We are grateful to the reviewers for their feedback. We address their concerns here.

Reviewer #1: Thank you very much for your thoughtful and detailed review. We will include all your suggestions. (1) **Learning costs from label information**. We discuss a principled way while ensuring the resulting formulation is efficiently solvable. Define $\ell(i, j) = 1$ for edge (i, j) if vertices i and j have the same label, and -1 otherwise. Let the cost function be parameterized by $\theta = (\theta_s, \theta_d)$, $\theta_s > 0, \theta_d > 0$. Define $c_\theta(i, j) = 0.5(\theta_d(1 - \ell(i, j)) + \theta_s(1 + \ell(i, j)))$. Thus, $c_\theta(i, j) \in \{\theta_s, \theta_d\}$ depending on whether i and j have the same label. We now extend eq. 3 in paper to have

$$\min_{\mathbf{0} \prec \theta} \min_{\rho_1 \in \Delta(V), ||\rho_1||_0 \leq k} \max_{t \in \mathbb{R}^{|V|}, -c_\theta \preceq Ft \preceq c_\theta} t^\top (\rho_1 - \rho_0) + 0.5\lambda ||\rho_1||^2 + 0.5||\theta||_2^2 \,.$$

Proceeding as in the paper, we obtain the following equivalent formulation (using ψ from eq. 9 in Theorem 5):

$$\min_{\mathbf{0} \prec \theta} \min_{\epsilon \in \mathcal{E}_k} \max_{\zeta \in \mathbb{R}} \max_{t \in \mathbb{R}^{|V|}, -c_\theta \preceq Ft \preceq c_\theta} \psi_{\rho_0}(\epsilon, t, \zeta) + 0.5||\theta||_2^2 \,.$$

We can thus efficiently solve the convex-concave problem obtained by relaxing each coordinate of ϵ to $[0, 1]$ in

$$\min_{\epsilon \in \mathcal{E}_k} \min_{\mathbf{0} \prec \theta, \mathbf{0} \preceq \alpha,\, \mathbf{0} \preceq \beta} \max_{\zeta \in \mathbb{R}, t \in \mathbb{R}^{|V|}} \psi_{\rho_0}(\epsilon, t, \zeta) + 0.5||\theta||_2^2 + \alpha^\top (Ft - c_\theta) - \beta^\top (Ft + c_\theta) \,.$$

(2) **Relation to other compression techniques**. Most successful algorithms try to preserve the graph spectrum via a multi-level coarsening procedure: at each level they compute a matching of vertices and merge the matched vertices, e.g., Heavy Edge contracts those edges (i, j) that are incident on low degree vertices. Likewise REC follows a randomized greedy procedure for generating maximal matching incrementally. If we set the cost $c(i, j) = \max(d_i, d_j)$ in our OTC framework, we will incentivize flow on edges with low degree vertices, and in turn, compression of one of their end points. The vertices not in support of target distribution ρ_1 may then be viewed as being matched to (a subset of) adjacent vertices that they transfer flow to. Unlike other methods, our approach is (a) flexible in terms of defining $c(i, j)$, and (b) not greedy, therefore, less susceptible to errors inherent in iterative greedy matching procedures.

Reviewer #2: Thank you very much for your constructive feedback. We address both your concerns here. (1) **Discussion of assumption in Theorem 4**. The quantity $|\hat{t}(v) - \hat{\nu}(v) + \hat{\zeta}|$ in eq. 7 may be viewed as the strength of a signal. Then, we require the vertices in support of optimal ρ_1 to have a strictly higher signal that the vertices not in the support. Such signal detection conditions appear in various contexts and often have information theoretic implications, e.g., Ising models (Santhanam and Wainwright, *IEEE Transactions on Information Theory*, 2012). Eq. 7 is also reminiscent of the β-min condition on regression coefficients for variable selection with Lasso in high-dimensional linear models (Bühlmann, *Bernoulli*, 2012), however, we do not require analogs for stringent assumptions required by Lasso such as restricted eigenvalue, and thus our Boolean relaxations are preferable to solving an ℓ_1-regularized problem. We will include this discussion based on your feedback.
(2) **Experiment settings for Fig. 1**. We apologize for the confusion. The setup for Fig. 1 is identical to that for Table 1. Fig. 1 provides visualization of compression times in addition to accuracy results from Table 1. To ensure the fairness of our experiments, for each dataset, we first executed the randomized algorithm (REC) over each graph, and then set the target number of nodes for other methods to the number of vertices in corresponding compressed graphs from REC. We performed 5 such independent executions of REC to mitigate the effect of randomness. Likewise, the compressed graphs were partitioned into multiple train-test sets for each of the different fractions, and average accuracy and standard deviations along with compression times were plotted (please see section 4.1 for the details of our implementation).

Reviewer #3: Many thanks for a comprehensive review. We are glad that you find our work exciting.
(1) **Scalability of our approach**. Please note that almost all state-of-the-art compression methods are compute-intensive since they need to perform matching as a sub-step at multiple levels. In contrast, there are no major computational bottlenecks in our approach. In particular, please note that both the projection steps in Algorithm 2 can be solved efficiently by existing algorithms. In Algorithm 1, we perform sorting that requires $O(|V| \log |V|)$ time. We believe that even this log factor may be removed from our computation by a more sophisticated algorithm, much like (Condat [43]) managed to improve on the algorithm for Euclidean projection on the simplex by (Duchi et al. [41]).
We also experimented with the Tox21_ARLBD data (https://tripod.nih.gov/tox21/challenge/data.jsp), which consists of 8589 graphs, based on your suggestion. Both our method and Algebraic distance performed very well in terms of classification accuracy ($\sim$ 97%) on this data. Our approach took in total about 39 seconds to compress graphs in this data to 90% (low compression), and about 41 seconds in total to compress to 10% (high compression). In contrast, the Algebraic distance method took about 48 seconds to compress to 90% and a significantly longer time, i.e., 3.5 minutes to compress to 10%. The other methods failed to compress this data. Please note that Algebraic distance is amongst the fastest state-of-the-art compression methods (Chen and Safro [13]).
(2) **Regarding DHFR dataset**. Indeed, though OTC performs the best on DHFR, the performance of most methods is similar (except REC, which lags behind). In contrast, REC performs better than all methods except OTC on MSRC-9. This seems to suggest that REC performs well on graphs with strong connectivity, while others might be better on data with a long backbone besides these ring structures. We believe robust performance of OTC across datasets comprising graphs with vastly different topologies underscores the promise of our approach.