Gradient-based Graph Structure Optimisation for Research Networks

Louis Mahon* **Mirella Lapata**
School of Informatics
University of Edinburgh
lmahonology@gmail.com, mlap@inf.ed.ac.uk

Abstract

The discovery and spread of novel ideas through a population is a well-studied phenomenon in computer science, computational social science, marketing and business research, and higher education studies. In this paper, we study a particular version of this phenomenon: the spread of ideas among a network of researcher agents. We use a computational model in which each agent has a finite attention budget to spend on a mixture of conducting its own research and learning from the research of others. By framing the problem as a continuous optimisation over the edge weights, we can apply gradient-based optimisation, similar to that used to train graph neural networks, to search for the graph configuration that maximises the amount of knowledge uncovered. We find that (i) the optimisation, which we call GLIDE, produces graphs with a higher rate of knowledge discovery than random, small-world and previously proposed optimised graphs, (ii) the optimised graphs are sparse, with low global efficiency and a low clustering coefficient, (iii) the nodes can be analysed as being either more like researchers, focussing almost entirely on their own research, or learners, focussing mostly on learning from the research of others, and (iv) the learners consistently have higher knowledge scores, which contributes to a rise in inequality across nodes over time.

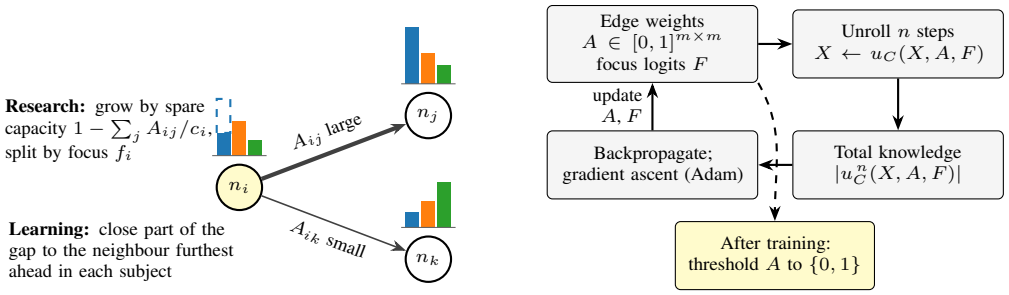
1 Introduction

In areas that require discovery of new ideas, a common trade-off is between individuals working on developing their own new ideas, which we can call *research*, and finding out about those already developed by others, which we can call *learning*. This tension between generating new knowledge and drawing on that of others has been studied as a balance between exploration and exploitation in networked groups (Lazer & Friedman, 2007; Mason & Watts, 2012). It would be inefficient for everyone to focus solely on research, as learning a piece of knowledge from someone else is presumably easier than discovering the same knowledge independently from scratch, i.e., easier than 're-inventing the wheel'. At the other extreme, if everyone focused solely on learning from others, then no new knowledge would be generated in the network and progress would halt. Thus, there is an optimal point midway between these two extremes, which may involve different research/learning ratios for different agents and different subjects. Also important is the choice of which others to learn from, and which subjects to focus on when conducting research. If you are deficient in a particular topic, it is worthwhile to learn from an agent that is particularly strong in that topic. The full problem can be stated as follows: each agent must divide their finite attention between other agents and their own research, so that the total amount of knowledge in the network increases as quickly as possible.

We model this as an optimisation problem over the edge weights of a weighted directed graph, and of the distribution over subject areas for each node. Each node is associated with an l-dimensional vector

*https://louismahon.substack.com/

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) One knowledge-sharing step

(b) Optimisation loop

Figure 1: Overview of GLIDE. (a) Bars show a node's knowledge in each subject area; X holds these values for all nodes. Through research, knowledge grows multiplicatively using the capacity c_i left after paying attention to neighbours, split by focus weights f_i. Through learning, n_i closes part of the gap (dashed) to the neighbour furthest ahead in each subject, scaled by edge weight A_{ij}. Arrows show attention: n_i learns from n_j and n_k. (b) Edge weights A for all m nodes and focus logits F (normalised to give each f_i) are optimised by gradient ascent through n applications of the update u_C, with capacities C, to maximise total knowledge ($|\cdot|$ sums all entries). A is then thresholded to give a discrete graph.

of positive real numbers, representing its degree of knowledge in each of l areas. Each edge (n_i, n_j) is associated with a single number in $[0, 1]$ during optimisation, which is then thresholded to $\{0, 1\}$ after optimisation, to represent whether n_i pays attention to n_j. At each time step, for each neighbour of n_i, the amount of knowledge it could gain for each of the l areas is equal to the difference between its value and that of the neighbour (clipped to zero if negative) for that area, multiplied by the edge weight (thresholded to $\{0, 1\}$ at test time). It gains the max of these values across its neighbours for that subject. Then it also gains knowledge from its own research, equal to its capacity c_i minus the sum of its outgoing edge weights, to account for the opportunity cost of devoting energy to learning instead of research. The knowledge acquired through research is divided up between the l areas in proportion to the agent's focus on each (Figure 1a).

After a fixed number of time steps, the total knowledge value, K, for the network is the sum of the knowledge values across all subjects and nodes. We model this with a slight variant of neural message passing from graph neural networks (GNNs) (Gilmer et al., 2017; Scarselli et al., 2009), and then gradient ascent can be used on K to optimise the research focus of each node and the edge weights (Figure 1b). The result is a graph structure, together with node-wise focus weights, that allocates each node's time between research and learning, and its attention across subject areas and neighbours, so as to maximise total knowledge.

The novel contributions of this paper are:

1. Extending the search for the optimal graph for knowledge sharing to include multiple subject areas and the optimal distribution over such areas;

2. The inclusion in the optimisation problem of a finite attention budget that has to be divided across areas and graph connections;

3. The casting of the optimal graph structure problem in a way that allows the application of gradient-based optimisation from graph neural networks;

4. Empirical experiments exploring the results of this optimisation, and their discussion in the context of social graph concepts such as agent properties and inequality.

In the remainder of this paper, Section 2 surveys existing work on the effect of social graph structure on idea propagation. Section 3 details the optimisation framework, Section 4 presents the results of computational experiments, Section 5 discusses findings in the context of existing work on optimal network structure and inequality, and finally Section 6 concludes and summarises.

2 Related Work

Graph structure and the spread of ideas. The general problem of what graph structure is optimal for idea sharing has been studied from various angles. A first line of research concerns small-world networks (Watts & Strogatz, 1998). Named after the phenomenon in social interactions (Milgram, 1967), small-world networks combine high clustering, as in a lattice, with short average path lengths, arising from mostly local connections plus a few random long-range ones. Watts and Strogatz (1998) introduced what is now a standard algorithm to create small-world graphs by beginning with a ring lattice, and then, for each edge, with probability p, changing one endpoint randomly to some other node in the graph.

