Lossy Compression of Dynamic, Weighted Graphs

Wilko Henecka
School of Mathematical Sciences
University of Adelaide, Australia
wilko.henecka@adelaide.edu.au

Matthew Roughan
School of Mathematical Sciences
University of Adelaide, Australia
matthew.roughan@adelaide.edu.au

***Abstract*—A graph is used to represent data in which the relationships between the objects in the data are at least as important as the objects themselves. Large graph datasets are becoming more common as networks such as the Internet grow, and our ability to measure these graphs improves. This necessitates methods to compress these datasets. In this paper we present a method aimed at lossy compression of large, dynamic, weighted graphs.**

***Keywords*-graph compression; dynamic, weighted graphs; shrinkage;**

I. Introduction

Graphs (or networks) are being used more and more to represent large relational datasets: *e.g.,* computer networks, social networks, biological pathways, and so on. The size of these graphs is increasing: sometimes because the networks under study are growing (for instance the Internet); but also because we can now collect larger datasets more easily (for instance through online social networks).

As graphs grow, there is an increasing need to find means to compress them, *i.e.,* create representations that take small amounts of memory. The obvious motivation is to be able to store larger graphs, and keep such stores for longer. However, compressed representations can also lead to more efficient algorithms for working with the data [1].

Compressing graphs has been a topic of interest for at least a decade [2]–[11]. Most of the work has focussed on simple graphs, however, many graph datasets represent *weighted* graphs, *i.e.,* graphs which have a set of values associated with each edge.

The weights might indicate a distance metric, a strength of the relationship, or some other data. In this paper, we consider weights that measure the strength of association between two individuals through the proxy measurement of the number of phone calls exchanged. It might be argued that this data should not be compressed (for instance for billing), but historical data is very useful for other tasks such as fraud detection [1]. If the data is not compressed somehow, then this limits the scope of detection possible.

Moreover, fraud detection would ideally involve exchange of data between telephone companies (for issues and methods for performing such comparisons see [12]), but frequent exchanges of very large data sets (phone record graphs in the United States alone could have hundreds of millions of nodes) creates a large overhead in such a process, particularly if it has to be done via a means that provides privacy [12].

Few works consider compression of weighted graphs. In addition, many graphs evolve over time, but graph compression algorithms typically target a single static graph. Those that do consider dynamic graphs often do so by providing a static graph plus edits, but this approach does not work well for weighted graphs, where the edits might involve the weights of every single link.

Within the general field of compression there are two main strands: lossless and lossy compression, and many graph-compression algorithms have focussed on lossless compression. However, lossy compression offers the possibility of much higher compression ratios in music, image and video data, and it is to be hoped that it may do so here as well.

Of the works on graph compression, we know only of one major stream of work aimed at lossy compression of weighted, dynamic graphs. In this work the *Top-k* approximation algorithm is used. However, the authors only consider the use of this approach in a single application: fraud detection.

Here we consider the quality of this approximation algorithm in the more general context of compression, and show that it has several problems that can be corrected by our *Shrinkage* approximation.

Noise reduction is another aim here. The call-record graph is a measure of some underlying social relationships, and there are many other examples of such data (*e.g.,* the Enron email dataset [13]). The measure itself contains noise though, for instance, miss-dialled calls. Ideally, an approximation scheme would smooth out some of the noise, and both Top-k and Shrinkage perform this task.

The net result is a lossy compression algorithm for dynamic, weighted graphs with better performance than the Top-k algorithm over a range of metrics.

II. Background and Related Work

Graph compression has been a topic of interest for more than a decade, starting perhaps with the need to provide compressed representations of the WWW [4]. There have, however, only been a relatively small number of subsequent works.

We classify these by whether they operate on unweighted or weighted, and static or dynamic graphs. We also categorise the methods by whether they are lossless or lossy. An overview of our classification of related work is given in Table I, and we describe more details below.

An unweighted static graph can be losslessly compressed by applying LZW compression to a search tree of the graph [3] or by reordering the edges in such a way that techniques borrowed from full-text indexing achieve good compression for the WWW hyperlink graph [4] and for social network graphs [5].

