the recall $r_{10} = 10/152$, and the precision is $p_{10} = 10/100 = 0.1$. The area under the precision-recall curve is referred to as the *mean average precision (MAP)*. A MAP of 1.0 is considered perfect performance, while MAP values closer to zero indicate bad performance. To report a summary statistic for all of μ, we average precision recall curves for all strings in the database $\mathcal{D}$, and compute the corresponding MAP.

Figure 10 displays precision-recall curves and MAPs for the different ranking mechanisms explored in this paper (which we will explain in subsequent sections), and Figure 11 explores the impact of noise, which we will explain more in Section 8. We also compute precision recall curves and MAPs for two other ways of partitioning the data into distinct classes. First, we decompose $\mathcal{D}$ into 51 different classes of size 30, where each class contains all sequences uttered by a particular speaker (i.e. each PR curve has 29 points). Figure 12 shows MAPs for this case over different noise levels. Finally, we explore a finer grained sorting into the intersection of speaker and digit class, for a total of 510 different classes of size 3 (i.e. each PR curve has 3 points), and figure 13 shows MAPs for this case.

3. DATA PREPROCESSING

We are now ready to explain our pipeline for comparing the 10 digit strings in more detail. First, we apply some simple preprocessing to the raw audio and video for each string, which is very similar to preprocessing done by the authors of [16]. For the video, we use pre-extracted lip regions for each speaker that are all resized uniformly to 25x25 grayscale pixels, so that size variations in the lips between subjects due to anatomy and camera position are factored out. Unlike [16], who perform a zigzagged/shrunken DCT (as in the JPEG standard) to further clean this up, we simply use the raw pixels. Each frame in the video then amounts to one sample in a time series which takes values in 625 Euclidean dimensions. For the audio, we resample it to 22050hz and compute 20 MFCC coefficients [4], using a window size of 4096 and a hop size of 256. As the windows jump by an interval equal to the hop size, this then traces out a time series taking values in a 20 dimensional Euclidean space.

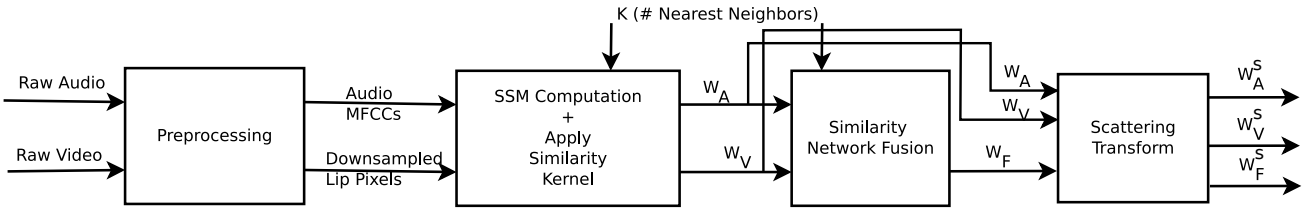


Figure 2. The pipeline for our upstream fusion and data summaries (Sections 3 through 6). Note that the only input parameter is the nearest neighbors K. So this is a nearly non-parametric, unsupervised model. This pipeline can also be easily generalized to handle any number of input features of any type.

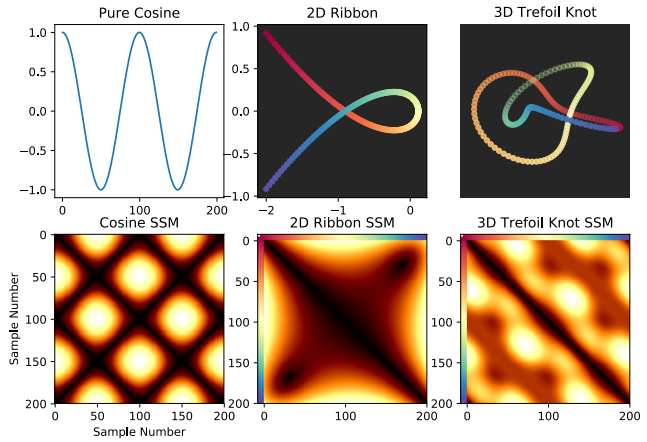


Figure 3. Three examples of self-similarity matrices (SSMs) built on three TOPCs: A 1D cosine, a 2D ribbon, and a 3D knot. Time is on the horizontal axis in the left panel, and is indicated by color in the other two panels. This was Figure 1 in [20].

4. Self-Similarity Matrices

This section provides basic background on Self-similarity Matrices (SSMs), which we advocate as a very useful and general tool for understanding the time evolution of a process in an arbitrary feature space. An earlier paper [20] of the authors used SSMs in the heterogeneous sensor framework, and much of this section is adapted from that paper as well as from the dissertation [23] of the first author.

Suppose that $\gamma : [a, b] \to (M, \rho)$ is a space curve defining a trajectory in some metric space. Such a curve gives rise to a *self-similarity image* (SSI) $D_\gamma : [a, b] \times [a, b] \to \mathbb{R}$ via $D_\gamma(x, y) = \rho(\gamma(x), \gamma(y))$ for all pairs of time points x, y. If we discretize the time domain $a = t_1 < t_2 < \ldots < t_N = b$, then we have a time series or *time-ordered point cloud* (TOPC) $X_1, \ldots, X_N \in M$, where $X_i = \gamma(t_i)$. Then the SSI becomes an $N \times N$ *self-similarity matrix* SSM with $D_{ij} = \rho(X_i, X_j)$. Three notional examples of SSMs appear in Figure 3.