Many works compare, under some measure, the behaviour of a given set of graph structures, including small-world graphs from the Watts–Strogatz algorithm. For example, Lazer and Friedman (2007) compare several network structures, including small-world graphs, and show that networks that disseminate information more efficiently perform better in the short run but worse in the long run, because less efficient networks maintain a diversity of solutions that supports exploration. This problem differs from the one we consider as it measures a variety of solutions to a single problem, whereas we model knowledge of multiple different areas, each by a scalar. Jafari et al. (2016) compare five different types of graphs for innovation dissemination. For a high innovation rate, they find that lattice structures perform the best; for a low innovation rate, the best-performing structure was scale-free, i.e., a power-law distribution in the degree of each node. In the related area of studying market adoption of a new idea or product, Choi et al. (2010) conducted simulations to examine the effect of graph structure. They considered a spectrum of graphs, including Watts–Strogatz small-world graphs under various values of the flip parameter p. They found mixed results: a high p gives a greater chance of failed penetration (below 16%), but also a greater chance of 100% penetration and a faster dissemination rate. Our work differs importantly from these studies (Choi et al., 2010; Jafari et al., 2016; Lazer & Friedman, 2007) in that it searches over the space of graph structures, rather than comparing a fixed set of hand-crafted ones.

In economics, a related line of work models how beliefs spread through a fixed social network, with agents repeatedly updating their opinions to a weighted average of their neighbours' (DeGroot, 1974; Golub & Jackson, 2010). This work characterises when such networks reach consensus, or converge to the truth, whereas ours optimises the network structure for the discovery of new knowledge.

Real-world knowledge-sharing networks. Several prior works examine the graph structures of real-world knowledge-sharing networks (Chen & Guan, 2010; Derex & Boyd, 2015; Eslami et al., 2013; Mason & Watts, 2012). There has also been work on intervening to improve the efficiency of real-world knowledge-sharing networks. Sciabolazza et al. (2020) selected academic researchers who, if connected to each other, would reduce the mean path length in the network. They then created the missing link by funding collaborative research between the pair in question, and found this improved research outcomes. These studies characterise, or intervene on, existing networks, whereas we ask which network structure would be optimal.

Optimising network structure. While some works have begun looking at gradient-based optimisation in specific, different applications, such as material discovery (Dold & Aranguren van Egmond, 2023) and solid and fluid mechanics (Zhao et al., 2024), such applications do not include knowledge-sharing. Existing methods for optimising the structure of a communication or social network employ discrete optimisation methods, such as genetic algorithms (Yu et al., 2021) or bespoke algorithms (Bhaskara et al., 2024; Li et al., 2021), but these are for the separate problem of optimising network robustness. Doherty et al. (2022) maximise algebraic connectivity (the second-smallest eigenvalue of the graph Laplacian) in the context of pose-graph SLAM in robotics, by selecting from a candidate set of edges on top of a fixed base. Unlike MAC, which optimises a spectral proxy for diffusion speed, we optimise the knowledge produced by the network, jointly with each node's research focus. More broadly, learning discrete graph structure by gradient descent on a continuous relaxation of the adjacency matrix has been studied in graph representation learning, for example to learn the edges that a graph neural network operates on (Franceschi et al., 2019) or to infer the interactions of a physical system from observed trajectories (Kipf et al., 2018). These methods fit a graph to data, whereas our method optimises a graph for the outcome of a simulated knowledge-sharing process. To our knowledge, there are no prior works showing a gradient-based method for optimising network structure for spread and discovery of knowledge. In contrast to all of the above, our method jointly

optimises the graph structure and each node's research focus across multiple subject areas, under a finite attention budget that trades off research against learning.

3 GLIDE: Gradient-Based Graph Optimisation

We now describe our optimisation method, which we term **G**radient **L**earning of **I**nferred **D**iscrete **E**dges (GLIDE); Figure 1 gives an overview. Assume we are given a set of nodes, $n_1, \dots n_m$, where each node n_i has a fixed capacity, c_i that indicates the budget of attention/energy it has available each time step. It can spend this budget on some mixture of learning from others it is connected to and conducting its own research. The knowledge possessed by each node is given by a positive real number for each of l different subject areas. We want to find connection weights that allow the fastest rate of knowledge discovery from the network as a whole, that is, we want to maximise the mean increase in knowledge score per time step across all nodes and subject areas.

During training, we relax the edge weights to the interval $[0, 1]$, and then a higher value for the edge weight e_{ij} increases the bandwidth for transferring knowledge from n_j to n_i, but decreases the rate at which n_i can discover its own new knowledge because of the cost of paying attention to n_j. At test time, the same trade off applies but the weights are discretised to $\{0, 1\}$. The continuous setting during training is analogous to assuming each agent can pay attention to, and learn from, any other to any degree. Discretisation assumes the decision of whether to pay attention to a given other node is binary. In reality, most settings will be somewhere between these two extremes, but considering the fully discrete case allows comparison to previous works studying a similar problem.

3.1 Formal Description

At each time step, there are two sources of knowledge gain: learning and research. The knowledge gain from learning is an additive factor quantified, for each of the l subject areas, as the difference between n_i's knowledge of that area and n_j's knowledge, multiplied by the weight of the connection between them, formally

$$\max_j e_{ij} \max(0, k_j - k_i) \,.$$

This means that, by learning from another, you partially catch up to them if they are ahead of you, by receiving a boost equal to (a scaled version of) the amount ahead they are. The knowledge gain from research, on the other hand, is a multiplicative factor quantified as the capacity reduced by the sum of all outgoing edge weights, times the relative focus of the given node on the given subject area, formally: $1 + \max(0, 1 - \sum_j A_{ij}/c_i) f_i$, where $f_i \in \Delta^{l-1}$ is a probability vector representing the focus weights for node i over the different subject areas.

We can write the graph structure in terms of an adjacency matrix $A \in \{0, 1\}^{m \times m}$ where the (i, j)th entry A_{ij} is 1 if n_i is paying attention to n_j, and 0 otherwise (or a matrix of reals between 0 and 1 at train time). Similarly, we can regard the amount of knowledge a node has in a subject area as a feature on that node, and write the features of each node as a matrix $X \in \mathbb{R}^{m \times l}$ where the (i, j)th entry X_{ij} is the jth feature of the ith node. We can also write the capacities as a vector $C \in \mathbb{R}^m$, whose ith entry C_i is the capacity of ith node, and the research focus of each node as a matrix of probability vectors $F \in (\Delta^{l-1})^m$. Then, we can write u_C, the update to the graph node features at each time step given capacities C, as

$$u_C(X, A, F) = S(X \cdot (1 + \max(0, C - A\mathbf{1}) \cdot F)) \,, \tag{1}$$

with $\mathbf{1}$ being the vector of all ones, the $\cdot$ operator indicating coordinate-wise multiplication (with broadcasting in the second instance), and S denoting the increase from knowledge sharing:

$$S(X)_{ij} = X_{ij} + \max_{k \neq i} \{A_{ik} \max(0, X_{kj} - X_{ij})\} \,.$$

Letting u^n represent the function u composed with itself n times,

$$u^n(x) = u(u \dots (x) \dots) \,, \tag{2}$$

the full optimisation problem can then be expressed as follows:

$$\operatorname*{argmax}_{A \in \{0,1\}^{m \times m}, F \in (\Delta^{l-1})^m} |u_C^n(X, A, F)| \,, \tag{3}$$

