On Optimality of Data Clustering for Packet-Level Memory-Assisted Compression of Network Traffic

Ahmad Beirami,† Liling Huang,‡ Mohsen Sardari,† and Faramarz Fekri†
†School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
‡School of Electronic, Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China
Email: {beirami, mohsen.sardari, fekri}@ece.gatech.edu, sunny_hll@sjtu.edu.cn

***Abstract*—Recently, we proposed a framework called memory-assisted compression that learns the statistical properties of the sequence-generating server at intermediate network nodes and then leverages the learnt models to overcome the inevitable redundancy (overhead) in the universal compression of the payloads of the short-length network packets. In this paper, we prove that when the content-generating server is comprised of a mixture of parametric sources, label-based clustering of the data to their original sequence-generating models from the mixture is optimal almost surely as it achieves the mixture entropy (which is the lower bound on the average codeword length). Motivated by this result, we present a K-means clustering technique as the proof of concept to demonstrate the benefits of memory-assisted compression performance. Simulation results confirm the effectiveness of the proposed approach by matching the expected improvements predicted by theory on man-made mixture sources. Finally, the benefits of the cluster-based memory-assisted compression are validated on real data traffic traces demonstrating more than 50% traffic reduction on average in data gathered from wireless users.**

***Index Terms*—Memory-Assisted Compression; Wireless Networks; Redundancy Elimination; K-Means Clustering.**

I. Introduction

The data explosion calls for new techniques to utilize the considerable amount of correlation in the data to reduce the costs for numerous applications that require processing/transmission of individual small pieces of data, e.g., data storage and data transmission. Recently, many researchers have observed considerable correlation at the packet level (network layer) and investigated the opportunities for eliminating them by equipping some nodes in the network with memorization capability in order to perform better packet-level correlation elimination via deduplication (cf. [1]–[5] and the references therein).

However, there is a lot to be gained beyond the simple deduplication if we exploit the statistical redundancies within a packet as well as significant dependencies that exist across packets. These statistical redundancies can potentially be suppressed using statistical compression techniques. On the other hand, for an IP packet with a length in the order of several kilobytes, the state-of-the-art compression techniques (e.g., context tree weighting [6]) are inefficient in capturing the redundancy in data as compression performance primarily depends on the sequence length (cf. [7]–[9] and the references therein). In other words, there is a significant penalty with respect to what is *asymptotically* achievable when we attempt to universally compress a finite-length packet [9]. Further, since each packet may be destined to a different user, these packets cannot be effectively compressed with traditional end-to-end compression techniques which cannot leverage the cross packet dependencies.

While the individual packets cannot be compressed well, many of them share common contexts and hence are expected to be compressed more effectively if the common context between them is utilized in compression [9]–[13]. Therefore, it is desirable to encode and compress each individual packet more efficiently by utilizing the *side information* provided by the rest of the packets (while we still deliver each packet separately). In short, an investigation for a protocol-independent and content-aware network packet compression scheme suitable for removing redundancy within each packet and the dependency across multiple packets is in order.

In [11], [12], we developed a framework for compression of small sequences, called *memory-assisted compression* and introduced its potential application in compression of network packets. In memory-assisted framework, compression of a sequence is performed using a memory of the previously seen sequences. Consequently, every sequence can be compressed far better compared to the case that the sequence is compressed on its own right without considering the memory. To realize memory-assisted compression in the network, in [14], we investigated clustering for data compression and proposed a clustering algorithm based on Hellinger distance metric in the context of memory-assisted compression that grouped sequences with similar statistics for better compression. However, it remained open to identify the optimal strategy for the compression of network packets and verify it over data from real network traces.

In this paper, we first prove that label-based clustering of the data to their original sequence-generating models is optimal for memory-assisted compression of mixture sources. We further present an effective (yet tractable) algorithm for memory-assisted compression with clustering and achieve significant performance improvement on every packet from real-world traffic, on the fly with low complexity. Finally, we validate the effectiveness of the presented approach on man-made data (for which theoretical limits are known) as well as real packet-level data from wireless network users proving the usefulness of cluster-based memory-assisted compression in practice.

