Abstract

A central concern for many learning algorithms and sensing systems is how to efficiently store what the algorithm/system has learned. Convolutively Non-negative Matrix Factorization (CNMF) finds parts-based convolutive representations of non-negative data. Convolutively extended NMF have not considered storage efficiency or a side constraint during the learning procedure. We contribute an algorithm, Storable NMF (SNMF), that fuses ideas from the (1) parts-based learning and (2) integer sequence compression literature. SNMF enjoys the merits of both techniques: it retains the good-approximation properties of CNMF while also taking into account the size of the symbol set which is used to express the learned convolutive factors and activations. We demonstrate that SNMF yields a compression ratio ranging from 10:1 up to 20:1, which gives rise to a similar freedom to choose an arbitrary lossy factorization with respect to the distortion constraint of choice. Can CNMF learn a factorization which is: (1) within 1-SBD of the expected CNMF Signal-to-Noise-Ratio and (2) amenable to efficient coding? SNMF Applied to Synthetic Data-set: We report average performance over 100 trials. Each trial was run for 100 iterations. K = 2, FFT = 512, Overlap = 512. SDR is improved by increasing R.

Noise Burst Data-set: Performance comparison NMF vs SNMF (T = 1 – 32).

Results

Hypothesis: Given that (CN)MF is non-unique and non-exact, but typically yields a satisfactory decomposition, we have the freedom to choose an arbitrary lossy factorization with respect to the distortion constraint of choice. Can SNMF learn a factorization which is: (1) within 1-SBD of the expected CNMF Signal-to-Noise-Ratio and (2) amenable to efficient coding? SNMF Applied to Synthetic Data-set: We report average performance over 100 trials. Each trial was run for 100 iterations. K = 2, FFT = 512, Overlap = 512. SDR is improved by increasing R.

Noise Burst Data-set: Performance comparison NMF vs SNMF (T = 1 – 32).

Motivation

Learn acoustic features [7] at the edge of a network that can be transferred between (and stored on) different sensors/networked devices.

Convolutive NMF

- Activates a cascade of T basis functions;
- Learns T = 1 matrices Wt, Wt ∈ IR × N, and H, where H is the average update.

Adaptive Quantization

- Adapt the quantization function’s step-size for each column of Wt and each row H
- Initially I = 1, as convergence, a subset of the positive integers.
- The cardinality of I, is generally minimized as I is indeedly learned.

Dictionary

- Learns NMF to store and transmit NMF decomposition of multivariate data [1, 2].
- Learns NMF to store and transmit NMF decomposition of multivariate data [1, 2].

Storable NMF

- Taking a NLA at convergence to improve compressibility of factors, introduces error indiscriminately (1) denotes rounding and S is the quantization step-size)
- I = \{ m \in \mathbb{N} | w \in M, n \in [n, l] \}
- Is there a better way to pick the set?

Adaptive step-size

- Adapt the step-size for each row of H [9, 10]
- H = [H[n, 1], H[n, 2], ... H[n, T]]
- \Delta H[n, t] = \frac{1}{T} \sum_{i=1}^{T} R \hat{H}[n, r]
- \hat{H}[n, r] = \text{quantize}(H[n, r])
- \text{quantize}(H[n, r])

Quantization

- Choose between SNMF factorizes the matrix V ∈ IR × N into the product of two element-wise integer matrices W ∈ IR × M and H ∈ IR × N and two diagonal matrices diag(ΔW) and diag(ΔH).
- \hat{W} = \sum_{i=1}^{T} \text{diag}(\Delta W) \cdot \hat{w}
- \hat{H} = \sum_{i=1}^{T} \text{diag}(\Delta H) \cdot \hat{h}

Quantization stepsizes:

- \Delta W[r, n] ∈ IR, though real-valued, are few, and are adopted to reduce the cardinality of I. The majority of elements of the factors are integer, W[r, n] = \lfloor \hat{w}[r, n] \rfloor ∈ IR.

Quantization step-size updates:

- Choose between \hat{W} = \sum_{i=1}^{T} \text{diag}(\Delta W) \cdot \hat{w}
- \hat{H} = \sum_{i=1}^{T} \text{diag}(\Delta H) \cdot \hat{h}

NMF | & Monotonic Convergence

- Quantization of elements of Wt may cause the KLD to increase;
- Generally, \Delta W[r, n] ∈ IR, (for the important (big) entries).
- We have freedom here in how we apply element-wise rounding, we desire monotonic convergence. Heuristic: we randomly substitute in a number of new quantization functions until KLD is minimized.

Choose between \hat{W} = \sum_{i=1}^{T} \text{diag}(\Delta W) \cdot \hat{w}
- \hat{H} = \sum_{i=1}^{T} \text{diag}(\Delta H) \cdot \hat{h}

SNMF-parsing-encoding scheme

- Concatenate the columns of W and take transposes to generate a sequence;
- Concatenate the rows of H to generate;
- \hat{H} = ([\hat{H}[1]], [\hat{H}[2]], ... [\hat{H}[R]])

Effects of T on CNMF versus SNMF [7, T = 1 – 32].

Each trial run for 500 iterations; R ≤ 20
- Set Is SNMF is smaller than CNMF;
- Compressibility is greater than CNMF;
- Compressibility is not affected by an increase in T;
- Error introduced by SNMF is reasonable relative to CNMF.

Conclusions

- Element-wise integer constraints are introduced and compression is improved. SNMF combined with a universal loss-less coding scheme is portable and efficiently stored.
- Error introduced by integrating the factors is folded into the CNMF updates. A 10:1 or 20:1 compression ratio the results of a CNMF is possible if the factors are restricted to a finite integer symbol set. This only incurs a distortion loss of 3%.

References