A CLASSIFIER-BASED DECODING APPROACH FOR LARGE SCALE DISTRIBUTED CODING

Kumar Viswanatha, Sharadh Ramaswamy⋆*, Ankur Saxena* †*and Kenneth Rose*

Department of Electrical and Computer Engineering,
University of California, Santa Barbara, CA 93106-9560, USA
{kumar, rsharadh, ankur, rose}@ece.ucsb.edu

ABSTRACT

Canonical distributed quantization schemes do not scale to large sensor networks due to the exponential decoder storage complexity that they entail. Prior efforts to tackle this issue have largely been limited to the suboptimal schemes of source grouping and decoding, thus failing to use all available information at the decoder. We propose a new decoding paradigm where all received bits are used in decoding. Essentially, to decode each source, we partition the space of received bit-tuples using a nearest neighbor quantizer at a decoding rate consistent with the allowed complexity and each partition is then mapped to a reconstruction value for that source. To avoid local minima in design, we resort to deterministic annealing to determine the nearest neighbor partition function (the partitioning prototypes) from the training set. Results on several data-sets show substantial gains over naive and other competing approaches.

***Index Terms*—** Distributed coding, large scale sensor networks, data compression, quantization, codebook complexity

1. INTRODUCTION

The field of distributed source coding (DSC) started with the seminal work by Slepian-Wolf [1] and Wyner-Ziv [2] in the seventies and has since been studied extensively both in the information theory and signal processing communities. Today's research related to DSC can be roughly categorized into two 'camps': one adopting ideas from channel coding (see, e.g., [3]), and another (see, e.g., [4, 5]) building directly on source coding methodologies. In this paper, the source coding based methods will be relevant to us and we will briefly describe the conventional DSC setup and its design in the next section.

Despite these numerous contributions, very little has been done related to scaling DSC for large number of sources. The conventional methods fail due to the exponential growth of decoder complexity with the number of sources. This issue was first addressed in [6], where the authors proposed a pdf-optimized source clustering scheme aimed at grouping Gaussian sources based on their correlations. By restricting the number of sources in each group, affordable decoder storage complexities were met. But this method ignores inter-cluster correlations and hence is inefficient if the number of clusters is high.

In our prior work on large scale distributed coding [7], we proposed a structured approach to operate at moderate complexities by introducing a new module at the decoder, called the 'bit-subset selector' which judiciously selects a subset of the received bits to decode each source. We showed that, even a naive module, such as the bit-subset selector which neglects a subset of the received bits to decode each source outperforms source grouping techniques. In this work, we propose a new setup for the decoder, which instead of discarding bits, optimally compresses the received bits to the allowable decoding rate.

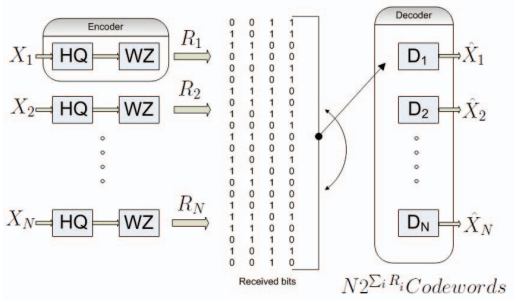

Fig. 1. Conventional DSC setup

2. MOTIVATION

We begin with a brief description of the conventional DSC [1]. Consider N correlated sources, $\{X_i, i = 1 \dots N\}$ communicating with the central receiver over noiseless channels at rates R_i respectively. The decoder uses all the received bits to reliably reconstruct each source, $\{\hat{X}_i, i = 1 \dots N\}$. The objective is to design the encoders and decoders to minimize the distortion between the observations and the reconstructions. A typical setup is illustrated as a block diagram in Fig. 1.