Graph	Knowledge score ↑	Avg. degree	Global Efficiency	CC	CC (W–S)
Reference graphs					
Complete	6.16 ± 1.43	1.00 ± 0.00	1.00 ± 0.00	3.60 ± 0.00	1.00 ± 0.00
Empty	$1.28 \times 10^8 \pm 3.42 \times 10^6$	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
Erdős–Rényi	$1.07 \times 10^8 \pm 5.41 \times 10^6$	0.20 ± 0.01	0.64 ± 0.01	3.06 ± 0.02	0.36 ± 0.01
SW (p_{flip}=0.01)	$5.70 \times 10^4 \pm 2.53 \times 10^4$	0.14 ± 0.00	0.39 ± 0.00	9.06 ± 0.05	0.49 ± 0.00
SW (p_{flip}=0.03)	$6.09 \times 10^4 \pm 2.75 \times 10^4$	0.14 ± 0.00	0.39 ± 0.00	8.79 ± 0.04	0.47 ± 0.00
SW (p_{flip}=0.06)	$6.38 \times 10^4 \pm 3.03 \times 10^4$	0.14 ± 0.00	0.40 ± 0.00	8.47 ± 0.06	0.46 ± 0.00
SW (p_{flip}=0.1)	$4.52 \times 10^7 \pm 9.51 \times 10^5$	0.14 ± 0.00	0.41 ± 0.01	7.06 ± 0.07	0.40 ± 0.00
Random (p=0.5)	$1.16 \times 10^6 \pm 2.13 \times 10^5$	0.75 ± 0.00	0.50 ± 0.00	3.51 ± 0.00	0.75 ± 0.00
Optimised graphs					
Greedy	$2.11 \times 10^8 \pm 3.50 \times 10^7$	0.49 ± 0.01	0.70 ± 0.02	3.50 ± 0.01	0.76 ± 0.01
MAC	$1.88 \times 10^8 \pm 1.74 \times 10^6$	0.08 ± 0.00	0.37 ± 0.00	0.79 ± 0.01	0.02 ± 0.01
GLIDE (ours)	$\mathbf{7.69 \times 10^8} \pm 9.07 \times 10^6$	0.20 ± 0.01	0.36 ± 0.01	2.69 ± 0.06	0.38 ± 0.02

Table 1: Final knowledge score, average degree (relative to the number of nodes), global efficiency, and clustering coefficient (CC; W–S: Watts–Strogatz definition) of the GLIDE-optimised graphs compared with other graph types. Values are means over 5 independent random seeds $\pm$ one std. The best knowledge score is in bold.

where $|Z|$ is the sum of all absolute values in Z, and $(\Delta^{l-1})^m$ is the m-fold product of the $(l-1)$-dimensional probability simplex.

During training, as well as relaxing the edge weights to continuous values, we also add some noise (from $\mathcal{U}(0, 0.1)$) to the updates to simulate the role of chance discovery in research. All of these operations are differentiable, so we can run gradient ascent on the adjacency weights A and the focus weights F to maximise the objective $|u_C^n(X, A, F)|$. In practice, we treat the focus weights as the unnormalised log probabilities, and so pass them through a softmax layer to normalise before treating as a distribution, as we find this improves stability of training.

3.2 Implementation Details

The graph edge weights are initialised uniformly at random in the interval $[0, 1]$. The focus weights logits are initialised randomly from a standard Normal distribution. The node capacities are drawn randomly from a Normal distribution with mean 20 and std. dev. 1. The knowledge scores at each iteration, during both training and testing, are initialised uniformly at random in the interval $[1, 1.3]$. We find that training is improved by setting a small, minimum threshold that edge weights need to exceed before any knowledge is transferred. This helps prevent the model learning many low-weight connections which would then be ignored at discretisation. We set this threshold to 0.01. Training runs for 30,000 iterations. The batch size is 1000 (so 3M total training examples) and the number of train time steps is 20, i.e., each iteration, 1000 times in parallel, we run 20 time steps of knowledge discovery and propagation and takes an ascent step with respect to the mean final knowledge score. Running for 20 time steps means that, with respect to Equation (2), $n = 20$. Gradient-based maximisation is performed using the Adam optimiser (Kingma & Ba, 2015) with initial learning rate $1e-3$ for the graph edge weights and $1e-2$ for the focus weight logits. We employ a scheduler that halves the learn rate at iterations 2,700, 4,200, 12,000, and 15,000. The core update process is expressed in PyTorch code in Appendix A. Total runtime is under 15min on a single RTX 2000. At evaluation, we run for 30 time steps with the thresholded adjacency matrix and real-valued focus weights.

3.3 Comparison to Graph Neural Networks

This optimisation procedure is similar to the training of GNNs (Gilmer et al., 2017; Scarselli et al., 2009). The differences are firstly that the edge weights are a single scalar for all features, whereas GNNs typically have a separate weight for each feature, and secondly that the value passed from node A to its neighbour B is the difference between the feature values for A and B (clipped at 0), whereas GNNs pass the feature value of A directly. The update from the focus vector, corresponding to the knowledge discovered via research by a given node, also does not have a standard GNNs counterpart.

Setting	Knowledge score ↑	Avg. degree	Global Efficiency	CC	CC (W–S)
GLIDE	6.24×10^8 $\pm2.43\times10^7$	0.18 ±0.01	0.36 ±0.01	3.72 ±0.21	0.32 ±0.03
GLIDE-unif-focus	55.29 ±0.04	0.18 ±0.01	0.36 ±0.01	3.72 ±0.21	0.32 ±0.03
GLIDE-rand-focus	145.03 ±15.36	0.18 ±0.01	0.36 ±0.01	3.72 ±0.21	0.32 ±0.03
Shuffled	3.98×10^8 $\pm5.27\times10^5$	0.18 ±0.01	0.36 ±0.01	3.09 ±0.25	0.26 ±0.06
Rewired	1.26×10^8 $\pm6.27\times10^6$	0.18 ±0.01	0.40 ±0.01	0.45 ±0.04	0.33 ±0.02

Table 2: Ablation studies showing the effect of removing or altering parts of GLIDE, on 20-node graphs. GLIDE-unif-focus and GLIDE-rand-focus replace the optimised focus weights of each node with a uniform and a random distribution across topics, respectively. Shuffled rewires the GLIDE graph while keeping its number of edges, so it has the same average degree as the GLIDE graph but otherwise a different structure. Values are means over 5 independent random seeds $\pm$ one standard deviation. CC: clustering coefficient; W–S: Watts–Strogatz definition.

4 Experimental Results

This section presents empirical measurements for several properties of the GLIDE graphs, and compares to that of other graph types in the literature. The results are presented in Table 1. For each graph type, we show results for a graph with 20, 30 and 40 nodes. 'Complete' is a fully-connected graph, 'Empty' is a graph with no edges, 'Erdős–Rényi' is a random graph with average degree equal to the average degree of the corresponding GLIDE graph, 'Small world (p_{flip})' is a small-world graph constructed according to the Watts and Strogatz (1998) method, with flip probability p_{flip}, 'Random (p=0.5)' is a graph formed by connecting each pair of nodes independently at random with probability 0.5, and MAC is the recent eigenvalue-based method described in Section 2 (candidates set 20% of edges selected randomly). We also compare to a simple greedy baseline we implement that begins with a random graph (independent edge probability 0.5) and then iteratively chooses 10 random edges and, for each, propagates a rollout of knowledge sharing for k steps (k=30, as in the main GLIDE experiments), and chooses whichever of the 10 gives the greatest improvement in final knowledge score. This continues until no improvements are found for 30 iterations. Results are computed for graph sizes $\{20, 30, 40\}$. The scores reported here are the mean across all sizes. All baselines get the optimised focus weights from GLIDE as replacing with random or uniform degrades results by orders of magnitude. Appendix E shows the full results.