In this paper, we transform each 10-digit string s into SSMs $D_V(s)$ and $D_A(s)$ by employing the preprocessing described in Section 3 and using standard Euclidean distance in R^{625} and R^{20}, respectively. Note that the entries in an SSM can go from zero to infinity, where small entries indicate close proximity. To facilitate the probability-based methods in Section 5, we transform each SSM as follows. Given any TOPC point cloud $X_1, \ldots, X_N$ and any metric ρ we define a *similarity kernel* W as

$$W_{ij} = \exp\left(-\frac{\rho^2(x_i, x_j)}{\sigma_{ij}}\right) \tag{1}$$

This is a similarity measure rather than a distance, as points which are far have a score close to 0, and points which are close have a score closer to 1. Often, σ_{ij} is set to a constant, but a smarter choice is to autotune it based on the average distance to the nearest neighbors of x_i and x_j. In particular, we can follow [25] and set:

$$\begin{aligned}\sigma_{ij}^{\kappa} &= \frac{\beta}{3}\Big(\frac{1}{\kappa N}\Big(\sum_{k \in N^\kappa(i)} \rho(x_i, x_k)\Big) \\ &\quad + \Big(\frac{1}{\kappa N}\sum_{k \in N^\kappa(j)} \rho(x_j, x_k)\Big) + \rho(x_i, x_j)\Big),\end{aligned}$$

where κ is the proportion of nearest neighbors taken (we use $\kappa = 0.1$ in this work), $N^\kappa(i)$ refers to the κN nearest neighbors of x_i, and β is a parameter that can be tweaked (usually in the range $[0.3, 0.8]$). Of course, kernels other than the Gaussian are possible, but we find this works well in practice for our applications.

Applying this procedure to $D_A(s)$ and $D_V(s)$ results in new SSMS, $W_A(s)$ and $W_V(s)$. Examples of each type appear in the first two columns of Figure 4. One can of course also compute an SSM directly from the raw audio data, treated as a simple one-dimensional time series. Figure 5 compares such an SSM on the right with the MFCC-aided SSM (left), and indicates that the latter picks up on more meaningful structure.

5. Upstream Similarity Network Fusion

Sections 3 and 4 describe how to transform a digit string into two SSMs, with W_A derived from audio and W_V derived from video. In general, one is often faced with the situation where one has two or more similarity measures defined on the same finite ordered set. The technique of *similarity network fusion* (SNF, [24], [25]) takes the SSMs of these similarity measures and outputs a single fused SSM which is meant to leverage the strengths of each individual SSM.

A notional example appears in Figure 6. We imagine that we have a TOPC consisting of three distinct clusters with 100 points each, and we noisily sample these clusters three separate times. Each time we do so, the ℓ_2 distance in the plane gives a similarity measure. The SSMs corresponding to

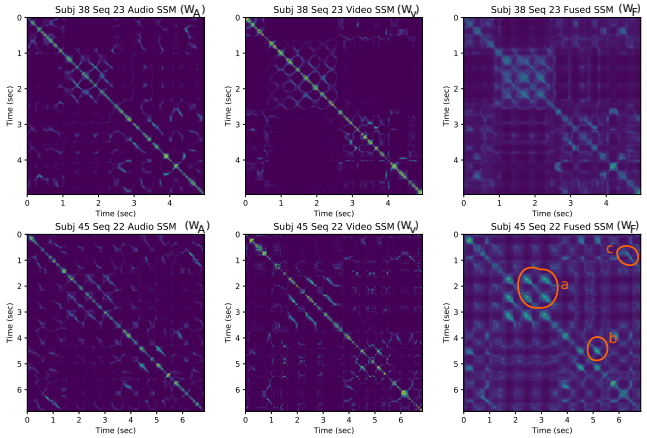


Figure 4. SSMs for the digit sequence "9 7 4 4 4 3 5 5 8 7" spoken by two different speakers. The SSMs for different speakers are on each row, the first column shows SSMs for audio, the second column shows SSMs for video, and the third column shows the fused SSMs. Bright means similar and dark means dissimilar. The repetitions of the two 4s are circled in region a in the bottom speaker, the repetitions of the 5 digit is circled in region b, and the repetition of the digit "7" is circled in region c. These structures are visible in both speakers.