The compression ratio can be improved by allowing for lossy compression. Navlakha *et al.* [6] replaced similar vertices in a graph with a super vertex whilst collecting the differences in the neighbourhoods of the affected vertices in a set of edge corrections. An approximate representation is achieved by allowing omission of some edge corrections, such that a user-defined bounded error is guaranteed. Gilbert and Levchenko [7] proposed several semantic compression schemes as an exploration tool to distill some important structures of a large graph. They differentiate between importance and similarity compression schemes. Whereas in the first, only 'important' nodes are retained in the compressed graph, in the latter, several 'similar' nodes are grouped into one super-node in the compressed graph. Measures of 'importance' can be (localised) node degree or shortest-path weight, and for similarity they proposed geographic or shared medium clustering, and redundant vertex elimination.

Liu *et al.* [8] represent a dynamic weighted graph as a three-dimensional tensor (Vertices $\times$ Vertices $\times$ Time). Since large dynamic graphs are mostly sparse, they only have to encode the sparse version of the tensor. They reduce the encoding cost for the sparse tensor by reducing the number of different values for the weights, thereby reducing the entropy of the weights. You *et al.* [9] used graph-rewriting rules to describe structural changes in a dynamic graph. They then use these to generate description rules to describe temporal patterns in the graph. Description rules can be used to predict future graph behaviour.

Another representation of dynamic graphs is used in fraud detection. Hill *et al.* [14] merge the historic values of the weight of the edges with their current ones by computing a weighted moving average. They further reduce the graph by only keeping the *Top-k* edges of each vertex. The memory requirements of the resulting graph thus only depend on the number of vertices.

The approach on which our work is based is that of [1], [14]. Though they did not present their work in the context of compression, their Top-k approximation is indeed a type of lossy compression. In the following we describe the natural generalisation of their approach, along with its details, and those of the Shrinkage alternative.

One final note concerns the difference between approximation and compression. A complete lossy compression scheme involves both approximation and encoding. The first step involves reduction of the information that is required to be stored, and the second stores that information efficiently. Often the two are designed in concert to gain the best compression. Here we only concentrate on the approximation step, and show that this alone provides substantial compression of the data. If encoding were incorporated into the algorithm, for instance using techniques to encode sparse tensors [8], then we should see even better performance.

Table I: Categorisation of related work.

	lossless	lossy
static, unweighted	[3]–[5]	[6], [7]
static, weighted	[2]	[10], [11]
dynamic, unweighted	[9]	
dynamic, weighted		[1], [8], [14]

III. Lossy Compression and Graph Approximation

The Top-k approximation can be seen as a specific case of a general set of approximation algorithm, which revolve around two key operations:

- the weighted sum $\alpha A \oplus \beta B$ of two weighted graphs A and B, and
- a graph approximation operation.

The former is a generalisation of the graph union operation to weighted graphs. If $G = \alpha A \oplus \beta B$ then the nodes and edges of the new graph G are the union of the nodes and edges of the graphs A and B,

$$N(G) = N(A) \cup N(B),$$
$$E(G) = E(A) \cup E(B),$$

where $N(X)$ and $E(X)$ are the nodes and edges of graph X. The weights of the edges of the new graph are

$$w_G(e) = \alpha w_A(e) + \beta w_B(e), \text{ for all } e \in E(G),$$

where we assume for convenience that

$$w_X(e) = 0, \text{ for all } e \notin E(X),$$

i.e., if an edge is not present, we treat it as if it has weight 0.

If the weights α and β are chosen such that $\alpha + \beta = 1$, then the resulting graph can be seen as a weighted average of the two input graphs.

This is a useful operation on weighted graphs, and is implicit in many graph measurement strategies which seek to create a measured graph by averaging observations over some time interval.

Here we apply it to the snapshot graphs for each day by creating an "EWMA" graph G_t using the daily graphs up to the time t as

$$G_t = \theta G_{t-1} \oplus (1 - \theta) g_t, \tag{1}$$

for some $0 < \theta < 1$. Note that the measured graph is therefore the graph equivalent of the EWMA (Exponentially Weighted Moving Average) often used in time series. The EWMA is a well-known estimator used in many domains (for instance Finance) to provide a local, smoothed estimate of dynamic values from noisy measurements.