The most important figure is in the first column, showing the average final knowledge score after the 30 evaluation time steps, which is the quantity the GLIDE graphs are trying to maximise. (Note that, as the number of time steps is fixed for all methods, this is proportional to the rate of knowledge discovery.) Those of GLIDE are consistently (t-test, $p << 0.01$) and significantly the highest. This validates the optimisation method itself, and shows that gradient-based optimisation on the continuous space of all weighted graphs, followed by discretisation of the weights to $\{0, 1\}$, is effective for solving the problem of optimal graph structure for knowledge discovery.

The graph structures found by GLIDE also have several other notable properties. Column two shows that they are sparse, having an average degree of $\sim 0.2n$, significantly different from the random graph it was initialised as before optimisation, which has average degree $0.5n$. Intuitively, this sparsity allows each node to devote more capacity to research.

Column three shows the global efficiency, computed as $\frac{1}{m(m-1)} \sum_i \sum_{j \neq i} \frac{1}{spl(n_i, n_j)}$, where $spl(x, y)$ is the shortest path length from node x to node y set to ∞ if they are disconnected. This roughly measures the inverse of the common average shortest path length measure, but can handle disconnected graphs. GLIDE optimised graphs have a low global efficiency (also see column three of Table 2). Of course the global efficiency tends to increase with degree, so for higher degree graphs such as random, the efficiency is higher. However, GLIDE has lower global efficiency even than others with lower degree, such as the small world graph (SWG). This is surprising as a short path length between nodes means it is easier for them to share knowledge with each other. It suggests either that short path length is actually less important for knowledge discovery than might be expected, or that other properties of GLIDE graphs compensate for not having short path length. See Appendix C for a visual example of this optimised structure.

Columns four and five show two different measures of the clustering coefficient. The first, denoted cc, is simply the percentage of all triplets that are closed, as defined in Newman et al. (2001). The second is the mean across nodes of the percentage of all triplets a given node is in that are closed. This is the definition used in Watts and Strogatz, 1998. See Appendix D for formal definitions. These are sometimes referred to as the global and local clustering coefficients, respectively.

Interestingly, the GLIDE graphs have a low clustering coefficient, which shows they are not small-world in structure. Previous work has often listed SWGs as one option that is potentially effective for knowledge sharing and discovery (Chen & Guan, 2010; Choi et al., 2010; Derex & Boyd, 2015; Eslami et al., 2013; Jafari et al., 2016; Lazer & Friedman, 2007; Mason & Watts, 2012). The comparison between small-world graphs and GLIDE graphs suggest that the latter achieves higher knowledge discovery rate partially by leveraging the increased efficiency gain from avoiding triplets.

4.1 Focus Weights

Nearly all nodes learn to concentrate their research on a single subject area, while the network as a whole covers every area. Figure 2 shows the learned focus weights (the F matrix in the terminology of Section 3.1) for a 20-node graph with 8 subject areas. The colour of the (i, j)th square shows the focus of node i on subject j. In most rows, the distribution is sharply peaked, with almost all the focus falling on a single subject. Quantitatively, the entropy of the optimised weights is significantly lower than would be expected by chance ($p << 0.01$ for all nodes, Monte Carlo vs Dirichlet samples with $\alpha = 1$).

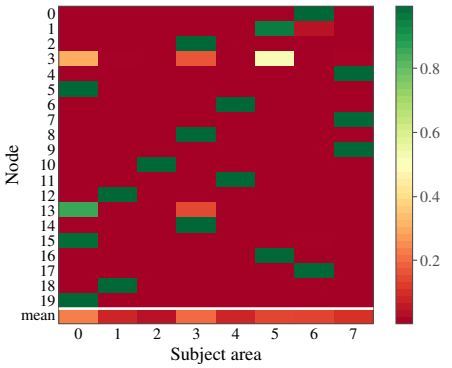


Figure 2: Learned focus weights for a 20-node GLIDE graph with 8 subject areas. The (i, j)th square shows the focus of node i on subject j. The bottom row shows the mean focus across all nodes for each subject.

This specialisation follows from the update rule. Research gains compound multiplicatively over time steps (Section 3.1), so concentrating a node's focus on one subject produces more knowledge than spreading it across several. The same incentive applies to a node with no neighbours, so specialisation does not itself depend on the graph structure. It also explains why replacing the learned focus weights with uniform or random ones reduces the knowledge score so sharply (Section 4.2). Peaked focus does not imply narrow knowledge. Focus determines only where a node directs its own research, and a node can still gain knowledge in other subjects by learning from its neighbours (Section 4.3).

Different nodes also specialise in different areas, so that, overall, each area receives roughly the same amount of focus. This is shown in the bottom row of Figure 2, where the distribution is roughly uniform. Quantitatively, if we take the mean focus vector across all nodes, its entropy is significantly greater than chance ($p << 0.01$, Monte Carlo vs Dirichlet samples with $\alpha = 1$), i.e., network learns to cover all areas roughly equally. Appendix F gives a formal explanation of this behaviour.

4.2 Ablation Studies

In order to quantify the contribution of the different parts of the GLIDE framework, we repeat the experiments from Table 1 on 20-node graphs with four different ablation settings. The first ablation setting, denoted 'GLIDE-unif-focus' uses the optimised graph structure, but focus weights that are uniform across subject areas. Similarly, the second, denoted 'GLIDE-rand-focus' uses the optimised graph structure but a random focus distribution (each focus weight sampled uniformly in $[0, 1]$). The third ablation setting keeps the focus weights but rewires the edges of the graph. This preserves the number of edges, and hence the average degree, but changes the rest of the graph structure, including the global efficiency and clustering coefficient, and destroys the relationship between the focus weights and the connectivity pattern. The fourth keeps the graph structure but relabels the nodes to destroy the relationship between the focus weights and the connectivity pattern.

All ablation settings show a drop in knowledge discovery rate. The biggest drop is in 'GLIDE-unif-focus' and 'GLIDE-rand-focus', which shows the learned distribution of focus weights is very important for the knowledge discovery rate. There is also a drop in the 'rewired' setting, which suggests that it is not just the focus distribution and the density of the graph that boost knowledge discovery rate, but how the edges are arranged relative to the each other. Interestingly, there is even a significant drop in the 'shuffled' setting, which shows the learned relationship between the focus weights and their position in the graph is important. The configuration learnt by GLIDE has thus learned to connect nodes that, in virtue of their focus distributions, would especially benefit from being connected, e.g., node A is deficient in subject area x, so it learns to pay attention to node B, which has a particularly high degree of knowledge in area x.

4.3 Research vs Learning

One feature that is evident from inspecting the GLIDE graphs (see Appendix C) is that there is a high variance across nodes in the number of outgoing connections, and that the nodes with the most outgoing connections tend also to be the nodes with the highest knowledge score. (Recall that the edge points in the direction of attention, so an edge from A to B indicates that A learns from B.) It turns out that both of these features are statistically significant: the distribution of in-degree across nodes is different from randomly assigning edges ($p < 0.001$, permutation test), and as shown in Figure 3, the correlation of mean knowledge with out degree is 0.290 and with in degree is -0.221 (both significant $p < 0.01$ permutation test). Thus, the structure of the GLIDE graphs is the following.

1. Some nodes focus their attention budget on research, meaning they have very low, often zero, out degree. These nodes tend to have lower than average knowledge score and there are many other nodes learning from them, meaning they have a high in degree.
2. Other nodes focus most of their attention budget on learning from others, meaning they have a high out degree. These nodes tend to have a higher than average knowledge score, and there are few other nodes learning from them, meaning they have a low in degree.