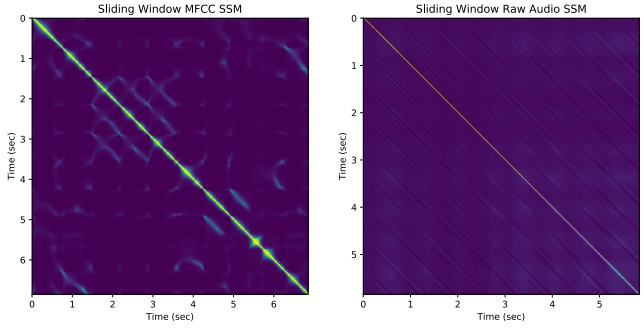


Figure 5. SSMs for audio on the digit sequence "9 7 4 4 4 3 5 5 8 7" as summarized by MFCC and as summarized by raw samples. Structure, such as the repeating 4s, 5s, and 7s, as in Figure 4), is not visible in the raw samples.

these three distinct similarity measures appear in the top row of the figure. While each measure is likely to see a pair of points belonging to a single cluster as more similar to each other (the yellow speckles along the diagonal) than a pair of points from distinct clusters, this is certainly not always the case (the many blue gaps among the yellow speckles). A similar critique holds for the simple average of the three SSMs. On the other hand, solid cluster membership is much more apparent in the SSM produced by the SNF algorithm. The third column of Figure 4 shows fused SSMs resulting from audio and video SSMs in the digits dataset.

We now give some technical details of SNF, referring the reader to [24], [25] for a fuller description. Given m different $N \times N$ similarity matrices $W^1, W^2, ..., W^m$ (as in Equation 1), we normalize them into corresponding matrices $P^1, P^2, ..., P^m$ as follows

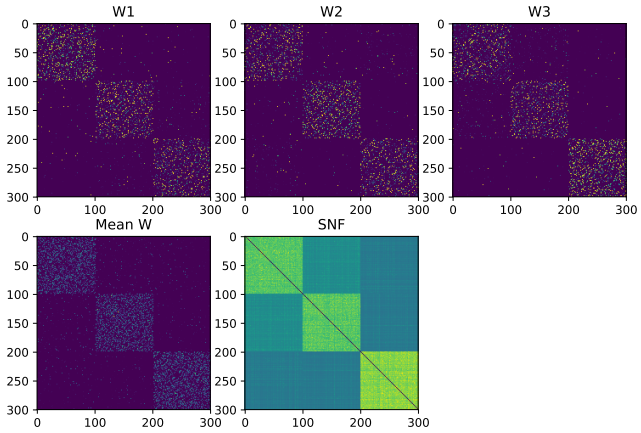


Figure 6. A notional example of similarity network fusion from three different (top row) noisy measurements of 3 simulated clusters. Simple averaging (bottom row, left) of the similarities is far inferior to similarity network fusion (bottom row, middle).

$$P^\ell(i,j) = \left\{ \begin{array}{ll} \frac{W^\ell(i,j)}{2\sum_{k\neq i} W^\ell(i,k)} & j \neq i \\ 1/2 & j = i \end{array} \right\} \quad (2)$$

Since each row now sums to 1, each P^ℓ can be thought of as a transition probability matrix on a graph with N vertices. In addition to the P matrices, we also define a "masked" transition probability matrix S^ℓ as follows

$$S^\ell(i,j) = \left\{ \begin{array}{ll} \frac{W^\ell(i,j)}{2\sum_{k\in N_i} W^\ell(i,k)} & j \in N_i \\ 0 & \text{otherwise} \end{array} \right\} \quad (3)$$

where N_i is the κN nearest neighbors of point i (those js with the κN largest similarity values in the i^{th} row of W^ℓ, excluding i). This is similar to P, except that the probabilities outside of the nearest neighborhoods are set to zero, and the remaining entries are reweighted to sum to 1. Given these sets of matrices, SNF updates the P matrices in an iterative fashion, with the following iterations for each $\ell = 1, 2, ..., m$ in sequence

$$P^\ell = S^\ell \times \left(\frac{\sum_{k\neq\ell} P^k}{m-1} \right) \times (S^\ell)^T \quad (4)$$

Intuitively, this can be thought of as one step of a random walk through neighborhoods determined by W^ℓ, using transition probabilities from all of the other Ws. At the end of some specified number of iterations (we use 20), the final similarity matrix can be obtained as $(\sum_{k=1}^m P^k)/m$. Note that for our pipeline so far, we only have $m = 2$ (W_A for audio and W_V for video), though in Section 7, we will show a case where $m = 3$.

Due to the matrix multiplication in Equation 4, the time complexity of this algorithm is $O(N^3)$ for an $N \times N$ similarity matrix, though this can be mitigated with a sparse matrix for S. This is not a computational bottleneck for the short sequences under consideration in this specific application.

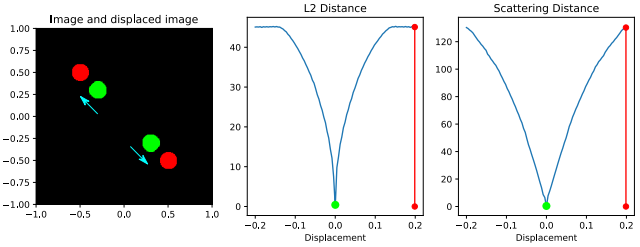


Figure 7. An example of raw L2 versus the scattering transform on an image of two blobs. The initial blobs are drawn in green and an example of displaced blobs are drawn in red. The cyan arrow shows the positive direction of displacement.

