Optimality of Inference in Hierarchical Coding for Distributed Object-Based Representations

Simon Brodeur, Jean Rouat
NECOTIS, Département génie électrique et génie informatique, Université de Sherbrooke, Sherbrooke, Canada
Email: simon.brodeur@usherbrooke.ca, jean.rouat@usherbrooke.ca

***Abstract*—Hierarchical approaches for representation learning have the ability to encode relevant features at multiple scales or levels of abstraction. However, most hierarchical approaches exploit only the last level in the hierarchy, or provide a multiscale representation that holds a significant amount of redundancy. We argue that removing redundancy across the multiple levels of abstraction is important for an efficient representation of compositionality in object-based representations. With the perspective of feature learning as a data compression operation, we propose a new greedy inference algorithm for hierarchical sparse coding. Convolutional matching pursuit with a L^0-norm constraint was used to encode the input signal into compact and non-redundant codes distributed across levels of the hierarchy. Simple and complex synthetic datasets of temporal signals were created to evaluate the encoding efficiency and compare with the theoretical lower bounds on the information rate for those signals. Empirical evidence have shown that the algorithm is able to infer near-optimal codes for simple signals. However, it failed for complex signals with strong overlapping between objects. We explain the inefficiency of convolutional matching pursuit that occurred in such case. This brings new insights about the NP-hard optimization problem related to using L^0-norm constraint in inferring optimally compact and distributed object-based representations.**

I. Introduction

Unsupervised learning is very attractive today because of the large amount of unannotated data available from social media and sensors in intelligent devices (e.g. smartphones, autonomous cars, home automation systems). Manifold learning (embedding) is currently one of the most popular forms of learning a latent, low-dimensional feature map that describes some temporal or spatial properties of the input signal (for review, see [1]). There is, however, very little work on evaluating the quality of those representations based on compression criteria. The goal of the paper is to investigate how the notion of data compression from both theoretical and empirical perspectives can help us design, compare and understand better the learning of embeddings that efficiently represent the input structures. This work is in line with clustering by compression (e.g. [2]) and minimum description length (MDL) approaches (e.g. [3], [4]) to feature learning and coding, which contrasts with discriminative approaches.

Object-based representations are very efficient to encode spatially or temporally independent structures of a signal, and are in fact very common in the real world (e.g. pulse-resonance sounds in audio, visual composition in images, syntactic items in language). Sparse coding (SC) approaches are well adapted to model such kind of signals (e.g. [14]). We suggest that dictionary-based feature learning and representation with sparse coding could be comparable to general lossless data compression algorithms such as LZ77 [5]. Those algorithms achieve compression by finding redundant sequences in the data, storing a single copy in a static or sliding window dictionary and then use a reference to this copy to replace the multiple occurrences in the data. This indexing scheme of the dictionary allows the representation to be sparse and compact, meaning that a single index (or reference to a dictionary element) describes a larger segment of the input sequence. Most prior work on dictionary learning and optimal data compression considers string sequences with discrete symbols (e.g. text documents, DNA sequences), as they are primarily used for file compression (e.g. [5]). We rather consider sequences of continuous values, and propose a new greedy inference algorithm for hierarchical sparse coding. In this work, we focus on the inference process, i.e. finding the optimal sets of sparse coefficients given the signal and learned dictionaries. The goal is to evaluate how the hierarchical and compression frameworks can improve object-based representations.

II. Sparse Coding

Assume a simple linear decomposition $\mathbf{S} = \mathbf{XD}$. The variable $\mathbf{S}$ defines a set of M signals of dimensionality N, i.e. $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_M] \in \Re^{M \times N}$. The variable $\mathbf{D}$ defines a dictionary of K bases of dimensionality N, i.e. $\mathbf{D} = [\mathbf{d}_1, \ldots, \mathbf{d}_K] \in \Re^{K \times N}$. The result is a set of coefficients $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_M] \in \Re^{M \times K}$ that represents the input $\mathbf{S}$. Sparse coding (SC) aims at providing a set of sparse coefficients by the penalization of the L^1 norm of $\mathbf{X}$, as shown in Equation 1 (for review, see [6]). The constant α controls the tradeoff between the reconstruction error and the sparsity of the coefficients.