In our application the motivation is similar. The underlying relationship graph changes (slowly) over time – new friends are made, and old ones forgotten – and so the picture of this graph must adapt. The EWMA allows for this adaptation because the weights of stale links decay away.

The second component of the approximation is the approximation operator, $A(\cdot)$. We define two such, described below, but in general the algorithm for approximating the dynamic graph can be specified as

$$\hat{G}_t = A\Big(\theta \hat{G}_{t-1} \oplus (1-\theta) g_t\Big). \tag{2}$$

Notice that the approximation algorithm differs from 1, in that it recursively approximates in terms of the cumulative approximation $\hat{G}_t$, not the graph G_t. Thus it can be calculated without recourse to storing the entire data for more than one day.

An approximation operator may perform two types of action:

1) it may prune edges from the graph (we do not allow pruning of nodes); and
2) it may perform an approximation of the edge weights.

In this work, we primarily consider the former action, though we have considered the affect of quantisation of edge weights in [12] in fraud detection, and this certainly would enhance the compression ratio in the encoding step.

A. Top-k

The underlying idea of the Top-k approximation is to generate a signature of a node's behaviour. Looking at a call-graph, in which nodes are phone numbers and directed edges represent communication between the users of those numbers, the majority of the call activity of each node is only towards a small number of their respective neighbours. Thus, a signature of the calling behaviour is the *Community of Interest (COI)* signature [1]. This signature consists of the Top-k numbers called by the target number and the Top-k numbers that call the target number.

In addition, the algorithm prunes those edges whose weights fall below some parameter ϵ, in order that stale data is removed from the approximation even for nodes that have fewer than k edges.

The pruning function ensures that only the most relevant nodes will appear in the COI signature. But since calling behaviour is heavily skewed such that most of an individual's calls are made to only a few numbers, we can choose the parameter θ and k of the COI framework such that typically 95% of all communication behaviour is accounted for in the Top-k edges. For a thorough discussion of the choice of parameters, see [14].

Pruning reduces the size of the data we need to track to at most k entries per subscriber, thus compressing the data. The reduced representation makes COI comparisons more efficient as well.

B. Shrinkage

The Top-k algorithm has two main advantages:

- it is easy and fast, and
- it performs well on the specific task of matching COIs, for instance in fraud detection.

However, it has several deficits as well:

- it has two parameters (k, ϵ), which need to be optimised on a particular set of networks,
- it is oriented specifically at a single task, and does not preserve other network characteristics (*e.g.,* the degree distribution).

We might also add to that list that it is a somewhat *ad hoc* procedure, and it might be considered desirable to have a more theoretically sound approach. Shrinkage fixes these issues, while preserving the advantages of the Top-k algorithm.

Shrinkage is used in statistics when the number of values to estimate is large in comparison to the number of data. A common instance is in estimation of autocovariance matrices [15], and this is directly analogous to our estimations problem. In both instances, a matrix with n^2 elements must be estimated from comparatively few samples, perhaps only $O(n)$.

The underlying idea of a Shrinkage estimator can be motivated by the instance of taking measurements $X \sim N(\theta, 1)$, and attempting to measure the parameters θ. The intuitive estimator is $\hat{\theta} = X$, but although this is a zero bias estimator, it doesn't always have the lowest mean squared error. There is a potential bias-variance tradeoff, *i.e.,* by allowing a small bias in the estimates, we might reduce the variance, and hence the overall errors.

Shrinkage is a simple approach to achieve this: one simply *shrinks* (moves) the estimates towards some value. The method seems counter-intuitive because we move away from the natural estimator. Sometimes this confusion is called Stein's paradox, but the approach is well-established in statistics [16], [17]. We shrink towards zero here because the weighted adjacency matrix should be a (very) sparse matrix – most of its entries should be zero. Hence our shrinkage estimator would look like $\hat{\theta} = [X - \lambda]^+$, for $0 < \lambda < 1$.