The encoding at each source consists of two stages. The first stage is a simple high rate quantizer, $\mathcal{H}_i$, which discretized the source-space into a finite number of regions N_i,[2] i.e.

$$\mathcal{H}_i : X_i \in \mathcal{R} \rightarrow \mathcal{L}_i = \{1 \dots N_i\} \quad (1)$$

The second module, which we call a 'Wyner-Ziv map' (WZ-map) relabels the N_i quantization regions with a smaller number, 2^{R_i}, of indices. Though the WZ-maps perform lossy compression, if properly designed, they exploit the inter-source correlation efficiently,

⋆S. Ramaswamy is now with Mayachitra Inc., Santa Barbara, USA

†A. Saxena is now with Samsung Telecom, Dallas, USA

This work was supported by the NSF under grants CCF-0728986 and CCF-1016861

[1] For details refer to [5, 4]

[2] These quantizers are made high rate so as to exclude them from the joint encoder-decoder design. This is a practical engineering necessity and we refer to [5] for more details.

thereby providing better rate-distortion performance. Mathematically, the WZ-map for each source, $\mathcal{W}_i$, is the following function :

$$\mathcal{W}_i : \mathcal{L}_i \rightarrow \mathcal{I}_i = \{1 \dots 2^{R_i}\} \tag{2}$$

and the encoding operation can be expressed as a composite function:

$$I_i = \mathcal{E}_i(x_i) = \mathcal{W}_i\left(\mathcal{H}_i(x_i)\right) \ \forall i \tag{3}$$

We denote the received bit-tuple by $I = (I_1, I_2 \dots I_N)$ and the set of all possible received indices by $\mathcal{I} = (\mathcal{I}_1 \times \mathcal{I}_2 \dots \mathcal{I}_N)$. The total number of bits received at the decoder is given by $R_r = \sum_{i=1}^{N} R_i$. The decoder reconstructs each source based on the received index $I \in \mathcal{I}$. Formally, for each source i the decoder is given by the mapping:

$$\mathcal{D}_i : \mathcal{I} \rightarrow \hat{X}_i \in \mathcal{R} \ \forall i \tag{4}$$

The WZ-maps and the decoders are typically iteratively designed using an available training set of source samples. Usually the decoder is assumed to be a lookup table, which has the reconstructions stored for each possible received index. For optimal decoding, the decoder has to store a unique reconstruction for each received index, thereby making the size of the lookup table grow exponential in N, given by $\mathcal{O}\left(N2^{R_r}\right) = \mathcal{O}\left(N2^{\sum_{i=1}^{N} R_i}\right)$. We call the total decoder storage as its complexity. This exponential growth in complexity of the optimal decoder makes it infeasible to use the conventional setup in practical settings even with moderately large number of sources.

In most of the prior research, this issue is addressed by grouping the sources together based on the source distributions [6, 8]. In our previous work [7], we introduced a new module at the decoder called the bit-subset selector, which uses a subset of the received bits to decode each source. In this work, we generalize this setup by introducing a 'Bit-mapper' at the decoder which facilitates the reuse of the stored codewords for different received indices.

It is worth mentioning an interesting decoding approach, alternative to the lookup table, that has been proposed in the literature, wherein, the decoder performs computations to generate the codewords when it receives an index. Here, the decoder storage complexity is translated to the decoder computational complexity, which grows exponentially for optimal decoding. Prior works [8] have tackled this issue by approximating the sources to form a low complexity Bayesian network. As the main objective in the two approaches are seemingly different, further comparison and applicability of the two approaches will be addressed as part of future work.

3. PROPOSED APPROACH

As mentioned before, the decoder receives information at the rate of R_r bits per instant. Clearly, source reconstructions at this rate are unfeasible for large networks as the required codebooks become exponentially bigger. We also note that the individual source decoding rates are much smaller. Hence, in our proposed approach, instead of discarding bits to match the individual decoding rates (as was done in prior works), we aim to find the optimal transformation that can compress the received bits down to the allowable decoding rate. To this end, we decompose the monolithic decoder into a compressor/bit-mapper followed by a look-up table of reconstructions/codebook. In its most general form, this bit-mapper is a classifier or a vector quantizer and is defined as the mapping $\mathcal{B}_i : \mathcal{I} \rightarrow \mathcal{K}_i$. Note that the domain of this mapping is the space of received indices and the most general form of the bit-mapper is a look-up table that maps each element of the domain to a value in the range. This