$$(\mathbf{X}^*, \mathbf{D}^*) = \arg\min_{\mathbf{X},\mathbf{D}} \frac{1}{2}\|\mathbf{S} - \mathbf{XD}\|_2^2 + \alpha\|\mathbf{X}\|_1 \quad (1)$$
$$\text{with constraint} \quad \|\mathbf{d}_k\|_2 = 1 \quad \text{for} \quad k \in \{1, 2, \ldots, K\}$$

Sparse coding allows to compact the energy of the representation into a few high-valued coefficients while letting the remaining coefficients sit near zero. Compression is achieved by limiting the amount of information that needs to be stored to represent those coefficients (e.g. by neglecting most low-valued coefficients). Sparse coding can be applied to sequences of variable lengths (e.g. temporal signals) by changing the

978-1-5090-6026-9/17/$31.00 ©2017 IEEE

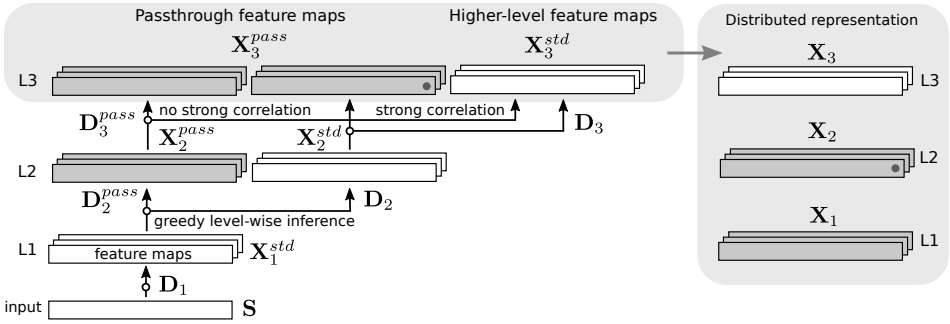


Fig. 1. Greedy level-wise inference scheme for distributed representations in hierarchical sparse coding. When the input $\mathbf{S}$ is convolved with the dictionary $\mathbf{D}_1$ at level $L1$ and greedy inference (shown as white dots) is applied to select sparse activations in the feature maps, the first-level sparse representation $\mathbf{X}_1^{std}$ is created. $\mathbf{X}_1^{std}$ now becomes the input signal for the next level. The second level $L2$ can choose during inference to combine several lower-level activations as modelled by the dictionary $\mathbf{D}_2$ and output information in the higher-level feature maps $\mathbf{X}_2^{std}$, or forward particular sparse activations as is with the passthrough feature maps $\mathbf{X}_2^{pass}$ if sparse activations in $\mathbf{X}_1^{std}$ do not correlate strongly with $\mathbf{D}_2$. The same process is repeated for the next levels in the hierarchy. From the output of level $L3$, both passthrough and higher-level features maps describe the input signal in a distributed way using the non-redundant sets of coefficients $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3\}$. The passthrough feature maps also allow learning over multiple scales. Note that the widths of the dictionaries increase over the hierarchy, so that the considered temporal context increases at each level and thus allows to encode longer time dependencies between sparse activations.

matrix multiplication operator into a discrete convolution [1], as shown in Equation 2. This adds time shift (translation) invariance properties to the dictionary $\mathbf{D}$, and $\mathbf{D} \in \Re^{K \times W}$ can now have temporal length $W < N$ to represent local features in the signal. Moreover, the direct penalization of the L^0-norm of $\mathbf{X}$ can be used instead. This is more adapted to compression, since only the information about non-zero coefficients needs to be stored. The drawback with the L^0-norm is that it makes the optimization problem NP-hard [7].

$$(\mathbf{X}^*, \mathbf{D}^*) = \arg\min_{\mathbf{X},\mathbf{D}} \frac{1}{2}\|\mathbf{S} - \mathbf{X} * \mathbf{D}\|_2^2 + \alpha\|\mathbf{X}\|_0 \quad (2)$$

$$\text{with constraint} \quad \|\mathbf{d}_k\|_2 = 1 \quad \text{for} \quad k \in \{1, 2, \ldots, K\}$$

Learning representations and encoding the input at a too restricted temporal length W does not allow to learn higher-level objects that derive from the composition of multiple local features. This degrades the ability to compress information. Thus, we propose to efficiently incorporate a hierarchical aspect into the convolutional sparse coding framework.