Applying shrinkage to the approximation operator we note its affect is to modify the weights so that

$$w_{\hat{G}}(e) = [w_G(e) - \lambda]^+ ,$$

where $[\cdot]^+$ denotes the positive part. When a weight is set to zero by the operation, the corresponding edge is pruned

from the graph. The value of the method depends on the choice of $\lambda > 0$ to suit the particular problem. We examine suitable choices in the results below.

The method can also be seen as *soft thresholding*, such as is conducted when denoising [18]. In this sense, it is removing or reducing "noise" in the measured graph, to more accurately obtain a picture of the underlying social network. There is a tradeoff between fidelity to the original and denoising: in our results there is no source of noise, so we show clearly what the loss of fidelity would be for a given reduction in noise (as indicated by the threshold value).

C. Complexity

Both, the Top-k and the Shrinkage approximation are composed of simple operations which only affect the edges with non-zero weights. Let $z(G)$ be the number of non-zero edges in graph G. Then the complexity to compute the EMWA graph G_t is $z(G_{t-1}) + z(g_t)$ multiplications and $min(z(G_{t-1}) + z(g_t))$ additions.

The Top-k approximation consists of two parts: First all elements below the ϵ threshold are removed. This step requires $z(G_t)$ comparisons. Then the top-k numbers are selected which can be achieved by first sorting the elements and then picking the top-k. The complexity of sorting is $\mathcal{O}(z(G_t) \log z(G_t))$.

In contrast, the Shrinkage approximation requires $z(G_t)$ subtractions and $z(G_t)$ comparisons, as we have to test for values smaller than zero.

Overall, Shrinkage has a slight advantage over Top-k, as it omits the comparatively expensive sorting operation. The complexity of both approaches depend only on the number of edges with non-zero weights.

IV. Methodology

The aim of our approach is to test compression in the setting of a dynamic, weighted graph that is a measure of some underlying set of relationships. A classic instance of such a graph is the telephone call-record graph. Telephone calls (along with their duration) stand as proxies to measure relationships.

There are many other such graphs (*e.g.,* email graphs, online social networks, ...) but call-record graphs are interesting because

1) they have been collected for almost a hundred years; and
2) they have particular applications [1] which have been used in the past to provide metrics for approximation "usefulness", if not accuracy.

Working with call-record graphs is difficult however, as the information contained in them is considered private, and is strictly regulated in most jurisdictions. In order that our results be reproducible, we instead work with simulated call data. This has the added advantage that we know the "ground-truth" social network, not just our measurements of that network.

Call-record graphs have a number of well-known characteristics: for instance, the graph is highly sparse with an approximate power-law degree distribution. These should be mirrored in any simulation dataset, so we use here a technique designed to simulate the commonly observed characteristics of the call graph.

In the following, we describe our experimental methodology, in particular we provide a brief description of the synthetic generation of a call-record graph.

A. Experiment

The generic structure of our experiments in as follows:

$$\text{Generate} \Rightarrow \text{Measure} \Rightarrow \text{Approximate}.$$

The principle is that there is some underlying network, which is measured through a proxy measurement (in our case telephone calls), and which we approximate. This approach has the advantage that we can generate multiple underlying graphs to obtain accurate statistical measures of success, and that we know the ground truth network, which we would not if all we analysed were measurements from a real network.

The detailed method of creation, and its justification are explained in [12]. However, we provide a brief description here to provide context for the results.

The underlying relationship graph S is generated by a Barabási-Albert preferential attachment graph. Each new node is connected to m existing nodes with a probability proportional to the number of links of the existing nodes. This generates the highly-variable (approximately power-law) degree distribution commonly observed in call records [19] and other social networks. There are other methods that also generate power-laws and problems with the assumption that social networks grow following this model [20], however, it is one of the simplest approaches to generate such an network. In future work, we plan to extend our analysis to other types of underlying network.

Each edge is then assigned an IID (Independent, Identically Distributed) random number $r_{ij} \sim U(0, 1)$ to which its weight will be proportional. We generate these using the uniform distribution according to Laplace's principle of indifference, which suggests this distribution in the absence of any information to the contrary. Weights are then assigned as $v_{ij} = \frac{2c}{\text{nd}(S)} r_{ij}$, where $\text{nd}(S)$ denotes the average node degree in graph S. This choice is made in order that expected average call rate per customer per day c, which is chosen to match known call rates.