Uniformly Rescaling SSMs

Note that SNF requires all of the W matrices to be the same size and in correspondence; that is, in the case of audio/video SSM fusion for a particular digit sequence by a particular speaker, the i^th row of W_A needs to correspond to the same point in time as the i^th row of W_V. Hence, we simply resize W_A and W_V to a common dimension (256×256) with image interpolation. Furthermore, a common dimension for all SSMs allows us to compare across digit sequences which have different lengths in time, which is important since even the same speaker is unlikely to say the same sequence in exactly the same amount of time during different runs.

6. Scattering Transform

Once W_A and W_V have been resized to the same dimension, it is possible to compare instances of each to each other with an L2 norm, otherwise known as a matrix Frobenius norm. That is, the distance between two $N \times N$ matrices A and B could be defined as

$$||A - B||_2 = \sqrt{\sum_{i=1}^{N}\sum_{j=1}^{N}(A_{ij} - B_{ij})^2} \quad (5)$$

For sequences which unfold at the same rate up to a uniform scale, this is a good distance to use. However, in practice, it is unlikely that two runs of the same digit sequence will line up exactly, even after a uniform rescaling. Instead, there are often local delays, or "time warps," that will cause them to be out of sync, and which will induce local perturbations in the SSMs[2]. Unfortunately, when small, high frequency details, such as those in SSMs (Figure 4), are perturbed slightly spatially, the L2 norm can be unstable. Figure 7 shows a synthetic example of this phenomenon. In general, one can show this instability mathematically by applying the 2D Fourier transform to the matrix, which is an isometry, and noticing that high frequency bins change dramatically for any small change in spatial position [6].

Since this instability only occurs at fine scales, one could instead try applying an orthogonal 2D wavelet transform [14] to the SSMs, which, unlike the Fourier Transform, is spatially localized in a hierarchical fashion. However, the wavelet transform is also an isometry, so the same instabilities are still present; this amounts to "bin hopping" of the wavelet coefficients at the fine scales, which is a problem that also plagues histograms compared with Euclidean distance. Motivated by this problem, Mallat [13] devised a nonlinear alternative to the wavelet transform known as the *scattering transform*, which is still hierarchical, but which is stable. Like a wavelet transform, a particular scattering transform starts with a *mother wavelet* $\psi(u, v)$ and its corresponding *scaling function*, or lowpass filter $\phi(u, v)$. However, the scattering transform insists that $\psi(u, v)$ be complex-valued, and in the case of 2D images, it is *directional*, so we index it with a direction $\psi_\gamma(u, v)$, where $\gamma = (\gamma_x, \gamma_y), \gamma_x^2 + \gamma_y^2 = 1$. Finally, the mother wavelet can be scaled in a dyadic fashion, so that

$$\psi_{\gamma,s}(u, v) = 2^{-2s}\psi_\gamma(u/2^s, v/2^s) \quad (6)$$

Then, given an 2D image $I(u, v)$, the level 0 scattering coefficients $S^0(u, v)$ are simply a lowpass filter

$$S^0(u, v) = I * \phi(u, v) \quad (7)$$

The level 1 scattering coefficients $S^1(u, v)$, on the other hand, are computed for each direction γ_i as follows

$$S^1_{\gamma_i}(u, v) = |I * \psi_{\gamma_i,0}(u, v)| * \phi(u, v) \quad (8)$$

where $|.|$ is the complex modulus (absolute value), and the level 2 scattering coefficients $S^2(u, v)$ are computed for each *pair* of directions γ_i, γ_j at scales 0 and 1, respectively, as

$$S^2_{\gamma_i,\gamma_j}(u, v) = ||I * \psi_{\gamma_i,0}(u, v)| * \psi_{\gamma_j,1}(u, v)| * \phi(u, v) \quad (9)$$

With this setup, the level 2 scattering coefficients compute interactions between different directions at different scales, which allows them to pick up on higher order information not possible with an ordinary Fourier or wavelet transform. To continue to even higher order scattering coefficients, one continues this pattern of convolving with wavelets at coarser and coarser scales, always followed by a complex modulus (the nonlinear element), and finished with a lowpass filter (the choice of a sequence of wavelets in this hierarchy is referred to as a "scattering path"). Hence, the architecture of the scattering transform is similar to the architecture of a deep convolutional neural network [11], [12], although the weights are not learned but instead fixed based on the mother wavelet. This crucial difference accounts for the stability of the scattering transform versus the empirical instability of deep neural networks that exposes them to adversarial attack in certain contexts [18], [15]. It also circumvents the need to learn weights from examples, keeping our pipeline unsupervised. For a more complete description of the 2D scattering transform in the context of texture classification, please refer to [7].