III. Proposed Hierarchical Sparse Coding

Many of the successes in machine learning that led to the sub-field of deep learning is about hierarchical processing. This means stacking in a hierarchy several levels of representations or processing layers to build an abstract representation of the input useful for a particular task (e.g. classification, control). However, many approaches to representation learning (e.g. [1], [8]) only consider the output of the last level and thus lose crucial information about the compositionality of objects. This means that fine information is mixed together with the higher-level, coarse information about objects. It also scales inefficiently with the context, as the number of possible objects typically increases at an exponential rate when the temporal or physical context is made larger. Some approaches (e.g. [9]) aggregate encodings at multiple independent scales prior to classification, but disregard the redundancy across the scales. An efficient hierarchical processing means encoding objects at the level at which they naturally appear, rather than re-encode those to propagate the information up in the hierarchy. This is similar to the multiresolution decomposition scheme of the discrete wavelet transform [10], except that hierarchical sparse coding is not constrained to the time-frequency domain analysis. This avoids the tradeoff between time and frequency resolutions. Hierarchical sparse coding also contrasts with other methods such as multistage vector quantization (e.g. [11]), which encode information maximally at each level and only pass the residual information (i.e. the reconstruction error) to the next level in the hierarchy. These methods do not have the ability to learn high-level objects, since the objects are encoded into their most basic constituents at the very first levels, and the residual energy decreases rapidly towards zero. This is because low-level constituents are typically easier to learn and encode.

The proposed method for hierarchical processing in a sparse coding framework is described in Equation 3. In this hierarchical decomposition scheme, the reconstruction is based on the sets of coefficients $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_L\}$ from L levels, with linear composition across levels based on their respective dictionaries $\{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_L\}$. The sets of coefficients at all levels do not share redundant information, which makes the representation compact in terms of data compression. Note that the dictionaries are now multidimensional arrays (tensors) of rank 3, i.e. $\mathbf{D}_l \in \Re^{K_l \times W_l \times K_{l-1}}$. The dictionary at level l defines a set of K_l bases (or convolution filters) of dimensionality $W_l \times K_{l-1}$, where W_l is the temporal length of the bases and K_{l-1} is the number of features from the previous level. The variable $\mathbf{S}$ now defines a (temporal) sequence of length N_t, with features vectors of dimensionality N_s, i.e. $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_{N_t}] \in \Re^{N_t \times N_s}$. The set of coefficients $\mathbf{X}_l$ at level l is a multidimensional array of rank 3, i.e. $\mathbf{X}_l \in \Re^{N_t \times K_{l-1} \times K_l}$, that are also called feature maps.

[1]The definition of convolution from machine learning is used, where the filter is not time-reversed. This convolution is equal to a cross-correlation.

$$\boldsymbol{\Theta} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_L,\ \mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_L\}$$

$$\boldsymbol{\Theta}^* = \arg\min_{\boldsymbol{\Theta}} \frac{1}{2}\left\| \mathbf{S} - \sum_{l=1}^{L} \left(\mathbf{X}_l * \left(\prod_{k=1}^{l}{}^{*}\ \mathbf{D}_k \right) \right) \right\|_2^2 + \alpha \sum_{l=1}^{L} \|\mathbf{X}_l\|_0 \tag{3}$$

with constraint $\|\mathbf{d}_{k,l}\|_2 = 1$ for $k \in \{1, 2, \ldots, K\}$, $l \in \{1, 2, \ldots, L\}$

and operator $\prod^*$ as a composition of convolutions:

$$\prod_{l=1}^{L}{}^{*}\ \mathbf{D}_l = \mathbf{D}_1 * \mathbf{D}_2 \cdots * \mathbf{D}_L$$