Once we have this network, we generate calls by dividing time into d discrete time intervals, and creating a call in each time interval with probability $p_{ij} = v_{ij}/d$. We create a measured graph g_t for each day by taking $N(g_t) = N(S)$, creating an edge whenever there is at least one call, and then

grouping d intervals into days, and counting the number of calls between each pair to create the weight. In principle the resulting count weight w_{ij} is an estimate of v_{ij}.

We then construct a cumulative measured graph M_t using all of the call records up to the time t. This is not an intended measurement: it lacks any locality (ability to adapt to changes in the underlying social graph), and requires storage of the complete set of data. However, it does provide us with a baseline estimate to compare with other techniques.

We also construct the EWMA graph G_t using 1. Here we will use the $\theta = 0.9$ value drawn from [14]. As noted earlier this allows for locality of the estimates, but note that our underlying graph S is not dynamic, so we can see the convergence properties of our estimates. In future work we will analyse the locality characteristics of different methods.

We also use the measured graphs to generate the two approximations by applying the Top-k or Shrinkage approximations as described above. Note that we use the same value of θ in all methods to allow for consistent comparisons.

B. Metrics

In [14], the major metric for success was the performance of the approximation in the specific task of forming COIs that could be used in fraud detection. This is an important, but very specific metric. Our goal here is to expand the measurement of approximation success, and to do so we introduce two simple metrics:

1) **Weight error:** we simple measure the errors in the approximated weights, *i.e.,* the error on a link is

$$\epsilon_{ij} = \left| w_{ij} - \hat{w}_{ij} \right|,$$

where we take $w_{ij} = 0$ if $(i, j) \notin \mathcal{E}$. We used the average of the summed error of each link over the whole graph as the overall metric.

2) **Degree distribution:** the degree of a node in the graph is simply the number of edges connecting to that node. The degree-distribution records the number (or proportion) of nodes with each degree. We use this to provide a visually intuitive means to show the difference between the two approximations.

There are many other metrics one could choose, but these provide both a practical perspective (the values of the weights are important for many applications), and the graph-centric perspective (node degree is often used as a means to characterise types of graphs).

In the results below we calculate metrics on 30 generated networks.

V. Results

We use the Top-k approximation with $k = 9$ and $\epsilon = 0.1$ to match past work [1], [12], [14], where this was found to be a useful setting in the COI application. The Top-k approximation then reduces the number of edges in our graphs from 5.68 Million to about 0.678 Million, a compression ratio of 8.38:1. The Top-k implementation of [1] stores the incoming and the outgoing call-graph separately. However, as the underlying call-data is the same for both graphs, some of the edges in both graphs have the same weight, and others are different, because the Top-k pruning is applied separately. Figure 1 shows a comparison of the number of edges for the different compression techniques. The 'Top-k max' line shows the maximum number of edges to be stored for Top-k, that is, storing the incoming and outgoing graph separately. In contrast, 'Tok-k min' shows the number of different edges in the combined incoming and outgoing graph.

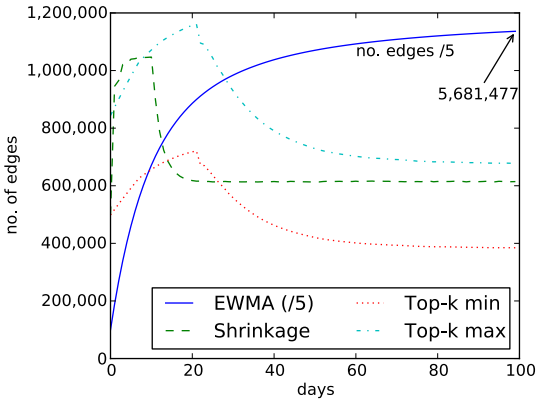

Figure 1: Comparison of number of edges for different compression techniques over 100 days.