In our case, we use complex Morlet (Gabor) wavelets, which take the form of a complex plane wave attenuated by a Gaussian

$$\psi_\gamma(u, v) = e^{i\gamma\cdot(u,v)}e^{-(u^2+v^2)/\sigma^2} \quad (10)$$

[2]The authors of [19] have made a similar observation for time series in dynamical systems and have used the 1D scattering transform to ameliorate this

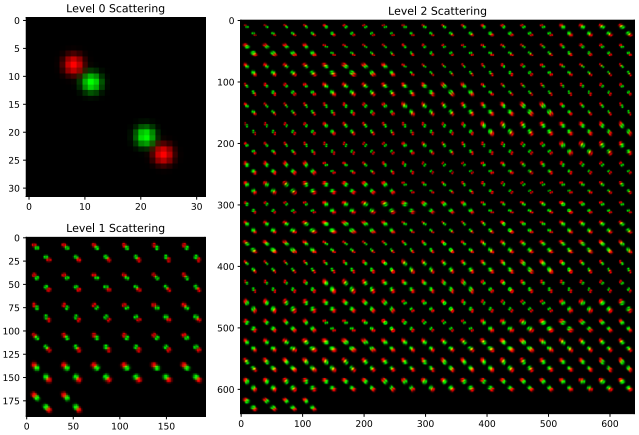


Figure 8. An example of the zeroth, first, and second order scattering coefficients for the blob image and the shifted blob image in Figure 7. Although they do not overlap in the original images, some of their first and second order scattering coefficients do overlap slightly (the perturbation has been "smoothed out"), leading to a non-saturated distance.

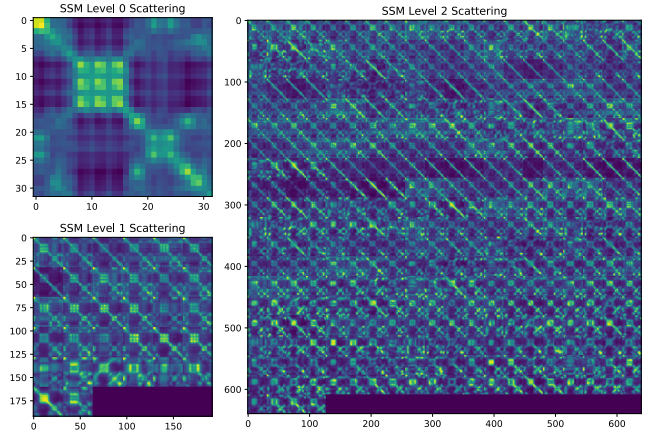


Figure 9. An example of the zeroth, first, and second order scattering coefficients for the fused SSM for subject 45 sequence 22 (bottom right of Figure 4).

with corresponding scaling functions which are the Gaussians $\phi(u,v) = \exp(-(u^2+v^2)/\sigma^2)$. We resize our images to a 256×256 resolution (65536 pixels), and we take 32 directions equally spaced between 0 and π (since the interval $[\pi, 2\pi)$ is redundant in directions with $[0,\pi)$). We then perform two levels of scattering and downsample each scattering path to a 32×32 resolution after lowpass filtering. This leaves us with $k = 32^2(1 + 32 + 32^2) = 1082368$ scattering coefficients (a roughly 16x increase in data size, but at a gain of stability). The scattering distance can then be taken as the Euclidean distance in this k-dimensional Euclidean space, and this is provably stable [13]. As an example, Figure 7 shows distances of the scattering transform versus straight L2 distances between images of blobs as the blobs are displaced along a line. Note that the L2 and scattering distance are both near zero for very small displacements, but the L2 distance maxes out once the blobs no longer overlap at an absolute displacement of 0.1, the radius of the blob. By contrast, the scattering transform distance continues to be sensitive to changes even beyond the support of the blob, hence demonstrating its stability to small deformations.

In our pipeline, we use the scattering transform exclusively on self-similarity images. Figure 9 shows an example of the scattering transform for the fused SSM for subject 45 sequence 22 (bottom right of Figure 4). Finally, note that, like the 2D wavelet transform, the scattering transform on our $N \times N$ images has time complexity $O(N^2 \log(N))$. Since N is small for this application, this is not a computational bottleneck for the scattering transform on individual SSMs.

7. Downstream Similarity Network Fusion

So far, all of the steps we have described can be performed upstream: that is, before any ranking decisions have been made. Figure 2 shows the flow from step to step. However, if we have access to N examples of digit sequences already and we perform all pairwise comparisons between them, we can perform one additional, optional step to improve classification, even without using any labels on these sequences. Let μ_1, μ_2 and μ_3 be three different $N \times N$ matrices measuring all of the pairwise similarities between digits, as measured by some path in our upstream pipeline. For instance we can use L2 on W_A (μ_1), W_V (μ_2), W_F (μ_3), or we can use L2 on their scattering counterparts W_A^S (μ_1), W_V^S (μ_2), or W_F^S (μ_3). Note that μ_1, μ_2, μ_3 are themselves SSMs, but *at the object level*. Because they are SSMs, we can apply SNF to them to potentially improve the ensuing rankings. We refer to this as *downstream similarity network fusion*, as it happens at the object level after all upstream features have been computed, and we do indeed see that it improves results in some cases in Section 8. Note that this also works with only two different similarity measures, though we see advantages to using all three at our disposal within L2 or scattering.

8. Experiments and Results

This section puts the geometric tools described in Section 4-7 together into several different pipelines, and then evaluates these pipelines on the dataset described in Section 2. As stated above, each pipeline creates a metric μ on the set of digit strings $\mathcal{D}$, and here they do so by transforming each string into a feature vector in a suitable Euclidean space (either raw SSM entries or scattering coefficients on SSMs) and then using the standard ℓ_2 metric between pairs of feature vectors in that space.