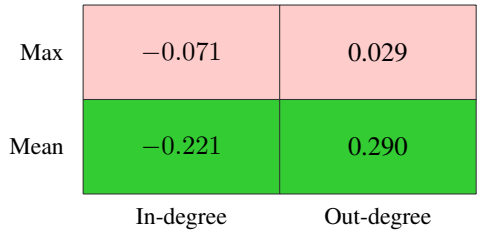


Figure 3: Rank correlation between in/out degree and mean/max knowledge score across subject areas. Green indicates significance, and red insignificance, at $p < 0.01$, under a permutation test.

It may be counter-intuitive that the research-focussed nodes, which have the highest in-degree, meaning that lots of other nodes are learning from them, actually have lower than average knowledge score. This, however, is due to the multiplicity of subject areas. As described in Section 4.1, the research-focussed nodes have learned to specialise in one or two areas only. They have a very high knowledge score for these areas, but a low score for every other area. The learner-focussed nodes, in contrast, have a reasonably high score in lots of areas, because they are learning from multiple different research-focussed nodes. This is shown in Figure 3, where we see that the out degree, which indicates how much nodes learn from other nodes, is strongly correlated with mean knowledge score across subject areas, but not with max knowledge score across subject areas. This suggests that learners do not excel at any one subject area, but instead achieve a reasonably high score across many subject areas.

Figure 3 also shows that the higher knowledge score of learners than researchers depends on how the knowledge across subject areas is aggregated. In particular, it applies when subject knowledge is aggregating by mean (or sum). If max is used as the aggregation function, as in the top row, then there is no significant difference between learners and researchers.

4.4 Rising Inequality Under Collective Optimisation

Figure 4 shows how inequality (coefficient of variation in knowledge score) changes over time under the considered graph structures. For all graph types considered, aside from the complete graphs, inequality grows over time. In other words, for any given graph structure, the more timesteps of research and learning take place, the higher the inequality becomes. This is because of the element of random noise in initial conditions: some nodes

begin with a slightly higher knowledge score by chance, and because the research-based increase in knowledge score is exponential, this small initial difference increases over time. The ability of nodes to learn from other nodes, in general, ameliorates this growing inequality, because when one nodes moves far ahead of another in knowledge score, the latter has an opportunity to make up the difference. Therefore, the denser the graph, the slower the rise in inequality, and for empty graphs, inequality rises quickly. The complete graph, which has maximum possible density, is the only one that resists and even reverses inequality.

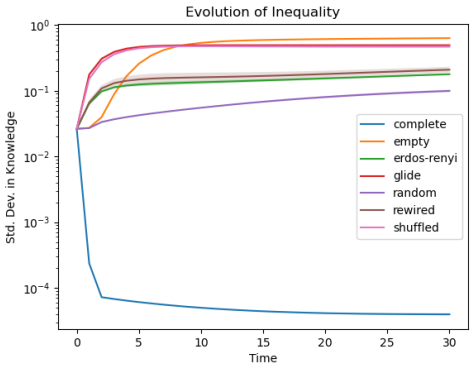


Figure 4: Evolution of inequality over time under various graph structures. Plotted values are the mean across five random seeds, shaded areas are plus/minus one standard deviation.

Interestingly, the GLIDE graphs, despite being non-empty, still have a fast rise in inequality, which initially is even faster than the empty graphs (though empty-graph inequality later slightly overtakes that of GLIDE). We conjecture that this is because of the dynamic described in Section 4.3, whereby some nodes devote their attention to discovering new knowledge (researchers), and some devote their attention to learning from others (learners), and the latter end up benefitting more. This difference in benefit applies at each time step, so over time the inequality between researchers and learners tends to increase, adds to the base rate arising from chance fluctuations compounding over time. Consequently, GLIDE graphs lead to the fastest rise in inequality of all graphs considered, other than empty graphs, including the shuffled GLIDE graphs.

Note that the rise in inequality is not due to some agents outcompeting others, because GLIDE does not model competition at all and instead maximises collective knowledge. It is interesting that, even though the graph is optimised solely for the collective good, as it might be in some top down network design, it still ends up propagating a rise in inequality.

4.5 Applying GLIDE to Larger Networks

The main results in Table 1 use graphs between 20 and 40 nodes. In its current form, GLIDE does not perform well on larger graph sizes. However, a simple approach to applying it to larger graph sizes is to zero out most of the edges in the adjacency matrix. In particular, we train only blocks along the diagonal. As long as there is no requirement for graphs to be connected, this allows the method to scale essentially unboundedly, as the blocks can be trained independently and then pieced together to form the full adjacency matrix. Table 3 shows the results for a 1000 node network with block size 30, where GLIDE still produces much more effective graph configurations than baselines. We also measured a version where blocks overlapped so the graph could remain connected, and found the knowledge score was the same as in Table 3.

5 Discussion

Our experiments help to validate gradient-based optimisation for exploring configurations of social graphs, and open the door to future research. Additionally, our optimisation setting allows a simple way to include the finite attention budget of each node, and hence a trade-off between learning and research.

SWGs are often discussed in the context of efficient knowledge sharing, and prior work has found mixed evidence for their effectiveness (Chen & Guan, 2010; Choi et al., 2010; Derex & Boyd, 2015; Eslami et al., 2013; Jafari et al., 2016; Lazer & Friedman, 2007). Our work suggests it is possible to find much more efficient configurations than SWG or other common named graph types. The graphs discovered by GLIDE do not appear to have a particular named structure. The properties they exhibit are sparsity, low global efficiency, and small clustering coefficient. Small clustering coefficient differs most notably from SWGs, but it is in fact intuitive that this should be low, because if A is learning from B and B from C, then there is already a reasonably short path by which knowledge discovered

Graph	Knowledge score	Avg. degree	global efficiency	CC	CC (W–S)
GLIDE	6.54×10^{8} $\pm3.53\times10^{5}$	4.01×10^{-3} ±0.00	6.86×10^{-3} ±0.00	2.33 ±0.00	0.24 ±0.00
SW (p_{flip}=0.01)	3.19×10^{7} $\pm2.33\times10^{5}$	4.00×10^{-3} ±0.00	4.83×10^{-2} $\pm7.99\times10^{-3}$	9.10 ±0.11	0.49 $\pm3.94\times10^{-3}$
SW (p_{flip}=0.03)	3.45×10^{7} $\pm3.64\times10^{5}$	4.00×10^{-3} ±0.00	8.17×10^{-2} $\pm1.50\times10^{-3}$	8.46 $\pm3.26\times10^{-3}$	0.46 $\pm2.64\times10^{-3}$
SW (p_{flip}=0.06)	3.89×10^{7} $\pm1.03\times10^{6}$	4.00×10^{-3} ±0.00	0.11 $\pm4.19\times10^{-3}$	7.41 ±0.15	0.41 $\pm6.85\times10^{-3}$
SW (p_{flip}=0.1)	4.38×10^{7} $\pm8.11\times10^{5}$	4.00×10^{-3} ±0.00	0.13 $\pm2.68\times10^{-3}$	6.49 ±0.18	0.37 $\pm1.00\times10^{-2}$
MAC	3.04×10^{4} $\pm9.70\times10^{3}$	0.02 ±0.00	0.41 ±0.00	0.09 ±0.00	0.02 ±0.00

Table 3: Applying GLIDE to a 1000 node graph by independently optimising blocks of size 30 on the diagonal of the adjacency matrix. It still outperforms comparison graphs.

by C can reach A. If A was going to spend additional attention budget learning from another node, then it would be better to choose a node other than C.