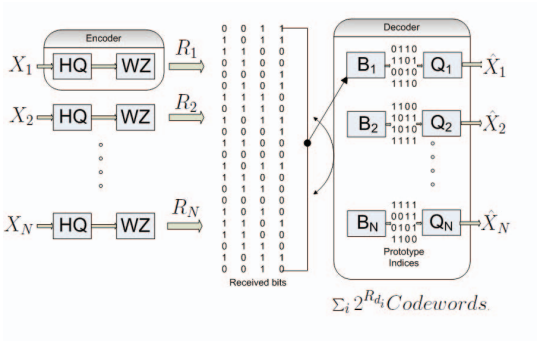

Fig. 2. Proposed Setup: Decoder operates in two stages, (1) Bit mapper (2) Labeling functions

is unfeasible for large networks as the domain grows exponentially larger with network size. Hence we impose a structural limitation on the bit-mapper by requiring it to be a nearest neighbor classifier [9], which is defined as a mapping $\mathcal{B}_i : \mathcal{I} \rightarrow \mathcal{J}_i$, $|\mathcal{I}| \geq |\mathcal{J}_i| = 2^{R_{d_i}} \ \forall i$, where the set $\mathcal{J}_i$ is called the prototype index set. This structure for the bit-mapper enforces each received index to cluster to the nearest prototype based on a minimum distance criterion. Mathematically:

$$\mathcal{B}_i(I) : \arg\min_{S \in \mathcal{J}_i} d_i(I, S) \ \forall i \tag{5}$$

where $\mathcal{J}_i = \{S_{i1} \dots S_{i|\mathcal{J}_i|}\}$ is the set of prototypes associated with decoding source i and $d()$ is any well defined distance measure. Note that this distance measure is not related to the end to end distortion measure between the source samples and their reconstructions. Since I is a R_r dimensional bit vector, we also choose the prototypes to be bit-vectors in the R_r dimensional space. A natural distance measure for this space is the Hamming distance[3]. To decode source i, the labeling function, $\mathcal{Q}_i$, assigns to every prototype a distinct codeword which is used as the reconstruction for source i. The reconstructions can now be expressed as $\hat{X}_i = \mathcal{Q}_i\left(\mathcal{B}_i(I)\right) \forall i$.

Enforcing this structure requires us to store the $\sum_i |\mathcal{J}_i| = \sum_{i=1}^{N} 2^{R_{d_i}}$ prototypes and N lookup tables with $|\mathcal{J}_i|$ entries for the codewords. By controlling the number of prototypes, we can operate at affordable decoder storage complexities. The total decoder storage required for the different approaches will be given in Section 3.4. This setup in shown schematically using a block diagram in Fig. 2.

What remains is to design the encoders, bit-mapper and codebooks efficiently to minimize the end to end distortion. The average distortion is measured as:

$$D = E\left[\sum_{i=1}^{N} \gamma_i D_i(X_i, \hat{X}_i)\right] \tag{6}$$

where $0 \leq \gamma_i \leq 1$, $\sum_{i=1}^{N} \gamma_i = 1$ weigh the relative importances of each source and D_i are any appropriately defined distortion measures. Hereafter, we will specialize the end to end distortion to be the mean squared error and assume equal weights for each source, i.e. $\gamma_i = \frac{1}{N}$. We replace expectation with an empirical average over an available training set as:

$$D = \frac{1}{N|\mathcal{T}|} \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{N} \left(x_i(k) - \hat{x}_i(\mathcal{B}_i(I(k)))\right)^2 \tag{7}$$

[3]Real-valued prototypes are also possible as are other distance measures

where $\mathcal{T}$ denotes the training set, $x_i(k)$ is the kth training sample for source i and $I(k) = (\mathcal{E}_1(x_1(k)), \ldots \mathcal{E}_N(x_N(k)))$.