The method we advocate (Figure 2) starts by extracting one audio and one video time series from each string as described in Section 3. These time series are transformed (Section 4) into similarity matrices $W_A(s)$ and $W_V(s)$, which are then fused (Section 5) to produce a single SSM $W_F(s)$. Finally, the scattering transform (Section 6) is applied to $W_F(s)$ to produce a sequence of scattering coefficients which form the needed feature vector. This method is referred to as FusedScatter in the test results below.

We also explore several other pipelines. FusedL2 stops at the third block in Figure 2, stacking the entries of the matrix $W_F(s)$ to form the feature vector. Similarity, AudioL2 (resp., VideoL2) outputs $W_A(s)$ (resp., $W_V(s)$) as the feature vector. AudioScatter (resp., VideoScatter) uses only the audio (resp., video) time series, transforms it into $W_A(s)$ (resp., $W_V(s)$), and then directly extracts scattering coefficients without any similarity network fusion. Finally, we can perform late fusion

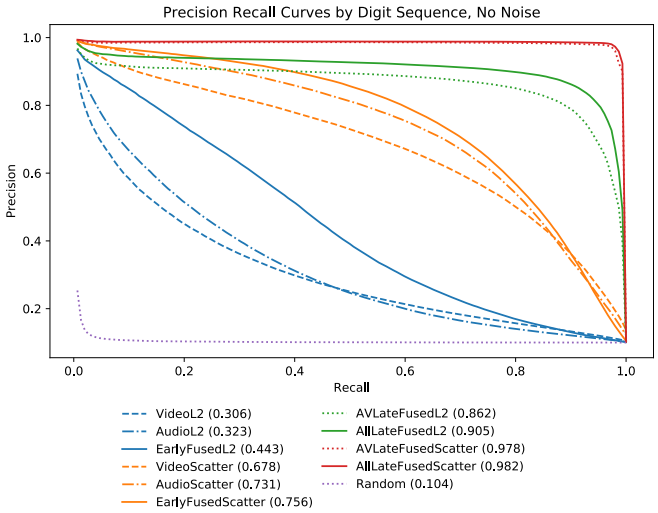


Figure 10. Precision-Recall curves for each pipeline under zero noise. Mean Average Precision (MAP) shown in parentheses.

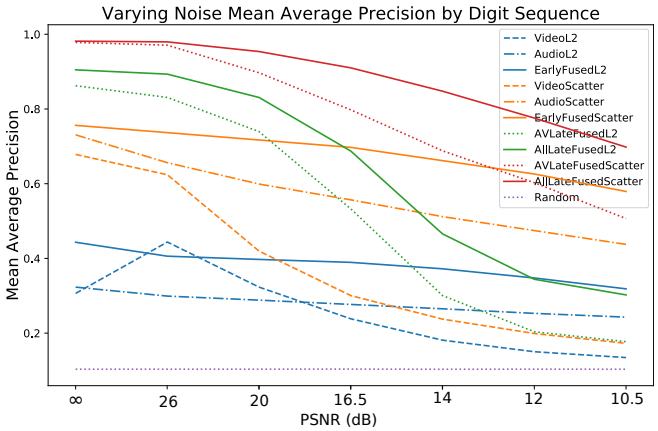


Figure 11. The impact of noise on our pipelines when classifying by digit sequence (regardless of speaker). For each pipeline, mean average precision is plotted against level of noise.

(Section 7) to AudioL2 and VideoL2, or AudioL2, VideoL2, and FusedL2, and similarly for AudioScatter and VideoScatter and AudioScatter, VideoScatter, and FusedScatter. We call these AVLateFusedL2, AllFusedL2, AVLateFusedScatter, and AllFusedScatter, respectively.

Precision-recall curves for each pipeline are shown in Figure 10. The benefits of using the scattering transform rather than operating directly on the SSMs is clear. The advantage of fusion before scattering, while less striking, is also apparent. To probe this further, we ran a second series of experiments, where simulated noise was added to both the raw video and raw audio data before pre-processing. Figure 11 shows the results for digit classification plotted against the peak signal to noise ratio (pSNR) in dB (∞ is no noise). Recall that the pSNR of a signal x is defined as $(20 \log_{10} \max |x|)/\sqrt{MSE(x)}$. At each level of noise, we compute a P-R curve, but only the MAP values are shown. The advantage of fusion before scattering increases in the presence of noise. Furthermore, downstream SNF yields near perfect performance at low levels of noise, and continues to dominate all other pipelines over all noise levels.