Also interesting is the fact that research-like nodes who spend their focus discovering new knowledge reach significantly lower mean knowledge score than learner-like nodes, who focus on learning the knowledge held by others. One real-world instance where a similar trend can be observed is the high citation count of academic review papers, compared to those proposing novel ideas (Miranda & Garcia-Carpintero, 2018). Insofar as citation count indicates knowledge content, this suggests it is easier to accrue useful knowledge by pulling from the works of others, rather than by discovering new ideas oneself. Similarly, in industry R&D, companies that build the most profitable products tend to be those that combine multiple existing ideas (Hargadon & Sutton, 1997; Taylor & Greve, 2006), rather than the original inventors (Griliches, 1991; Mansfield, 1991; Teece, 1986). All new knowledge ultimately comes from research, with learning merely spreading around existing knowledge. In our simulations, the complete graphs, which maximise learning and minimise research perform by far the worst. However, in our simulations, the learner nodes can achieve a higher mean knowledge score than the research nodes who actually produce the knowledge.

The advantage of learners over researchers also feeds into a more general trend for knowledge inequality to rise under GLIDE graphs (see Section 4.4). There is a very wide literature on rich-get-richer forces in the economy and social interaction (Atkinson, 2002; Glaeser, 2005), most of which are driven by other forces than knowledge sharing. Indeed as shown in Section 4.4, one might expect that knowledge sharing counter-acts inequality by allowing those behind to catch up. However, Section 4.4 showed that, under the optimised graphs, inequality still rises quickly. Initially the rise is even faster than for empty graphs. We suspect this is because the graph structure allows the learner nodes to learn the knowledge discovered by researcher nodes and move ahead of them on mean score. In the economic context, Eeckhout and Jovanovic (2002) have made a similar argument that knowledge transfer can actually increase inequality by enabling free-loading.

Our study has two main limitations. The results are purely simulated, with no validation against real-world networks, and we scale to larger graphs only by training independent blocks that are not connected to each other (Section 4.5). Future work should also explore agents optimising selfishly, as that fits many real-world cases better than a shared global objective.

6 Conclusion

This paper presented a model for optimising the graph structure, and individual research focus, for a network of interacting agents that discover and share knowledge with each other. Using gradient-based optimisation, and in particular, a setting related to the training of graph neural networks, we produce a set of edges and, for each agent, a distribution over a fixed number of subject areas, so as to maximise the total amount of knowledge across all nodes. The success of gradient-based optimisation for this discrete task opens possibilities for future work on optimising network structure for other metrics, such as robustness or path length. We then examined the optimised graphs, and found they are sparse, with low global efficiency, and a low clustering coefficient. The low clustering coefficient sets them apart from small-world graphs, which prior works have examined as a structure for knowledge sharing, with mixed findings. Finally, we showed how nodes in the optimised graphs who focus more on conducting research, though essential to knowledge discovery, end up with a lower mean knowledge score than nodes who focus on learning from others, and that this inequality between learners and researchers contributes to growing inequality in the network over time. This could pave the way for future explorations of inequality and free-loading in social networks.

Acknowledgements

We gratefully acknowledge the support of the UK Engineering and Physical Sciences Research Council (grant EP/W002876/1).

References

Atkinson, T. (2002). Is rising income inequality inevitable? a critique of the 'Transatlantic Consensus'. In P. Townsend & D. Gordon (Eds.), *World poverty: New policies to defeat an old enemy* (pp. 25–52). Policy Press.

Bhaskara, A., Crane, A., Jain, S., Mazumder, M. M. H. U., Sullivan, B. D., & Yalamanchili, P. (2024). Optimizing probabilistic propagation in graphs by adding edges. *arXiv preprint arXiv:2407.02624*.

Chen, Z., & Guan, J. (2010). The impact of small world on innovation: An empirical study of 16 countries. *Journal of Informetrics*, *4*(1), 97–106.

Choi, H., Kim, S.-H., & Lee, J. (2010). Role of network structure and network effects in diffusion of innovations. *Industrial Marketing Management*, *39*(1), 170–177.

DeGroot, M. H. (1974). Reaching a consensus. *Journal of the American Statistical Association*, *69*(345), 118–121. https://doi.org/10.1080/01621459.1974.10480137

Derex, M., & Boyd, R. (2015). The foundations of the human cultural niche. *Nature Communications*, *6*(1), 8398.

Doherty, K. J., Rosen, D. M., & Leonard, J. J. (2022). Spectral measurement sparsification for pose-graph slam. *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 01–08.

Dold, D., & Aranguren van Egmond, D. (2023). Differentiable graph-structured models for inverse design of lattice materials. *Cell Reports Physical Science*, *4*(10), 101586. https://doi.org/10.1016/j.xcrp.2023.101586

Eeckhout, J., & Jovanovic, B. (2002). Knowledge spillovers and inequality. *American Economic Review*, *92*(5), 1290–1307.

Eslami, H., Ebadi, A., & Schiffauerova, A. (2013). Effect of collaboration network structure on knowledge creation and technological performance: The case of biotechnology in Canada. *Scientometrics*, *97*(1), 99–119.

Franceschi, L., Niepert, M., Pontil, M., & He, X. (2019). Learning discrete structures for graph neural networks. *Proceedings of the 36th International Conference on Machine Learning*.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. *Proceedings of the 34th International Conference on Machine Learning*, *70*, 1263–1272.

Glaeser, E. L. (2005). *Inequality* (NBER Working Paper No. 11511). National Bureau of Economic Research. https://doi.org/10.3386/w11511

Golub, B., & Jackson, M. O. (2010). Naïve learning in social networks and the wisdom of crowds. *American Economic Journal: Microeconomics*, *2*(1), 112–149. https://doi.org/10.1257/mic.2.1.112

Griliches, Z. (1991). *The search for R&D spillovers* (NBER Working Paper No. 3768). National Bureau of Economic Research. https://doi.org/10.3386/w3768

Hargadon, A., & Sutton, R. I. (1997). Technology brokering and innovation in a product development firm. *Administrative Science Quarterly*, *42*(4), 716–749.

Jafari, A., Shariatpanahi, S. P., Zolfagharzadeh, M. M., & Mohammadi, M. (2016). The effect of network structure on innovation initiation process: An evolutionary dynamics approach. *arXiv preprint arXiv:1604.04758*.

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations*.

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., & Zemel, R. (2018). Neural relational inference for interacting systems. *Proceedings of the 35th International Conference on Machine Learning*.

Lazer, D., & Friedman, A. (2007). The network structure of exploration and exploitation. *Administrative Science Quarterly*, *52*(4), 667–694.

Li, Z., Liu, Q., & Chen, L. (2021). A novel edge rewire mechanism based on multiobjective optimization for network robustness enhancement. *Frontiers in Physics*, *9*, 735998.

Mansfield, E. (1991). Social returns from R&D: Findings, methods and limitations. *Research-Technology Management*, *34*(6), 24–27.

Mason, W., & Watts, D. J. (2012). Collaborative learning in networks. *Proceedings of the National Academy of Sciences*, *109*(3), 764–769.

Milgram, S. (1967). The small world problem. *Psychology Today*, *2*(1), 60–67.