The rest of this paper is organized as follows. In Section II,

This material is based upon work supported by the National Science Foundation under Grant No. CNS-1017234.

978-1-4799-4903-8/14/$31.00 ©2014 IEEE

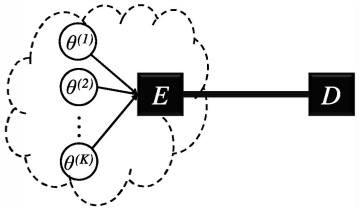

Fig. 1. The basic scenario of universal compression with side information for a mixture source.

we present the necessary background on universal compression as well as the setup of the problem. In Section III, we compare different possible schemes for memory-assisted compression of mixture sources and prove the optimality of the cluster-based memory-assisted compression. In Section IV, we briefly describe memory-assisted compression of network packets with K-means clustering. In Section V, we provide simulation results on both man-made data as well as real network traces demonstrating the effectiveness of the clustering for memory-assisted of network packets. Finally, Section VI concludes this paper.

II. Problem Definition

In this section, we first briefly review the necessary background on the universal compression. Let a parametric source be defined using a d-dimensional parameter vector $\theta = (\theta_1, ..., \theta_d) \in \Lambda_d$ that is a priori unknown, where d denotes the number of the source parameters and $\Lambda_d \subset \mathbb{R}^d$ is the space of d-dimensional parameter vectors of interest. Denote μ_θ as the parametric source (i.e., the probability measure defined by the parameter vector θ on sequences of length n).

Let $\mathcal{A}$ denote a finite alphabet. Let X^n denote a sample (random vector of length n) from the probability measure μ_θ. We further denote $x^n = (x_1, ..., x_n) \in \mathcal{A}^n$ as a realization of the random vector X^n. Then, define $H_n(\theta) \triangleq H(X^n|\theta)$ as the source entropy given the parameter vector θ, i.e.,

$$H_n(\theta) = \mathbf{E}\log\left(\frac{1}{\mu_\theta(X^n)}\right) = \sum_{x^n} \mu_\theta(x^n)\log\left(\frac{1}{\mu_\theta(x^n)}\right). \tag{1}$$

Please note that throughout this paper $\log(\cdot)$ always denotes the logarithm in base 2 and expectations are taken over the random sequence X^n with respect to the probability measure μ_θ unless otherwise stated.

In this paper, we focus on the class of strictly lossless uniquely decodable fixed-to-variable codes defined as the following. The code $c_n : \mathcal{A}^n \to \{0,1\}^*$ is called strictly lossless (also called zero-error) on sequences of length n if there exists a reverse mapping $d_n : \{0,1\}^* \to \mathcal{A}^n$ such that $\forall x^n \in \mathcal{A}^n$, we have $d_n(c_n(x^n)) = x^n$. Further, let $l_n : \mathcal{A}^n \to \mathbb{R}$ denote the universal strictly lossless length function for the codeword $c_n(x^n)$ associated with the sequence x^n such that $l_n(\cdot)$ satisfies Kraft's inequality to ensure unique decodability. That is $\sum_{x^n \in \mathcal{A}^n} 2^{-l_n(x^n)} \leq 1$.

Denote $R_{n,d}(l_n, \theta)$ as the average redundancy of the code c_n with length function l_n on a sequence of length n for the d-dimensional parameter vector θ, defined as

$$R_{n,d}(l_n, \theta) = \mathbf{E}l_n(X^n) - H_n(\theta). \tag{2}$$

Note that the average redundancy is non-negative. Further, a code is called universal if its average codeword length normalized to the sequence length uniformly converges to the source entropy rate, i.e., $\lim_{n\to\infty} \frac{1}{n} R_{n,d}(l_n, \theta) = 0$ for all $\theta \in \Lambda_d$.

Next, we present the setup of the universal compression with common side information at the encoder and the decoder. Let $\Delta \triangleq \{\theta^{(i)}\}_{i=1}^K$ denote the set of $K \triangleq |\Delta|$ parameter vectors of interest where $\theta^{(i)} \in \Lambda_{d_i}$ is a d_i-dimensional parameter vector. Note that we let K deterministically scale with n. Let $d_{\max} \triangleq \lim_{n\to\infty} \sup\{d_1, \ldots, d_K\}$ denote the maximum dimension of the parameter vectors, where we assume that $d_{\max}$ exists and is finite. Further, let $\mathbf{d} \triangleq \{d_1, \ldots, d_K\}$. We assume that for any $d < d'$, we have $\Lambda_d \subset \Lambda_{d'}$, and hence, Δ consists of K points on the space $\Lambda_{d_{\max}}$.