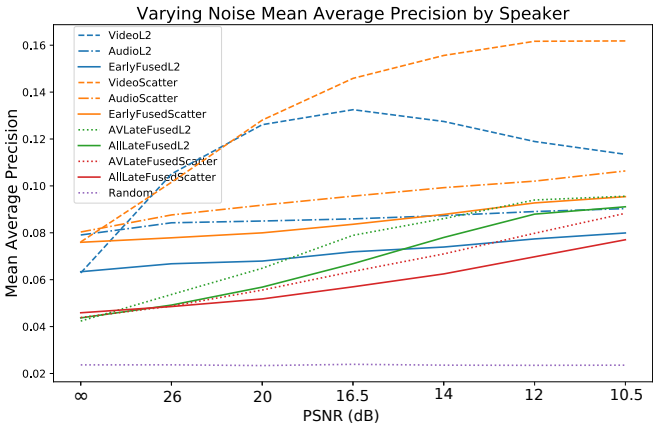


Figure 12. The impact of noise on our pipelines when classifying by speaker (ignoring the digit sequence)

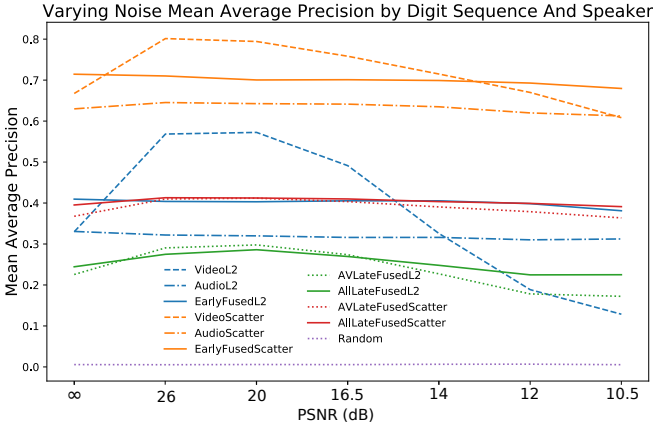


Figure 13. The impact of noise on our pipelines when classifying by speaker *and* by digit sequence that they uttered.

Finally, we repeat our noise experiment under different classification schemes to get an even better idea of what SSMs retain. Based on our experience with cover song analysis [22], we originally hypothesized that SSMs would be approximately invariant to speaker and hence do a particularly poor job of distinguishing speakers from one another. We do indeed get poor results when attempting to classify which of the 51 speakers is uttering a particular digit, as shown in Figure 12. However, when we drill down even further and attempt to classify which speaker *and* which digit sequence is being uttered, we do surprisingly well, as shown in Figure 13. The high MAP values there are particularly striking, as there are 510 classes between which to disambiguate, and there are only 3 examples per class; random guessing only has a MAP of about 0.002. Figure 13 does, however, highlight one potential pitfall of downstream SNF, which actually degrades results in all cases. We believe this is because there are so few examples per class, that the random walk smooths out these finer details. Hence, we recommend downstream SNF only when one has access to a rich set of examples, with enough examples in each class.

9. Conclusions

This paper discussed three geometric techniques–Self-Similarity Matrices, Similarity Network Fusion, and the Scat-

tering Transform–and used them together for the first time (to our knowledge) in the fusion of multi-modal time series. As described above, self-similarity matrices (SSMs) permit the comparison of disparate time series on a common footing, similarity network fusion uses graph-diffusion methods to produce a single SSM that combines the best features of two or more modality-specific SSMs, and the scattering transform extracts multi-scale features from SSMs in a provably robust way.

We constructed several pipelines, involving both early feature-level and late ranking-level fusion, and demonstrated their benefits on a well-known dataset. The pipeline performance on digit-string recognition was particularly striking, especially given the unsupervised nature of our methodology. The specific tests run in this paper used pre-processed streams of video and audio data, but the pipelines are entirely agnostic to choices of modalities and preprocessing thereof. An important next step is to exploit these methods in other multi-modal contexts, for example to seismic-acoustic fusion [3].

The time complexity of our pipeline is not that bad, as commented on at the end of Sections 5 and 6. However, this only refers to one possible cost of a fusion pipeline within a notional sensor network. Assuming a model where there are individual audio/visual sensors and a fusion center, the SNF computation must be done by the fusion center, and modality-specific SSMs must be transmitted to this center from the local sensors. If transmission is "expensive" (e.g,, there may be bandwidth limitation imposed by a mission or there may be detection risks associated with excessive communication), then sending entire SSMs may not be advisable. Some of our ongoing research efforts attacks this problem, by exploring the construction of effective image compression techniques before transmitting the SSMs.

REFERENCES

[1] I. Anina, Z. Zhou, G. Zhao, and M. Pietikinen. Ouluvs2: A multi-view audiovisual database for non-rigid mouth motion analysis. In *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*, volume 1, pages 1–5, May 2015.

[2] Pradeep K Atrey, M Anwar Hossain, Abdulmotaleb El Saddik, and Mohan S Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia Systems*, 16(6):345–379, nov 2010.

[3] E. P. Blasch, J. Dezert, and P. Valin. Dsmt applied to seismic and acoustic sensor fusion. In *Proceedings of the 2011 IEEE National Aerospace and Electronics Conference (NAECON)*, pages 79–86, July 2011.

[4] Bruce P Bogert, Michael JR Healy, and John W Tukey. The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. In *Proceedings of the symposium on time series analysis*, volume 15, pages 209–243. chapter, 1963.

[5] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1872–1886, Aug 2013.

[6] Joan Bruna. *Scattering representations for recognition*. PhD thesis, Ecole Polytechnique X, 2013.

[7] Joan Bruna and Stephane Mallat. Classification with scattering operators. In *CVPR 2011*.