A. Greedy Inference for Hierarchical Sparse Coding

The hierarchical inference scheme is illustrated in figure 1. We extended convolutional matching pursuit (e.g. [14]) and applied it to the hierarchy in a greedy level-wise manner. At higher levels of the hierarchy (i.e. $l > 1$), we introduce two types of feature maps defined by $\mathbf{X}_l^{pass}$ and $\mathbf{X}_l^{std}$: passthrough features maps from the previous levels, and standard feature maps obtained directly at level l from the decomposition with dictionary $\mathbf{D}_l$. The passthrough features maps allow each level to forward to the next level the sparse activations that do not correlate well with the bases of dictionary $\mathbf{D}_l$. This can be implemented by considering during matching pursuit a second dictionary $\mathbf{D}_l^{pass} \in \Re^{K_l^{pass} \times W_l \times K_l^{pass}}$ containing a single Kronecker delta function per dictionary basis so that $\mathbf{S} * \mathbf{D}_l^{pass} = \mathbf{S}$. This corresponds to copying specific sparse activations from the input to output feature maps $\mathbf{X}_l^{pass}$. $K_l^{pass} = \sum_{i=1}^{l-1} K_i$ is the number of aggregated features maps (both passthroughs and standard) from the previous level. To allow learning features over multiple scales, the dictionary $\mathbf{D}_l$ at each level l can also cover the passthrough feature maps of the previous level, thus increasing its dimensionality to $\mathbf{D}_l \in \Re^{K_l \times W_l \times \left(K_{l-1}^{pass} + K_{l-1}\right)}$. The hierarchical sparse coding inference algorithm SOLVE-HSC is described below. Note that more sophisticated matching pursuit methods can be integrated in SOLVE-MP, such as low complexity orthogonal matching pursuit (LoCOMP) [13]. The constant β allows to control the allocation of coefficients in the passthrough feature maps.

function SOLVE-HSC($\mathbf{S}, \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_L\}, \beta$)
- ▷ Solve for coefficients at first-level (no passthrough)
  Note that $\mathbf{X}_1^{pass}$ will be an empty coefficient set
- $\mathbf{X}_1^{pass}, \mathbf{X}_1^{std} \leftarrow$ SOLVE-MP($\mathbf{S}, \mathbf{D}_1, 0$)
- ▷ Iterate for all higher levels (with passthrough)
- **for** $l = 2$ to L **do**
  - ▷ Concatenate passthrough and standard feature maps, then solve with matching pursuit algorithm
  - $\mathbf{X}_l^{pass}, \mathbf{X}_l^{std} \leftarrow$ SOLVE-MP($\mathbf{X}_{l-1}^{pass} \| \mathbf{X}_{l-1}^{std}, \mathbf{D}_l, \beta$)
- **end for**
- ▷ Extract output coefficients sets from level L
- $\mathbf{X}_1, \mathbf{X}_2, \cdots \mathbf{X}_{L-1} \leftarrow$ SPLIT($\mathbf{X}_L^{pass}$)
- $\mathbf{X}_L \leftarrow \mathbf{X}_L^{std}$
- ▷ Calculate the overall residual on the input
- $\mathbf{R} \leftarrow S - \sum_{l=1}^{L} \left(\mathbf{X}_l * \left(\prod_{k=1}^{l}{}^{*}\ \mathbf{D}_k \right) \right)$
- **return** $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_L\}$, $\mathbf{R}$