In this setup, we assume that $\forall i \in [K]$, we have $\theta^{(i)} = (\theta_1^{(i)}, \theta_2^{(i)}, \ldots, \theta_{d_i}^{(i)})$ are iid and chosen at random according to the least favorable prior on the d_i-dimensional parameter space Λ_{d_i}. This is to ensure that the analysis holds for the prior that maximizes the average redundancy. In this setup, as in Fig. 1, the source is a mixture of K parametric sources $\mu_{\theta^{(1)}}, \ldots, \mu_{\theta^{(K)}}$, where for all $i \in [K]$, $\theta^{(i)}$ is a d_i-dimensional unknown parameter vector. For the generation of each sequence of length n, the generator source is selected according to the probability law $\mathbf{w} = (w_1, \ldots, w_K)$ from the mixture, i.e., Δ. In other words, $p(\theta|\Delta) = \sum_{i=1}^K w_i \delta(\theta - \theta^{(i)})$, where $\theta^{(i)}$ follows the least favorable prior on Λ_{d_i} and w_i is the probability that the sequence is generated by source $\theta^{(i)}$ in the mixture. Please note that the random set Δ (which is unknown a priori) is randomly generated once according to the least favorable prior and is used thereafter for the generation of all sequences from the mixture source. Let S be the random variable associated with the source index, which follows the distribution $\mathbf{w}$ over $[K]$, i.e., $\mathbf{P}[S = i] = w_i$. Then, by definition, given Δ we have $\theta = \theta^{(S)}$. Unlike Δ that is generated once and used thereafter for the generation of sequences, S is chosen with the distribution $\mathbf{w}$ every time a new sequence is generated. Let the mixture entropy $H(\mathbf{w})$ be defined as $H(\mathbf{w}) = -\sum_{i\in[K]} w_i \log w_i$.[1]

We consider the following scenario. We assume that, in Fig. 1, both the encoder E and the decoder D have access to a common side information of T previous sequences (indexed by $[T]$) from the mixture of K parametric sources, where each of these sequences is independently generated according to the above procedure. Let $m \triangleq nT$ denote the aggregate length of the previous T sequences from the mixture source.[2] Further, denote $\mathbf{y}^{n,T} = \{y^n(t)\}_{t=1}^T$ as the set of the previous T sequences shared between E and D as side information, where $y^n(t)$ is a sequence of length n generated from the source $\theta^{S(t)}$ and $S(t)$ follows $\mathbf{w}$ on $[K]$. In other words, $y^n(t) \sim \mu_{\theta^{(S(t))}}$. Further, denote $\mathbf{S}$ as the vector $\mathbf{S} = (S(1), ..., S(T))$, which contains the indices of

[1] We define entropy $H(\mathbf{r})$ for any vector $\mathbf{r}$ such that $\sum_i r_i = 1$ in the same manner throughout the paper.

[2] For simplicity of the discussion, we consider the lengths of all sequences to be equal to n. However, most of the results are readily extendible to the case where the sequences are not necessarily equal in length.

the sources that generated the T previous side information sequences.

Let $\hat{l}(x^n, \mathbf{y}^{n,T})$ denote a generic length function that utilizes the side information $\mathbf{y}^{n,T}$ in the compression of a new sequence x^n. Please note that decoding is performed using a function $d_n : (\{0,1\}^*, \mathcal{A}^{n\times T}) \to \mathcal{A}^n$ that utilizes the side information sequence $\mathbf{y}^{n,T}$ as well. The objective is to analyze the average redundancy in the compression of a new sequence x^n that is independently generated by the same mixture source with source index Z (which also follows $\mathbf{w}$). We investigate the fundamental limits of the universal compression with side information $\mathbf{y}^{n,T}$ that is shared between the encoder and the decoder and compare with that of the universal compression without side information of the previous sequences. It is straightforward to verify that $H(X^n|\mathbf{Y}^{n,T})$ and $H(X^n)$ for different values of the sequence length n, memory (side information) size $m = nT$, the weight of the mixture $\mathbf{w}$, and the dimensions of the parameter vectors $\mathbf{d}$ serve as two of the main fundamental limits of the compression, which seek the minimum number of bits required to represent a random sequence X^n when $\mathbf{Y}^{n,T}$ is present as side information or not.