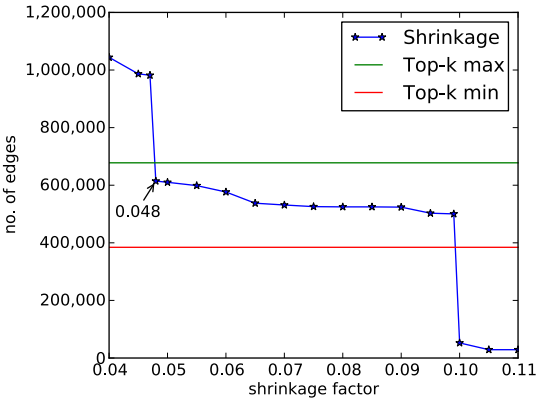

Figure 2: Number of edges after Shrinkage approximation.

In our first test we aim to find the value of λ that produces the same compression ratio. Figure 2 shows the resulting number of edges of the Shrinkage approximation as a function of λ. We can see two big jumps that can be explained with the discrete nature of the weights. As the average call rate per day is 5.2 which is spread over at least 30 links, the most common values for the measurements on the links in the daily call graph g_t are zero or one call. In the

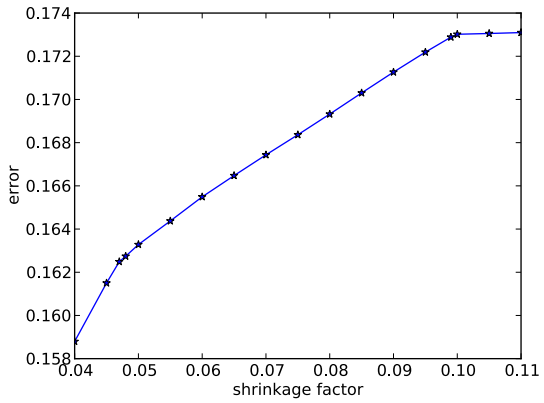

Figure 3: Weight error of Shrinkage approximation for different shrinkage factors λ.

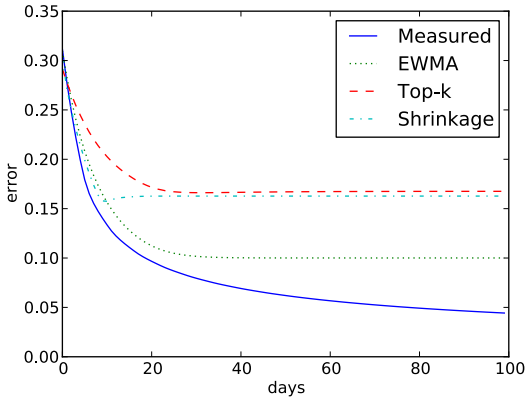

Figure 4: Average weight error for the measured graph, the top-k, and the shrinkage approximation.

EWMA computation, one call results in a value of 0.1 to be added to the historic call behaviour. Thus, a shrinkage factor $\lambda = 0.1$ prevents most new edges in the approximation graph $\hat{G}_t$. Similarly, if there is one call at one day followed by no call the next day, then a shrinkage factor $\lambda \geq 0.0474$ results in the removal of the link, as following inequality holds:

$$((1-\theta)\cdot 1-\lambda)\cdot 0.9-\lambda \leq 0.$$

The Shrinkage approximation achieves a similar compression as Top-9 for $0.0474 < \lambda < 0.1$.

Figure 3 shows the average weight error of the shrinkage compression after 100 days for different shrinkage factors λ. Compared with Figure 2 it shows that although there are two big drops in number of edges, the corresponding errors change only slightly. Thus the best trade-off between compression and weight error is just after a drop. Therefore we will use $\lambda = 0.048$ for all further comparisons.

Figure 4 shows the weight error as defined in Section IV-B over a 100 day set of measurements. The solid blue curve shows the error for the cumulative measured graph: this isn't realistic (there is no compression), but it forms a baseline estimate of the best possible estimate obtainable from the measured data. We can see that after an initially rapid decrease, the curve slowly converges towards an accurate estimate. We would expect this convergence to be in line with the Central Limit Theorem, *i.e.,* the error should converge as $1/\sqrt{t}$.