end function

function SOLVE-MP($\mathbf{S}, \mathbf{D}, \beta$)
- ▷ Create a passthrough dictionary such that $\mathbf{S} * \mathbf{D}^{pass} = \mathbf{S}$
- $\mathbf{D}^{pass} \leftarrow$ Kronecker delta functions
- $\mathbf{R}_1 \leftarrow \mathbf{S}$ ▷ Initialize residual
- $\mathbf{X}^{pass}, \mathbf{X}^{std} \leftarrow \emptyset$ ▷ Empty coefficient sets
- $n \leftarrow 1$
- ▷ Iterate over the residual until convergence at given SNR
- **while** $10 \log \left(\frac{\|\mathbf{S}\|_F^2}{\|\mathbf{R}_n\|_F^2} \right) < \text{SNR}_{\text{thres}}$ **do**
  - ▷ Calculate convolution with dictionaries
  - $\mathbf{C}^{pass}, \mathbf{C}^{std} \leftarrow \mathbf{R}_n * \mathbf{D}^{pass},\ \mathbf{R}_n * \mathbf{D}$
  - ▷ Select best activated basis for both dictionaries, at respective dictionary index k and center position t
  - $\mathbf{d}_{k,t}^{pass}, \mathbf{d}_{k,t}^{std} \leftarrow \arg\max_{k,t} |\mathbf{C}^{pass}|,\ \arg\max_{k,t} |\mathbf{C}^{std}|$
  - ▷ Compare coefficients from both selected bases
  - **if** $\beta \cdot \left|\mathbf{C}_{k,t}^{pass}\right| > \left|\mathbf{C}_{k,t}^{std}\right|$ **then**
    - ▷ Add coefficient to passthrough set
    - $\mathbf{X}^{pass} \leftarrow \mathbf{X}^{pass} \cup \{\mathbf{C}_{k,t}^{pass}\}$
    - ▷ Remove contribution of selected basis to residual
    - $\mathbf{R}_{n+1} \leftarrow \mathbf{R}_n - \left(\mathbf{C}_{k,t}^{pass} * \mathbf{d}_k^{pass}\right)$
  - **else**
    - ▷ Add coefficient to standard set
    - $\mathbf{X}^{std} \leftarrow \mathbf{X}^{std} \cup \{\mathbf{C}_{k,t}^{std}\}$
    - ▷ Remove contribution of selected basis to residual
    - $\mathbf{R}_{n+1} \leftarrow \mathbf{R}_n - \left(\mathbf{C}_{k,t}^{std} * \mathbf{d}_k^{std}\right)$
  - **end if**
  - $n \leftarrow n + 1$
- **end while**
- **return** $\{\mathbf{X}^{pass}, \mathbf{X}^{std}\}$

end function

IV. Experiment

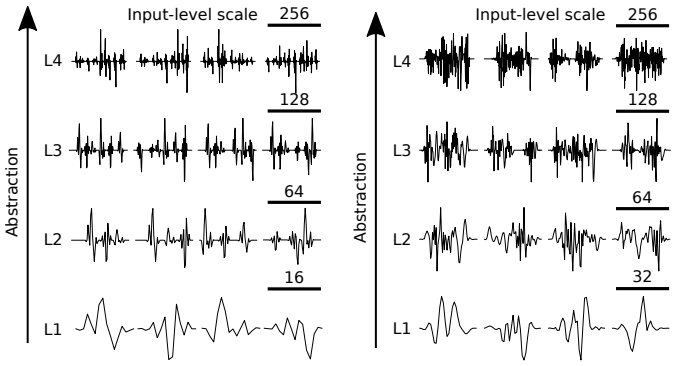
A. Dataset Generation

Multiple synthetic datasets were created to evaluate the hierarchical encoding performance in terms of compression. The first dataset named *dataset-simple* has no overlap between objects in the decomposition and the generated signal when sparse activations are composed in time. The decomposition depends only on the previous level, and thus the dictionaries do not use the passthrough feature maps. The second dataset named *dataset-complex* has no such constraints for overlap, and is useful to assess the encoding of compositionality at multiple scales with complexity closer to real-world signals.

The first-level dictionary bases were generated using 1D Perlin noise [12]. Hanning windowing was used to make the bases localized in time and avoid sharp discontinuities when overlaps occur. Higher-level dictionary bases were generated by uniformly sampling a fixed number N_d^l of activated sub-bases from the previous level. N_d^l corresponds to the decomposition size at level l. The relative random positions of the selected sub-bases were uniformly sampled in time domain within the defined temporal length. The relative random co-

TABLE I
SPECIFICATIONS OF THE GENERATED DATASETS.