III. Fundamental Limits of Universal Compression for Mixture Sources

In this section, we state our main results on the fundamental limits of universal compression for mixture sources with and without side information. In order to see the impact of the universality and side information on the compression performance, i.e., to investigate the impact of Δ being unknown, we will need to analyze and compare the average codeword length of the following important schemes described in the sequel.

- Ucomp: Universal compression, which is the conventional compression based solution that does not utilize the memory. In this case, $H(X^n)$ determines the fundamental limit of the universal compression.
- UcompSM: Simple universal compression with side information (common memory between the encoder and the decoder), which treats the side information as if it were generated from a single parametric source.
- UcompPCM: Universal compression with perfectly clustered side information (based on the source indices), which assumes that the source indices of the side information sequences are labeled, and hence, only the relevant side information is used toward the compression of a new sequence. In this case, $H(X^n|\mathbf{Y}^{n,T}, \mathbf{S}, Z)$ determines the fundamental limit of the universal compression.[3]
- UcompOM: Optimal universal compression with side information, which optimally utilizes the side information sequence $\mathbf{y}^{n,T}$ to minimize the average redundancy. In this case, $H(X^n|\mathbf{Y}^{n,T})$ determines the fundamental limit of the universal compression.
- UcompCM: Universal compression with clustering of the side information, which is the practical clustering-based scheme proposed in this paper and shall be described in Section IV.

In [15], we exactly characterized the performance (average codeword length) of the abovementioned schemes(except UcompCM as it is just one example of a practical algorithm presented in this paper). We restate the main findings here and refer the interested reader to see [15] for the details of the results and the proofs.

Theorem 1. *In the case of* Ucomp*, we have*

$$H(X^n) = \sum_{i=1}^{K} w_i H_n(\theta^{(i)}) + \sum_{d=1}^{d_{max}} \frac{d}{2} v_d \log n + O(1)\, a.s.$$

Remark. According to Theorem 1, in the universal compression of a sequence of length n from the mixture source, the main term of the redundancy scales as the weighted average of $\frac{d}{2}\log n$ terms. This can be significantly large if $H(\mathbf{w}_d)$ is much smaller than $\frac{d}{2}\log n$. Again, if $K = 1$, we have $H(X^n) = H_n(\theta) + \frac{d}{2}\log n + O(1)$; this is exactly the well-known behavior of the average codeword length in the case of one unknown d-dimensional source parameter vector.

Theorem 2. *In the case of* UcompOM*, we have*

$$H(X^n|\mathbf{Y}^{n,T}) = \sum_{i=1}^{K} w_i H_n(\theta^{(i)}) + \sum_{d=1}^{d_{max}} v_d \sum_{i=1}^{K} \hat{w}_{d,i}\hat{R}_{d,i} + O\left(\frac{1}{\sqrt{T}} + \frac{1}{n}\right) a.s.,$$

where $\hat{R}_{d,i}$ *is given by*

$$\hat{R}_{d,i} = \frac{d}{2}\log\left(1 + \frac{n}{\hat{w}_{d,i} m}\right). \tag{3}$$

Remark. Theorem 2 characterizes the redundancy of the optimal universal compression scheme with side information, which uses a memory of size $m = nT$ (T sequences of size n) in the compression of a new sequence of length n. It is natural to expect that the side information will make the redundancy decrease. The redundancy of the UcompOM decreases when $H(\mathbf{w})$ or roughly K is sufficiently small. Again, $K = 1$, gives $H(X^n|\mathbf{Y}^{n,T}) = H_n(\theta) + \frac{d}{2}\log\left(1 + \frac{n}{m}\right) + O\left(\frac{1}{n} + \frac{1}{\sqrt{T}}\right)$, which was also characterized in [11].