3.1. Decoder design by deterministic annealing

In (7) the prototypes are present inside a non-differentiable function and the final end-to-end distortion does not depend on the prototypes in an analytic fashion. Hence it is not easy to find the optimal prototypes for this cost function. Iterative design of prototypes is not practical as it tends to get trapped in local minima. Therefore, we adopt a global optimization technique (DA) to design the prototypes [9].

Here the deterministic bit-mapper is replaced by a random mapper during the design stage. Each training set element is mapped in probability to all prototypes and this probability is denoted as, $P_i(j|k)\ \forall k \in |\mathcal{T}|, i \in \{1, \ldots, N\}, j \in \mathcal{J}_i$. Distortion is now measured as:

$$D = \frac{1}{N|\mathcal{T}|} \sum_{k=1}^{|\mathcal{T}|} \sum_{i=1}^{N} \sum_{j \in \mathcal{J}_i} P_i(j|k) \left(x_i(k) - \hat{x}_i(j)\right)^2 \tag{8}$$

Note that this includes the initial 'hard' cost function as a special case when $P_i(j|k)$ = 0 or 1. We restrict $P_i(j|k)$ to the class of Gibbs distributions of the form:

$$P_i(j|k) = \frac{e^{-\beta_i\left(d_i\left(I(k), S_{ij}\right)\right)}}{\sum_j e^{-\beta_i\left(d_i\left(I(k), S_{ij}\right)\right)}} \tag{9}$$

where β_i are called the inverse pseudo-temperatures which govern the peakiness of the distribution. This form has the advantage of converging to a nearest-neighbor classifier as $\beta_i \to \infty$, since correspondingly $P_i(j|k) = 1$ for the nearest neighbor and zero every where else.

The soft mappings introduce randomness into the system which is captured by the system entropy measured as:

$$H = \frac{1}{N|\mathcal{T}|} \sum_{k \in \mathcal{T}} \sum_{i=1}^{N} \sum_{j \in \mathcal{J}_i} P_i(j|k) \log P_i(j|k) \tag{10}$$

During each iteration of design, the distortion is minimized subject to a constraint on the system randomness, where the level of randomness is controlled by a Lagrange parameter termed 'temperature' (given the analogy to thermodynamics). The Lagrangian is denoted as:

$$J = D - TH \tag{11}$$

At each temperature, we minimize J (11) with respect to $\mathcal{J}_i$, β_i and $\mathcal{Q}_i\ \forall i$. This minimization can be achieved using any standard local minimization techniques. As β_i are real valued, the gradients with respect to β_i are given by:

$$\frac{\delta J}{\delta \beta_i} = \frac{1}{N|\mathcal{T}|} \sum_k \sum_j \Big\{ \left(x_i(k) - \hat{x}_i(k)\right)^2 + T\left(1 + \log(P_i(j|k))\right) P_i(j|k) \Big(\sum_{j'} P_i(j'|k) \left\| I(k) - S_{ij'} \right\| - \left\| I(k) - S_{ij} \right\| \Big) \Big\} \tag{12}$$

As $\mathcal{J}_i$ are R_r bit numbers, we employ a discrete optimization approach, where in each step, we find an incrementally better prototype among the neighboring prototypes. Mathematically, prototype update is given by :

$$S_{ij}^* = \arg \min_{S \in N(S_{ij})} J \tag{13}$$

where $N(S_{ij})$ is the set of prototypes which includes S_{ij} and its neighbors.

Finally, differentiating (11) w.r.t $\mathcal{Q}_i$, we get the optimal codebook update rule to be a generalization of the centroid rule and is given by:

$$\hat{x}_i(j) = \mathcal{Q}_i(j) = \frac{\sum_k P_i(j|k) x_i(k)}{\sum_k P_i(j|k)} \tag{14}$$

We now describe the algorithm for decoder design for fixed encoders. The initial temperature is set to ∞ (high value) and the β_i are initialized to 0 (any low value). $\mathcal{J}_i$ are set to the median of the training indices in the R_r dimensional bit plane and the $\mathcal{Q}_i$ are set to the mean of the training set. T is gradually lowered, and at each T $\mathcal{J}_i$, β_i and $\mathcal{Q}_i$ are optimized using (12),(13) and (14). As the temperature is lowered, the entropy of the system reduces. As $T \to 0$, we get a hard decoder with every received index mapping to the nearest prototype.