Dataset name		L1	L2	L3	L4
dataset-simple	Input-level scales:	16	64	128	256
	Dictionary sizes (K):	4	8	16	32
	Decomposition sizes (N_d):	-	3	2	2
dataset-complex	Input-level scales:	32	64	128	256
	Dictionary sizes (K):	4	8	16	32
	Decomposition sizes (N_d):	-	4	3	3



(a) Dictionaries for *dataset-simple* (b) Dictionaries for *dataset-complex*

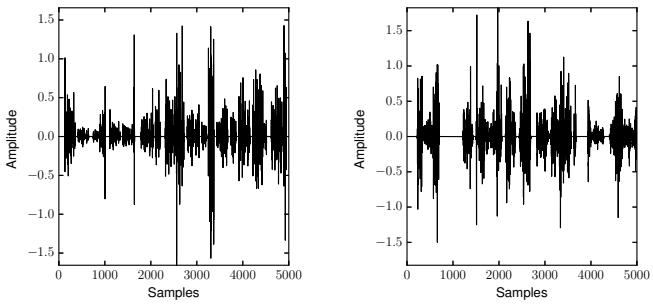
Fig. 2. Example of dictionary bases at each of the 4 levels in the hierarchy that was used to generate the datasets. The bold bars on the right indicate the temporal span of each basis (input-level scale), in samples. As the level increases, the bases become more complex and abstract.

efficients of the selected sub-bases were uniformly sampled from the interval [0.25, 1.0]. Table I gives the specifications of the dictionaries for the two datasets, where the input-level scales, sizes of the dictionaries or decomposition sizes vary across levels. Figure 2 shows some bases of the datasets.

For *dataset-simple*, the temporal signal was generated by concatenating the input-level representations of randomly chosen bases across all levels (with equal probabilities), since there is no overlapping. For *dataset-complex*, the temporal sequence was generated from the dictionaries using independent homogeneous Poisson processes to sample sparse activations (events) in the sets of coefficients $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4\}$. Each dictionary basis was assigned a Poisson process with fixed rate $\lambda = 1.09 \times 10^{-4}$ (in units of events/sample), chosen so that the generated signal was not temporally too compact or sparse. In both databases, random coefficients for the sparse activations were uniformly sampled in the interval [0.25, 4.0]. The length of the generated signals was 100,000 samples, with roughly 650 activations distributed across the various levels. Figure 3 shows segments of the datasets. We implemented the hierarchical sparse coding algorithm and dataset generation in Python using the Numpy and Scipy libraries.

B. Evaluation Method and Results

In the experiments, the optimal (reference) dictionaries that generated the time series were given to the matching pursuit algorithm. This was to test the optimality of inference only, and not dictionary learning. To quantify the performance of the proposed inference algorithm, we evaluated the information



(a) Segment for *dataset-simple* (b) Segment for *dataset-complex*

Fig. 3. Example of segments from the datasets. When averaged over a longer time scale, the information rate needed to encode each signal remains constant.

rate at the output of the encoder when it is limited to a given maximum level $l_{\max}$, and compared with the lower bound $I^{\text{opt}}_{l_{\max}}$ for that level. Equation 4 defines how the optimal information rate $I^{\text{opt}}_{l_{\max}}$ can be calculated for both datasets, based on the known coefficient sets that were used to generate the signals, and the known decomposition sizes. Equation 5 describes how the actual information rate $I^{\text{act}}_{l_{\max}}$ can be calculated from the output coefficients of an encoder at evaluation time. Note that there must be tolerance on the reconstruction signal-to-noise ratio (SNR) during inference. We assumed 40 dB SNR to be sufficient for many types of signals (e.g. images, audio). We used 32 bits floating point precision for the amplitudes of the coefficients (i.e. $I_{fp} = 32$).