Next, we state the most important conclusion that is drawn from our results, which also happens to be the main motivation for us to perform clustering of the network data for achieving better memory-assisted compression performance.

Theorem 3. *The performance of* UcompOM *and* UcompPCM *are almost surely asymptotically equivalent. That is,*

$$H(X^n|\mathbf{Y}^{n,T}) = H(X^n|\mathbf{Y}^{n,T}, \mathbf{S}, Z) + O\left(\frac{1}{\sqrt{T}} + \frac{1}{n}\right) a.s.$$

[3]Please note that UcompPCM scheme is not practically interesting as the index labels S and Z of the sequence is usually not available.

Remark. This is indeed a significant result with significant implications. It states that the performance of optimal universal compression with side information (UcompOM), which uses a memory of size $m = nT$ (T sequences of size n) in the compression of a new sequence of length n is equal to that of the universal compression with perfectly clustered memory (UcompPCM) up to $O\left(\frac{1}{\sqrt{T}} + \frac{1}{n}\right)$ terms. Hence, when T is sufficiently large, we expect that both have the same performance. This indeed demonstrates that *clustering* is optimal for the universal compression with side information. As such, we pursue the clustering of the side information (i.e., memory) in this paper in Section IV.

IV. Parametric Clustering for Mixture Models

In this section, we present the parametric clustering solution for network packets. The K-means algorithm can be used for this purpose provided that proper feature space and distance metrics are selected.

A. Feature Extraction

Feature extraction deals with extracting simpler descriptions for a large set of data that can accurately describe characteristics of original data. For memoryless source models, the frequency of each alphabet in the sequence defines an empirical probability density distribution vector which also happens to be the sufficient statistics. Although for more sophisticated source models, the empirical probability distribution of the packets is not a sufficient statistics anymore as collisions may occur between different parametric sources in the marginal symbol distribution, the empirical probability distribution would still match for packets from the same source. In this paper, we choose the vector of the empirical probability distribution as our features and since we work at the byte granularity (i.e., $|\mathcal{A}| = 256$), the feature vector is 255-dimensional. Please note that the chosen feature space is not necessary optimal but simulations confirm that it works well in practice for packets of size 1,500 bytes.

B. Clustering

As discussed earlier in Section II, we have a side information sequence of packets $\mathbf{y}^{n,T}$ that consists of T packets that originated from a mixture source model. The goal is to classify the packets into K different clusters without knowing K. Intuitively, we can think of a cluster as a group of packets that are close to each other in some space defined by a distance metric when compared with the distances to points outside of the cluster. In this paper, we choose to use the Euclidean distance metric between any two packets. Please note that Euclidean distance metric is not necessarily the optimal metric for the purpose of clustering of network packets. For each packet $y^n(t)$, we introduce a binary indicator $b_{tk} \in \{0,1\}$ (where $k = 1,\ldots,K$) that describes which of the K clusters the packet $y^n(t)$ is assigned to, so that if packet $y^n(t)$ is assigned to cluster k then $b_{tk} = 1$, and $b_{ik} = 0$ for $i \neq k$ [16]. We can define an objective function as

$$J = \sum_{i=1}^{k} \sum_{k=1}^{K} b_{ik} \left|\left| y^n(t) - u_k \right|\right|,$$

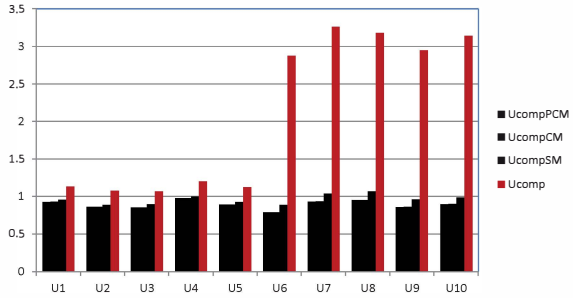

Fig. 2. Average compression-rate for a mixture of five memoryless and five first-order Markov sources.

which represents the sum of the distances of each packet to its assigned vector u_k, where u_k is the probability distribution vector of the symbol set. The goal of clustering algorithm is to find values for the $\{b_{ik}\}$ and the $\{u_k\}$ that minimize J.