3.2. Design of Encoders

Once the decoder has been designed using DA, we choose the WZ-map at each encoder to be the optimal choice given the decoder as (also refer [7]):

$$W_n(m) = \arg \min_{l \in \mathcal{I}_i} \sum_{x(k) \in \mathcal{T}_{n,m}} \sum_{i=1}^{N} \left(x_i(k) - \mathcal{Q}_i\left(\mathcal{B}_i\left(I_{n,l}(k)\right)\right)\right)^2 \tag{15}$$

$\forall n, m$ where $\mathcal{T}_{n,m} = \{x \in \mathcal{T} : H_n(x_n) = m\}$ and

$$I_{n,l} = (\mathcal{E}_1(x_1), \ldots \mathcal{E}_{n-1}(x_{n-1}), l, \mathcal{E}_{n+1}(x_{n+1}), \ldots \mathcal{E}_N(x_N))$$

3.3. Algorithm for system design

The high rate quantizers are designed independent of the rest of the modules using a standard Lloyd-Max approach. Given some initial WZ-maps, prototypes and codebooks, the encoders and decoders are designed iteratively till convergence. To mitigate the effect of local minima due to the WZ-maps, the design is repeated over multiple random initializations. It can be shown that the design complexity grows as $\mathcal{O}(|\mathcal{T}|N^2)$, which is same as that with the source grouping methods, but we omit the derivation here due to space constraints.

3.4. Storage complexity comparison

To compare the storage requirements we assume $R_{d_i} = R$ and $R_i = R_s\ \forall i$, hence, $R_r = NR_s$. Without getting into further details, we enumerate the storage requirements for the different approaches in Table 1 where F denotes the number of bits required to store a real number.

	Codebook storage	Module storage
Source grouping	$N2^R F$	$N \log(\frac{NR_s}{R})$
Bit-subset selector	$N2^R F$	$NR \log(NR_s)$
Bit mapper	$N2^R F$	$N^2 R_s 2^R$

Table 1. Comparing storage complexities

4. RESULTS

We conducted preliminary simulations for two sensor networks consisting of 8 and 50 sources respectively. The sources were randomly

deployed on a square grid of dimensions 100×100. We considered synthetic Gaussian sources, $\mathcal{N}(0,1)$, with the correlation dropping exponentially with the distance. Specifically, the correlation between two sources at a distance d was assumed to be $\rho^{\frac{d}{d_o}}$. For both the simulations, the system was trained using a training set and tested on a test set, each of length $20,000$ samples.

We assumed the decoding rate to be the same for all sources, i.e., $R_{d_i} = R \ \forall i$. By varying R, different points on the complexity-distortion curve were obtained, where the complexities are calculated based on Table 1. We compare the performance of our technique with the source grouping method, where the sources are grouped heuristically based on their correlations, making sure that the sources with higher correlations are grouped together.

We assumed each source transmits 1 bit, i.e., $R_i = 1 \forall i$ and the high rate quantizers partition the source space into 16 regions. For all methods, we report the best performance over several random initializations (limited to 25). To avoid any potential fairness issues, we used the same high-rate quantizers for all the methods. We also initialized each system using the modules which gave the best performance for the other setups.