$$I^{\text{opt}}_{l_{\max}} = \sum_{l=1}^{l_{\max}} \underbrace{\frac{\|\mathbf{X}_l\|_0}{N_t}}_{\substack{\text{Activation}\\ \text{rate at level } l}} \cdot \underbrace{I_l}_{\substack{\text{Information cost of one}\\ \text{activation at level } l}} + \sum_{l=l_{\max}+1}^{L} \underbrace{\left(\prod_{l_h=l_{\max}+1}^{l} N_d^{l_h} \right) \frac{\|\mathbf{X}_l\|_0}{N_t}}_{\substack{\text{Activation rate transfered from higher-level } l_h\\ \text{to level } l_{\max} \text{ based on decomposition size } N_d^{l_h}}} \cdot \underbrace{I_{l_{\max}}}_{\substack{\text{Information cost of one}\\ \text{activation at level } l_{\max}}} \tag{4}$$

$$\text{with} \quad I_l = \underbrace{(\log_2 L + \log_2 K_l + \log_2 N_t)}_{\text{Indexing bits for level, basis and time}} + \underbrace{I_{fp}}_{\text{Amplitude bits}}$$

$$I^{\text{act}}_{l_{\max}} = \sum_{l=1}^{l_{\max}} \underbrace{\frac{\|\mathbf{X}_l\|_0}{N_t}}_{\substack{\text{Activation}\\ \text{rate at level } l}} \cdot \underbrace{I_l}_{\substack{\text{Information cost of one}\\ \text{activation at level } l}} \tag{5}$$

Figure 4 shows that the weighting factor β can indeed control the distribution of non-zero coefficients amongst levels. Figure 5 shows that for a hierarchical inference task on *dataset-simple*, the framework achieves performance close to the lower bound of the information rate. It means that multiple levels of the hierarchy are used effectively assuming the proper choice of β. If β is too low, all sparse activations will be reconstructed from the higher-level dictionaries, which may not be adapted and thus significantly degrade compression

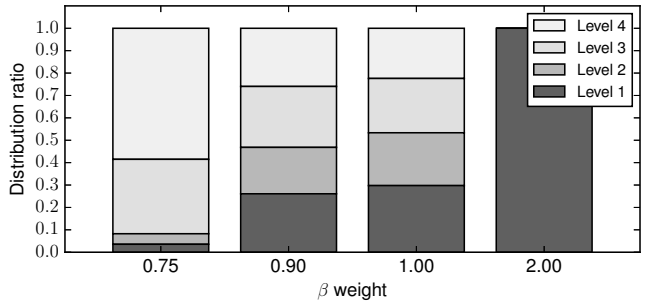
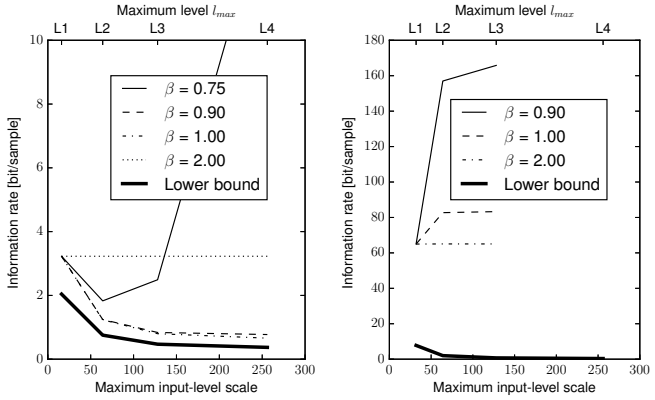


Fig. 4. Distribution of non-zero coefficients inferred across levels for *dataset-simple*, as a function of the weighting factor β. Lower β values (e.g. $\beta = 0.75$) increased the percentage of higher-level activations. When β was too high (e.g. $\beta = 2$), all was eventually encoded by the first level only.