The second curve, the dotted green curve, shows the accuracy of the EWMA graph. The aim of this graph is to estimate by averaging the underlying graph, but allow some scope for locality. As such, we can see that although its initial shape is similar to the measured graph, by about day 30 it has converged to a stationary error rate. Choosing different values of θ would alter the exact level to which it converged, and the time, but not the general shape of the curve.

The dashed red curve shows the accuracy of the Top-k approximation. It has a similar shape to the EWMA curve, but it converges to a higher error value, as a direct consequence of the approximation. This is to be expected.

The surprising curve is that of the Shrinkage approximation (dot-dashed green), which also converges to a stationary level very similar to the Top-k method, but the convergence is much faster. There is almost no "burn in" period before the estimates reach a reasonable level of approximation.

The fast convergence is a particularly useful property: for instance, it makes the method easier to use in fraud detection, as less history is needed before we can create approximate COIs. It also means that we could (potentially) adjust θ to allow for faster adaptation to the underlying graph than Top-k can accommodate.

The eventual level of the two approximation methods is slightly different, but the difference is small in comparison to our baselines methods, and this might reflect the difficulty of choosing a value of λ that exactly approximates the compression achieved in the Top-k method.

The other facet of the methods that we aim to highlight here is the generic distortion of the graph caused by the Top-k approximation. Many social-relationship networks have exhibited high-variability in their node degree distribution, which means that some nodes will have degree orders of magnitude higher than others. In this case, truncating the degrees by the Top-k approximation cannot help but distort the distribution.

Figure 5 shows the node degree distribution of the underlying graph, the Top-k, and the Shrinkage approximation. We can see that the Top-k distribution is completely changed, and that would be the case for almost any underlying graph, even those without a true power-law distribution of node degree.

On the other hand, the Shrinkage estimate does reduce the average degree. It has to do so in order to provide any compression benefit. However, it preserves the shape of the

distribution, and would do so for almost any node-degree distribution in the same manner. This is the property that fundamentally makes Shrinkage a better approximation for the generic compression task.

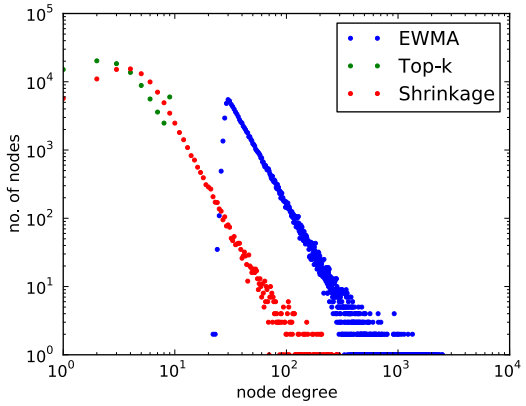

Figure 5: Node degree distribution of the measured graph, the Top-k, and the Shrinkage approximation.

VI. Conclusion

We propose a lossy compression technique for dynamic, weighted graphs. The compression is achieved in a two-stage process. First, the historic and the current behaviour are merged by an exponentially weighted moving average, thereby removing the need to store more than one version of the graph. In the second stage we remove edges of the graph by using a shrinkage technique. We compare our compression with the similar Top-k approximation technique of [1]. Their approach performs well on the specific task of COI matching, but it does not preserve other characteristics of the network well. Whereas the proposed Shrinkage approximation preserves the trend of the node degree distribution, achieves a similar level of approximation error quicker, and has only two parameter which need to be optimized on the particular set of networks instead of three parameter for the Top-k approach.

In this work, we only considered approximation operators that prune edges from the graph, though we have considered the affect of quantisation of edge weights in [12] on fraud detection, and this certainly would enhance the compression ratio for graph data. We aim to test this for compression in future work.

Acknowledgements

This work was supported by ARC grant DP0985063, and by the ARC Centre of Excellence for Mathematical & Statistical Frontiers.

References

[1] C. Cortes, D. Pregibon, and C. Volinsky, "Communities of interest," in *Advances in Intelligent Data Analysis*. Springer, 2001, pp. 105–114.