4.1. 8 Sources

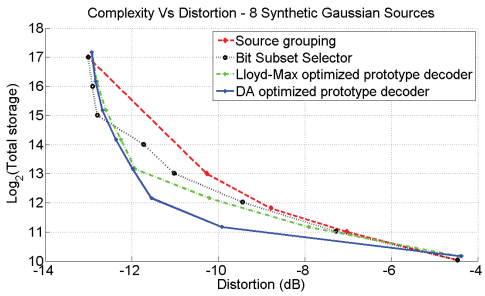

Fig. 3. Complexity vs Distortion for 8 Synthetic Gaussian sources placed randomly on a square grid

We chose $\rho = 0.95$ and $d_o = 100$ for this case. In Fig. 3 we plot the total decoder storage vs the distortion for the different approaches. We see that the bit-subset selector itself gains over the source-grouping method up to 0.7 dB. The proposed approach further improves the gains to about 1.7 dB at a decoder storage of 2^{13}. It can also be observed that the bit-subset selector outperforms the proposed bit-mapper at some storage complexities due to the additional storage required to store the prototypes. To demonstrate the issue of local-minima with the decoder design, we also plot the best results obtained by using a Lloyd-Max type iterative scheme to design the decoder optimized over multiple random initializations. We can see DA performing considerably better.

4.2. 50 Sources

For this case, we chose $\rho = 0.9$ and $d_o = 100$. We compare the performances of source-grouping method with the proposed technique in Fig. 4. The proposed approach outperforms the grouping method by about 1.5 dB at a complexity of 2^{16}. Equivalently, the proposed approach requires about $3X$ reduction in storage at an average distortion of about -8 dB. We will investigate the results for real sensor network datasets and various other synthetic source models as part of future work.

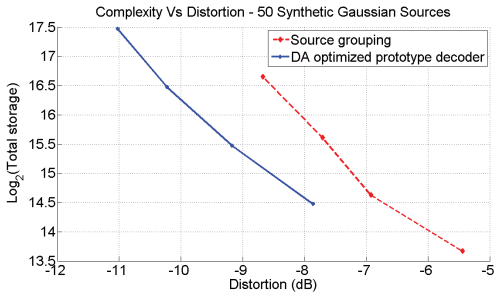

Fig. 4. Complexity vs Distortion for 50 Synthetic Gaussian sources placed randomly on a square grid

5. CONCLUSIONS

In this paper, we proposed a new methodology for distributed coding of large number of sources. The proposed scheme is a nearest neighbor classifier based approach and uses same set of codewords for different received combination of bits at the decoder. This results in low complexity decoders that operate at practical storage requirements. Simulation results show considerable improvement over other traditional naive source grouping methods. As part of future work, we seek to investigate other potential advantages of the proposed approach in error and erasure resilience.

6. REFERENCES

[1] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE. Trans. on Information Theory*, vol. 19, pp. 471–480, Jul 1973.

[2] A. D. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder," *IEEE Trans. on Information Theory*, vol. 22, pp. 1–10, Jan 1976.

[3] S. Pradhan and K. Ramchandran, "Distributed source coding using syndromes (discus): Design and construction," *IEEE Trans. on Information Theory*, vol. 49, pp. 626–643, Mar 1999.

[4] D. Rebollo-monedero, R. Zhang, and B. Girod, "Design of optimal quantizers for distributed source coding," in *in proceedings of IEEE Data Compression Conference (DCC)*, Mar 2003, pp. 13–22.

[5] A. Saxena, J. Nayak, and K. Rose, "Robust distributed source coder design by deterministic annealing," *IEEE Trans. on Signal Processing*, vol. 58, pp. 859–868, Feb 2010.

[6] G. Maierbacher and J. Barros, "Source-optimized clustering for distributed source coding," in *in Proceedings of IEEE Globel telecommunications conference (GLOBECOM)*, Nov 2006.

[7] S. Ramaswamy, K. Viswanatha, A. Saxena, and K. Rose, "Towards large scale distributed coding," in *in Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Mar 2010.

[8] R. Yasaratna and P. Yahampath, "Design of scalable decoders for sensor networks via bayesian network learning," *IEEE Trans. on communications*, pp. 2868–2871, Oct 2009.

[9] K. Rose, "Deterministic annealing for clustering, compression, classification, regression, and related optimization problems," *Proc. of IEEE*, vol. 86, no. 11, pp. 2210–2239, Nov 1998.