Miranda, R., & Garcia-Carpintero, E. (2018). Overcitation and overrepresentation of review papers in the most cited papers. *Journal of Informetrics*, *12*(4), 1015–1030. https://doi.org/10.1016/j.joi.2018.08.006

Newman, M. E., Strogatz, S. H., & Watts, D. J. (2001). Random graphs with arbitrary degree distributions and their applications. *Physical Review E*, *64*(2), 026118.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, *20*(1), 61–80. https://doi.org/10.1109/TNN.2008.2005605

Sciabolazza, V. L., Vacca, R., & McCarty, C. (2020). Connecting the dots: Implementing and evaluating a network intervention to foster scientific collaboration and productivity. *Social Networks*, *61*, 181–195.

Taylor, A., & Greve, H. R. (2006). Superman or the fantastic four? knowledge combination and experience in innovative teams. *Academy of management Journal*, *49*(4), 723–740.

Teece, D. J. (1986). Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy. *Research Policy*, *15*(6), 285–305.

Watts, D. J., & Strogatz, S. H. (1998). Collective dynamics of 'small-world' networks. *Nature*, *393*(6684), 440–442.

Yu, P., Shi, Y., Wang, L., Ke, S., & Yin, M. (2021). A method for optimizing communication network topology based on genetic algorithm. *International Conference on Wireless Communications and Applications*, 30–40.

Zhao, Y., Li, H., Zhou, H., Attar, H. R., Pfaff, T., & Li, N. (2024). A review of graph neural network applications in mechanics-related domains. *Artificial Intelligence Review*, *57*(11), 315.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: See results and methods sections.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section 3 and the attached code.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Attached as tarball.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section 3.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section 3.

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

   Answer: [Yes]

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [N/A]

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

A Simplified Method Pytorch Code

```
def propagate(A, I, n_steps, bs):
    X = torch.ones(bs, N_NODES, N_FEATS)
    X += torch.rand_like(X)*0.3
    A_eff = (A - 0.01).clip(min=0, max=1)
    base_capacities = 20 + torch.randn(N_NODES)
    x_capacities = (1 - A.clip(min=0).sum(axis=1)/base_capacities).clip(min=0)
    improve_rates = 1 + x_capacities[:, None]*I.softmax(axis=1)
    for step in range(n_steps):
        X *= improve_rates
        X += torch.rand_like(X)
        diffs = F.relu(X[:,None] - X[:,:,None])
        X += (A_eff[:,:,None]*diffs).max(axis=2).values
    return X.mean(axis=0)

edge_weights = nn.Parameter(torch.rand(N_NODES, N_NODES))
focus_weights = nn.Parameter(torch.randn(N_NODES, N_FEATS))
opt = torch.optim.Adam([edge_weights], lr=1e-3)
opt.add_param_group({'params': focus_weights, 'lr': 1e-2})
```

```
for i in range(N_ITERATIONS):
    avg_knowledge = propagate(edge_weights, focus_weights, 20, bs=1000)
    loss = -avg_knowledge.sum()
    loss.backward()
    opt.step(); opt.zero_grad()
edge_weights = (edge_weights > 0.5).float()
```

B Significance Calculation

Here we show the statistical significance of the differences in mean knowledge score from Table 1. We need only show this for the MAC graphs because it is closer to GLIDE than any other baseline. For this comparison, we compute an independent one-sided t-test. The pooled standard deviation is

$$s = \sqrt{\frac{(9.07e6)^2 + (1.74e6)^2}{2}} \approx 6.53e6\,.$$

The t-score is then

$$\frac{7.69e8 - 1.88e8}{6.53e6\sqrt{\frac{5}{2}}} \approx 56.27\,.$$

The corresponding p-value is less than $1e-6$.

C Qualitative Examples of Discovered Graphs

Figure 5 of the optimised graph structures discovered by GLIDE and its comparison with the structure of a small-world graph. For readability, we show a graph with only 12 nodes.

D Formal Definition of Clustering Coefficient Metrics

Section 4 showed two different measures of clustering coefficient. The first, denoted cc, is simply the percentage of all triplets that are closed, as defined in Newman et al. (2001). That is, for graph $G = (V, E)$,

$$cc = \frac{100 \times |\{(n_1, n_2, n_3) \in V^3 : (n_1, n_2) \in E \wedge (n_2, n_3) \in E \wedge (n_3, n_1) \in E\}|}{|\{(n_1, n_2, n_3) \in V^3 : (n_1, n_2) \in E \wedge (n_2, n_3) \in E\}|}. \tag{4}$$

The second is the mean across nodes of the percentage of all triplets a given node is in that are closed. This is the definition used in Watts and Strogatz, 1998. Formally

$$\frac{100}{|V|} \sum_{v \in V} \frac{|\{(v, n_2, n_3) \in V^3 : (v, n_2) \in E \wedge (v, n_3) \in E \wedge (n_2, n_3) \in E\}|}{|\{(v, n_2, n_3) \in V^3 : (v, n_2) \in E \wedge (v, n_3) \in E\}|}\,. \tag{5}$$

E Extended Results

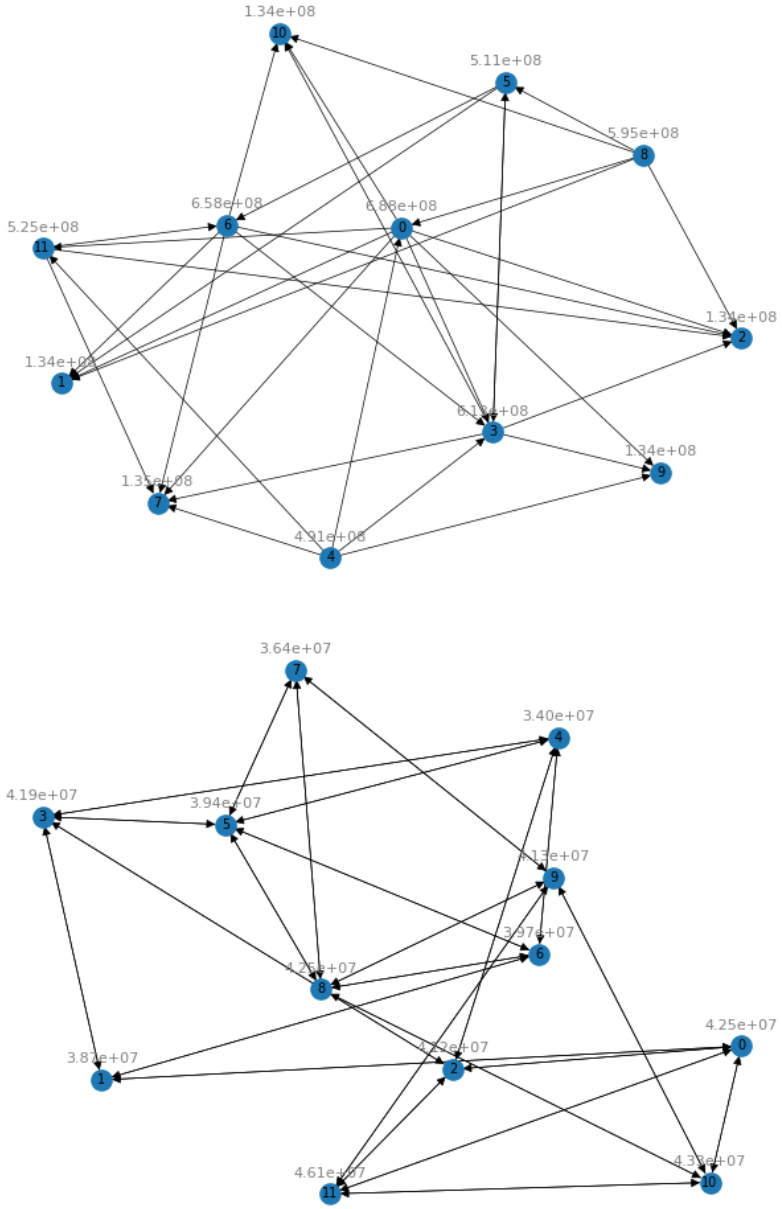


Figure 5: Examples of different graph structures. The grey figure above each node is its final knowledge score after 20 iterations, averaged over 1000 random seeds. By convention, we write the arrow direction to show the direction of attention flow, so an arrow pointing from A to B means that A is spending attention budget to learn from B, and knowledge is moving from B to A. For readability, we show only 12 nodes, but the quantitative experiments used 10-40. **Top:** GLIDE. Nodes 2, 7, 9 and 10 are examples of researcher nodes, while nodes 0, 4 and 8 are examples of learner nodes. **Bottom:** Small world graph, formed from the Watts-Strogatz method with flip probability 0.1. Compared to GLIDE, it has lower mean knowledge score, and smaller variance in degree and knowledge scores across nodes.