C. Classification

Once the clustering of memory is performed, to compress a test packet x^n, we need to classify the packet to one of the K clusters. Then we use the assigned cluster of packets as the side information to compress x^n. The test packet x^n is assigned to the closest cluster by function

$$b = \arg \min_{1 \leq j \leq K} \left|\left| x^n - u_j \right|\right|.$$

V. Simulation Results

The simulations are divided to two parts. In the first part, we generate mixture sources and validate our theoretical findings. Next, we also present results of simulation on real network traffic traces.

A. Simulations on Man-Made Mixture Models

In order to validate the theoretical results of the paper, we chose to use a mixture of parametric sources as the content-generator for the traffic. In particular, we used a mixture of five memoryless and five first-order Markov sources on 256-ary alphabet ($|\mathcal{A}| = 256$). Each packet is selected uniformly at random from the mixture source. Consequently for a memoryless Markov source the number of source parameter d is 255, and for a first-order Markov source d is 256×255 which is the number of independent transition probabilities. For short-length sequences, we generate 18,000 packets at random from this source model, where each packet is 1,500 bytes long. Then, we use 200 packets from each source as test packets for the purpose of evaluation.

Figure 2 demonstrates the results of the simulation on man-made data generated from the described mixture source. Users U1 through U5 are memoryless whilst users U6 through U10 are first-order Markov sources and the memory is comprised of the mixture of these 10 years. We use lite PAQ-based compression for Ucomp, UcompSM, UcompCM, and UcompPCM. Please note that lite PAQ is a very effective compression scheme which performs better than the well-known Lempel-Ziv [17] and context tree weighting [6] algorithms. As can be seen, lite PAQ is already doing a poor job when the sequence is from a first-order Markov demonstrating the need for memory-assisted compression. This is

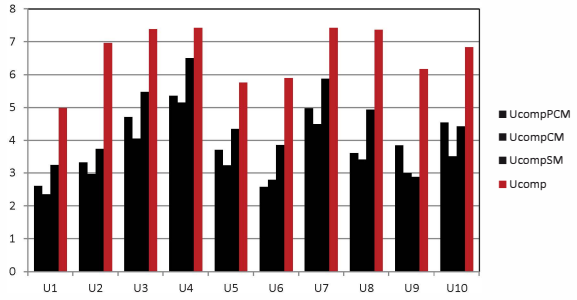

Fig. 3. Average compression-rate for the data gathered from the mixture of 10 network users.

TABLE I
THE AVERAGE COMPRESSION RATE (BITS/BYTE) AND AVERAGE TRAFFIC REDUCTION (%) OF DIFFERENT COMPRESSION SCHEMES ON THE REAL NETWORK TRAFFIC TRACES USING LITE PAQ.

Scheme	Avg. Comp. Rate	Avg. Traffic Reduc.
Ucomp	6.62	17.2%
UcompSM	4.53	43.4%
UcompPCM	3.93	50.9%
UcompCM	3.50	56.2%

also in agreement with the predictions from [8], [9]. We can see that UcompPCM is consistently better than UcompSM and UcompCM as it is the optimal way of classification and clustering, however, UcompPCM is impractical in most scenarios.

B. Simulations on Real Network Traces

Next, we perform experiments on data gathered from 10 wireless users using real network traces. We chose to mix the data from these 10 users in order to simulate the situation that occurs in an intermediate router that is serving these users. Again, the total number of packets in the memory are 18,000 packets at random from this source model. Then, we use 200 packets from each source as test packets for the purpose of evaluation and average out the result.

Figure 3 contains the average compression-rate on these data. Please note that we slightly abused the notation and used UcompPCM for compression based on the user from which the data is gathered (and not the *unknown* content-generating source). Here, indeed we do not have access to anything other than the user ID. As can be seen, UcompCM, which is the cluster-based memory-assisted compression presented in this paper, consistently outperforms all other schemes as data from one user is not necessarily from one source. Table I demonstrates the average compression-rate over all the 10 users as well the average traffic reduction achieved in this scenario. As can be seen, while lite PAQ (which is one of the very best compression algorithms) only offers 17% traffic reduction on average for the data gathered from these 10 users, by using cluster-based memory-assisted compression more than 50% traffic reduction is achieved. Furthermore, clustering an additional 5% improvement over the situation where the data from the users are clustered according to the user they are destined to. This confirms that the data destined to a single user are not necessarily from the same content-generating source.