[8] Ning Chen. Ci-snf: Exploiting contextual information to improve snf based information retrieval. *Information Fusion*, 2018.

[9] Adriana Fernandez-Lopez and Federico M Sukno. Survey on automatic lip-reading in the era of deep learning. *Image and Vision Computing*, 78:53–72, 2018.

[10] Imran N Junejo, Emilie Dexter, Ivan Laptev, and Patrick Perez. View-independent action recognition from temporal self-similarities. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):172–185, 2011.

[11] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[12] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[13] Stephane Mallat. Group invariant scattering. *Communications on Pure and Applied Mathematics*, 65(10):1331–1398.

[14] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[15] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2574–2582, 2016.

[16] M. E. Sargin, Y. Yemez, E. Erzin, and A. M. Tekalp. Audiovisual synchronization and fusion using canonical correlation analysis. *IEEE Transactions on Multimedia*, 9(7):1396–1403, Nov 2007.

[17] Laurent Sifre and Stéphane Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on*

computer vision and pattern recognition, pages 1233–1240, 2013.

[18] Jiawei Su, Danilo Vasconcellos Vargas, and Sakurai Kouichi. One pixel attack for fooling deep neural networks. *arXiv preprint arXiv:1710.08864*, 2017.

[19] Jeremias Sulam, Yaniv Romano, and Ronen Talmon. Dynamical system classification with diffusion embedding for ecg-based person identification. *Signal Processing*, 130:403–411, 2017.

[20] C. J. Tralie, A. Smith, N. Borggren, J. Hineman, P. Bendich, P. Zulch, and J. Harer. Geometric cross-modal comparison of heterogeneous sensor data. In *2018 IEEE Aerospace Conference*, pages 1–10, March 2018.

[21] Christopher J Tralie. Early mfcc and hpcp fusion for robust cover song identification. In *18th International Society for Music Information Retrieval (ISMIR)*, 2017.

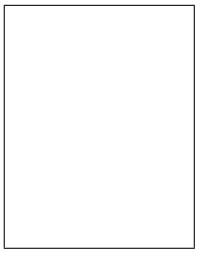
[22] Christopher J Tralie and Paul Bendich. Cover song identification with timbral shape. In *16th International Society for Music Information Retrieval (ISMIR) Conference*, 2015.

[23] Christopher John Tralie. *Geometric Multimedia Time Series*. PhD thesis, Duke University Department of Electrical And Computer Engineering, 2017.

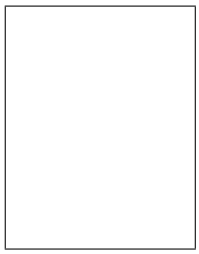
[24] Bo Wang, Jiayan Jiang, Wei Wang, Zhi-Hua Zhou, and Zhuowen Tu. Unsupervised metric fusion by cross diffusion. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2997–3004. IEEE, 2012.

[25] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature methods*, 11(3):333, 2014.

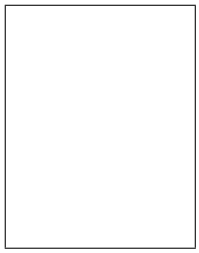
BIOGRAPHY



Christopher J. Tralie *is a data science researcher working in applied geometry and geometric signal processing. His work spans shape-based music structure analysis and cover song identification, video analysis, multimodal time series analysis, and geometry-aided data visualization. He received a B.S.E. from Princeton University in EE in 2011, a master's in ECE at Duke University in 2013, and a Ph.D. in ECE at Duke University in 2017. His Ph.D. was primarily supported by an NSF Graduate Fellowship, and his dissertation is entitled "Geometric Multimedia Time Series." He was also awarded a Bass Instructional Teaching fellowship at Duke University, and he maintains an active interest in pedagogy and outreach. He is currently a postdoctoral associate at the Information Initiative at Duke, sponsored by the department of mathematics.*



Paul Bendich *is Associate Research Professor of Mathematics at Duke University, where he also serves as Associate Director for Curricular Engagement at the Information Initiative (iiD) at Duke. He is also Chief Scientist for GDA He received his Ph.D. in Mathematics from Duke University in 2008, and his B.A. in Physics from Grinnell College in 2001. He was one of the first-generation of students doing the theoretical development of topological methods for data analysis, and has been at the forefront of efforts to properly integrate topological and geometric data analytic methods within classical machine-learning and statistics. He has been PI or co-PI on numerous grants from NSF, and has also contributed significant effort towards grants and contracts from AFOSR, DARPA, DTRA, OSD, and DHS.*



John Harer *is Professor of Mathematics at Duke University, and is CEO and Subject Matter Expert at GDA. He got his Ph.D. in 1979 from the University of California, Berkeley. Since then he has held faculty positions at Columbia University, the Universities of Maryland and Michigan, and at Duke, where he came as full professor in 1992. He has extensive managerial experience, serving for 5 years as chair of the Department of Mathematics and another 5 years as Vice-Provost for Academic Affairs. At Duke he has been PI or co-PI on numerous grants from NSF, NIH, AFOSR, DARPA, DTRA, OSD and NSA. Professor Harer founded GDA in 2012 in order to develop applications of basic research done at Duke.*