[2] J. Willcock and A. Lumsdaine, "Accelerating sparse matrix computations via data compression," in *Proceedings of the 20th Annual International Conference on Supercomputing*, ser. ICS '06. New York, NY, USA: ACM, 2006, pp. 307–316.

[3] S. Chen and J. Reif, "Efficient lossless compression of trees and graphs," in *Proceedings of the IEE Data Compression Conference (DCC '96)*, Mar 1996, pp. 428–437.

[4] P. Boldi and S. Vigna, "The Webgraph Framework I: Compression Techniques," in *Proceedings of the 13th International Conference on World Wide Web*, ser. WWW '04. New York, NY, USA: ACM, 2004, pp. 595–602.

[5] F. Chierichetti, R. Kumar, S. Lattanzi, M. Mitzenmacher, A. Panconesi, and P. Raghavan, "On compressing social networks," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '09. New York, NY, USA: ACM, 2009, pp. 219–228.

[6] S. Navlakha, R. Rastogi, and N. Shrivastava, "Graph summarization with bounded error," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '08. New York, NY, USA: ACM, 2008, pp. 419–432.

[7] A. C. Gilbert and K. Levchenko, "Compressing network graphs," in *Proceedings of the LinkKDD workshop at the 10th ACM Conference on KDD*, August 2004.

[8] W. Liu, A. Kan, J. Chan, J. Bailey, C. Leckie, J. Pei, and R. Kotagiri, "On compressing weighted time-evolving graphs," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, ser. CIKM '12. New York, NY, USA: ACM, 2012, pp. 2319–2322.

[9] C. H. You, L. Holder, and D. Cook, "Graph-based data mining in dynamic networks: Empirical comparison of compression-based and frequency-based subgraph mining," in *IEEE International Conference on Data Mining Workshops ICDMW '08*, Dec 2008, pp. 929–938.

[10] F. Zhou, S. Mahler, and H. Toivonen, "Simplification of networks by edge pruning," in *Bisociative Knowledge Discovery*, ser. LNCS, M. Berthold, Ed. Springer, 2012, vol. 7250, pp. 179–198.

[11] H. Toivonen, F. Zhou, A. Hartikainen, and A. Hinkka, "Compression of weighted graphs," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '11. New York, NY, USA: ACM, 2011, pp. 965–973.

[12] W. Henecka and M. Roughan, "Privacy preserving fraud detection across multiple phone record databases," *IEEE Transactions on Dependable and Secure Computing*, vol. PP, no. 99, pp. 1–1, 2014.

[13] W. W. Cohen, "Enron email dataset," 2009, https://www.cs.cmu.edu/~./enron/.

[14] S. B. Hill, D. K. Agarwal, R. Bell, and C. Volinsky, "Building an effective representation for dynamic networks," *Journal of Computational and Graphical Statistics*, vol. 15, no. 3, pp. 584–608, 2006.

[15] C. C. Kwan, "An introduction to shrinkage estimation of the covariance matrix: A pedagogic illustration," *Spreadsheets in Education (eJSiE)*, vol. 4, no. 3, 2011, epublications.bond.edu.au/cgi/viewcontent.cgi?article=1099&context=ejsie.

[16] B. Efron and C. Morris, "Stein's paradox in statistics," *Scientific American*, vol. 236, no. 5, pp. 119–127, 1977.

[17] P. Hoff, "Shrinkage estimators," 2013, www.stat.washington.edu/people/pdhoff/courses/581/LectureNotes/shrinkage.pdf.

[18] D. L. Donoho, "De-noising by soft-thresholding," *IEEE Transactions on Information Theory*, vol. 41, no. 3, pp. 613–627, 1995.

[19] A. A. Nanavati, S. Gurumurthy, G. Das, D. Chakraborty, K. Dasgupta, S. Mukherjea, and A. Joshi, "On the structural properties of massive telecom call graphs: Findings and implications," in *Proceedings of CIKM '06*. ACM, 2006, pp. 435–444.

[20] M. Roughan and W. Willinger, "Internet topology research redux," in *Recent Advances in Networking, Vol. 1*. ACM SIGCOMM, Aug. 2013.