VI. CONCLUDING REMARKS

In this paper, we studied the problem of memory-assisted network packet compression from a theoretical point of view. We compared several different possible schemes for memory-assisted compression of mixture sources and proved that cluster-based memory-assisted compression is indeed optimal. We provided a simple clustering algorithm based on K-means clustering for memory-assisted compression of network packets. Our simulation results validated the effectiveness of the clustering for memory-assisted of network packets in practice. Currently, we are investigating the the proper distance metric, clustering method, and feature space that can further improve the performance of the cluster-based memory-assisted compression.

REFERENCES

[1] S. Hsiang-Shen, A. Gember, A. Anand, and A. Akella., "Refactoring content overhearing to improve wireless performance," in *MobiCom*, Las Vegas, NV, 2011.
[2] A. Anand, V. Sekar, and A. Akella, "SmartRE: an architecture for coordinated network-wide redundancy elimination," *SIGCOMM*, vol. 39, no. 4, pp. 87–98, 2009.
[3] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker, "Packet caches on routers: the implications of universal redundant traffic elimination," *SIGCOMM*, vol. 38, pp. 219–230, 2008.
[4] E. Halepovic, C. Williamson, and M. Ghaderi, "Enhancing redundant network traffic elimination," *Computer Networks*, vol. 56, pp. 795–809, 2012.
[5] E. Zohar, I. Cidon, and O. O. Mokryn, "The power of prediction: cloud bandwidth and cost reduction," in *Proceedings of the ACM SIGCOMM 2011 conference*, ser. SIGCOMM '11. New York, NY, USA: ACM, 2011, pp. 86–97. [Online]. Available: http://doi.acm.org/10.1145/2018436.2018447
[6] F. Willems, Y. Shtarkov, and T. Tjalkens, "The context-tree weighting method: basic properties," *IEEE Trans. Info. Theory*, vol. 41, no. 3, pp. 653–664, May 1995.
[7] J. Rissanen, "Universal coding, information, prediction, and estimation," *IEEE Trans. Info. Theory*, vol. 30, no. 4, pp. 629 – 636, Jul 1984.
[8] N. Merhav and M. Feder, "A strong version of the redundancy-capacity theorem of universal coding," *IEEE Trans. Info. Theory*, vol. 41, no. 3, pp. 714 –722, May 1995.
[9] A. Beirami and F. Fekri, "Results on the redundancy of universal compression for finite-length sequences," in *IEEE Intl. Symp. Info. Theory (ISIT)*, Jul 31-Aug 5 2011, pp. 1504–1508.
[10] R. E. Krichevsky and V. K. Trofimov, "The performance of universal encoding," *IEEE Trans. Info. Theory*, vol. 27, no. 2, pp. 199–207, March 1981.
[11] A. Beirami, M. Sardari, and F. Fekri, "Results on the fundamental gain of memory-assisted universal source coding," in *2012 IEEE International Symposium on Information Theory (ISIT '2012)*, July 2012, pp. 1092–1096.
[12] M. Sardari, A. Beirami, and F. Fekri, "Memory-assisted universal compression of network flows," in *IEEE INFOCOM*, Orlando, FL, March 2012, pp. 91–99.
[13] J. Ziv, "A universal prediction lemma and applications to universal data compression and prediction," *IEEE Trans. Info. Theory*, vol. 47, no. 4, pp. 1528 –1532, May 2001.
[14] M. Sardari, A. Beirami, J. Zou, and F. Fekri, "Content-aware network data compression using joint memorization and clustering," in *IEEE INFOCOM*, Turin, Italy, April 2013.
[15] A. Beirami, M. Sardari, and F. Fekri, "Results on the optimal memory-assisted universal compression performance for mixture sources," in *51st Annual Allerton Conference*, Oct. 2013, pp. 890–895.
[16] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.
[17] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Info. Theory*, vol. 23, no. 3, pp. 337–343, May 1977.