(a) *dataset-simple* (b) *dataset-complex*

Fig. 5. Optimality of the hierarchical inference on the datasets as a function of the maximum level $l_{\max}$ considered. On *dataset-simple* (shown on left), using the weighting factor $\beta = 1.0$ achieved the best $I^{\text{act}}_{l_{\max}}$ performance, near the lower bound defined by $I^{\text{opt}}_{l_{\max}}$. Too low weighting when $\beta = 0.75$ leads to less efficient encoding with levels. Too high weighting when $\beta = 2.0$ leads to no decrease in information rate $I^{\text{act}}_{l_{\max}}$ with levels. On *dataset-complex* (shown on right), the inference failed to find an efficient encoding for all values of β because of the limitations of convolutional matching pursuit.

performance. If β is too high, the input will be encoded by the first level only and propagated up in the hierarchy only by the passthrough feature maps. The same analysis on *dataset-complex* led to very different results when we tested the limits of convolutional matching pursuit, indicating it can struggle on complex signals when overlapping between objects occurs frequently. For all values of β tested, we observed an increase in the information rate as more levels are considered at the output of the encoder. It was observed that using L^0-norm regularization during inference would add noise in the residual that needed to be removed later, and thus required a significant number of bits to encode. We also observed similar results (not shown) when using the LoCOMP [13] variant of matching pursuit. Allowing dictionary learning in the optimization process (see Equation 3) could solve this problem, as the higher-level dictionaries can be adapted to the noise introduced at inference due to matching pursuit.

V. Conclusion

In this work, we proposed a hierarchical sparse coding algorithm to efficiently encode an input signal into compact and non-redundant codes distributed across levels of the hierarchy. The main idea is to represent features at the abstraction level at which they naturally appear, rather than re-encode those through the higher-level features of the hierarchy. With the proposed method, it is possible to empirically achieve near-optimal representations in terms of signal compression, as indicated by the information rate needed to transmit the output sparse activations. The experiments highlighted the important role of the greedy inference algorithm at each level. Poor inference such as allocating too much coefficients because of interference effects directly impacts the information rate, as seen with convolutional matching pursuit. This effect was observed on complex signals, where the compression ability vanished because of the interference-related variability. Future work should aim at solving this problem that is primarily due to the usage of L^0-norm regularization.

Acknowledgment

The authors would like to thank the ERA-NET (CHIST-ERA) and FRQNT organizations for funding this research as part of the European IGLU project.

References

[1] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[2] R. Cilibrasi and P Vitányi, "Clustering by compression," *IEEE Transactions on Information Theory*, vol. 51, no. 4, pp. 1523–1545, 2005.

[3] A. Barron, J. Rissanen, and B. Yu, "The Minimum Description Length Principle in Coding and Modeling," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2743–2760, 1998.

[4] I. Ramirez and G. Sapiro, "An MDL framework for sparse coding and dictionary learning," *IEEE Transactions on Signal Processing*, vol. 60, no. 6, pp. 2913–2927, 2012.

[5] J. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Transactions on Information Theory*, vol. 23, no. 3, pp. 337–343, 1977.

[6] B. R. Rubinstein, A. M. Bruckstein, and M. Elad, "Dictionaries for Sparse Representation Modeling," *Proceedings of the IEEE*, vol. 98, no. 6, pp. 1045–1057, 2010.

[7] B. K. Natarajan, "Sparse Approximate Solutions to Linear Systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[8] L. Bo, X. Ren, and D. Fox, "Multipath Sparse Coding Using Hierarchical Matching Pursuit," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 660–667, 2013.

[9] K. Yu, Y. Lin, and J. Lafferty, "Learning image representations from the pixel level via hierarchical sparse coding," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1713–1720, 2011.

[10] S. Mallat, "Multiresolution Representations and Wavelets," Doctoral Dissertation, University of Pennsylvania, 1988.

[11] B.-H. Juang and A. Gray, "Multiple Stage Vector Quantization for Speech Coding," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, no. 3, pp. 597–600, 1982.

[12] K. Perlin, "Improving noise," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 2–3, 2002.

[13] B. Mailhe, R. Gribonval, F. Bimbot, and P. Vandergheynst, "A low complexity Orthogonal Matching Pursuit for sparse signal approximation with shift-invariant dictionaries," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 3445–3448, 2009.

[14] M. S. Lewicki and T. J. Sejnowski, "Coding time-varying signals using sparse, shift-invariant representations," in *Advances in Neural Information Processing*, pp. 730–736, 1999.