	knowledge-score-mean	avg-degree	global-efficiency	cc	cc-watts-strogatz
complete-n20	10.16 (4.29)	1.00 (0.00)	1.00 (0.00)	4.99 (0.00)	1.00 (0.00)
complete-n30	4.15 (0.00)	1.00 (0.00)	1.00 (0.00)	3.33 (0.00)	1.00 (0.00)
complete-n40	4.18 (0.00)	1.00 (0.00)	1.00 (0.00)	2.50 (0.00)	1.00 (0.00)
complete-n50	4.20 (0.00)	1.00 (0.00)	1.00 (0.00)	2.00 (0.00)	1.00 (0.00)
empty-n20	1.45e+08 (6.61e+06)	0.00 (0.00)	0.00 (0.00)	nan (nan)	0.00 (0.00)
empty-n30	1.42e+08 (8.13e+06)	0.00 (0.00)	0.00 (0.00)	nan (nan)	0.00 (0.00)
empty-n40	9.69e+07 (6.95e+06)	0.00 (0.00)	0.00 (0.00)	nan (nan)	0.00 (0.00)
empty-n50	5.99e+05 (1.98e+05)	0.00 (0.00)	0.00 (0.00)	nan (nan)	0.00 (0.00)
er-n20	1.79e+08 (1.14e+07)	0.17 (0.00)	0.71 (0.01)	4.04 (0.02)	0.33 (0.01)
er-n30	1.34e+08 (9.44e+06)	0.15 (0.00)	0.53 (0.00)	2.77 (0.04)	0.28 (0.01)
er-n40	7.04e+06 (4.10e+06)	0.27 (0.03)	0.69 (0.01)	2.36 (0.01)	0.48 (0.04)
er-n50	4.20 (0.00)	0.94 (0.01)	0.69 (0.01)	2.00 (0.00)	1.00 (0.00)
swg-flip0.01-n20	4.81e+07 (1.79e+06)	0.20 (0.00)	0.48 (0.00)	7.30 (0.19)	0.41 (0.01)
swg-flip0.01-n30	4.87e+07 (1.74e+06)	0.13 (0.00)	0.37 (0.00)	6.92 (0.11)	0.39 (0.00)
swg-flip0.01-n40	3.89e+07 (1.36e+06)	0.10 (0.00)	0.31 (0.00)	6.94 (0.13)	0.39 (0.01)
swg-flip0.01-n50	1.35e+05 (37377.62)	0.08 (0.00)	0.31 (0.00)	6.80 (0.08)	0.38 (0.00)
swg-flip0.03-n20	5.53e+07 (1.30e+06)	0.20 (0.00)	0.48 (0.00)	5.84 (0.17)	0.33 (0.01)
swg-flip0.03-n30	5.60e+07 (1.96e+06)	0.13 (0.00)	0.38 (0.00)	5.30 (0.08)	0.31 (0.00)
swg-flip0.03-n40	4.68e+07 (1.76e+06)	0.10 (0.00)	0.32 (0.00)	5.33 (0.13)	0.31 (0.01)
swg-flip0.03-n50	1.51e+05 (43372.84)	0.08 (0.00)	0.32 (0.00)	5.01 (0.06)	0.29 (0.00)
swg-flip0.06-n20	6.12e+07 (1.54e+06)	0.20 (0.00)	0.48 (0.00)	4.93 (0.09)	0.28 (0.00)
swg-flip0.06-n30	6.29e+07 (2.36e+06)	0.13 (0.00)	0.38 (0.00)	4.34 (0.07)	0.26 (0.00)
swg-flip0.06-n40	5.20e+07 (1.76e+06)	0.10 (0.00)	0.33 (0.00)	4.23 (0.07)	0.25 (0.00)
swg-flip0.06-n50	1.68e+05 (48385.50)	0.08 (0.00)	0.33 (0.00)	3.95 (0.06)	0.23 (0.00)
swg-flip0.1-n20	6.59e+07 (1.43e+06)	0.20 (0.00)	0.49 (0.00)	4.41 (0.07)	0.26 (0.00)
swg-flip0.1-n30	6.80e+07 (2.64e+06)	0.13 (0.00)	0.39 (0.00)	3.69 (0.06)	0.22 (0.00)
swg-flip0.1-n40	5.77e+07 (1.84e+06)	0.10 (0.00)	0.34 (0.00)	3.50 (0.07)	0.21 (0.00)
swg-flip0.1-n50	1.76e+05 (53444.72)	0.08 (0.00)	0.34 (0.00)	3.21 (0.04)	0.19 (0.00)
true-rand-n20	3.33e+06 (6.27e+05)	0.50 (0.00)	0.75 (0.00)	4.81 (0.00)	0.75 (0.00)
true-rand-n30	1.43e+05 (27493.64)	0.50 (0.00)	0.75 (0.00)	3.26 (0.00)	0.75 (0.00)
true-rand-n40	3456.36 (1662.89)	0.50 (0.00)	0.75 (0.00)	2.45 (0.00)	0.75 (0.00)
true-rand-n50	6.85 (1.01)	0.50 (0.00)	0.75 (0.00)	1.97 (0.00)	0.75 (0.00)
MAC-n20	1.85e+08 (0.00)	0.11 (0.00)	0.38 (0.00)	1.88 (0.00)	0.04 (0.00)
MAC-n30	1.88e+08 (4.58e+06)	0.08 (0.00)	0.37 (0.00)	0.00 (0.00)	0.00 (0.00)
MAC-n40	1.92e+08 (6.36e+05)	0.07 (0.00)	0.35 (0.00)	0.49 (0.02)	0.03 (0.01)
MAC-n50	1.77e+08 (9.91e+05)	0.06 (0.00)	0.36 (0.00)	0.39 (0.06)	0.01 (0.01)
glide-n20	6.24e8 (2.43e7)	0.18 (0.01)	0.36 (0.01)	3.72 (0.21)	0.32 (0.03)
glide-n30	7.76e+08 (1.90e+07)	0.15 (0.01)	0.39 (0.02)	2.29 (0.06)	0.25 (0.01)
glide-n40	8.87e+08 (1.84e+07)	0.28 (0.03)	0.70 (0.04)	2.48 (0.01)	0.60 (0.04)
glide-n50	2.68e+07 (9.70e+06)	0.96 (0.01)	1.00 (0.00)	2.00 (0.00)	1.00 (0.00)

Table 4: Degree (relative to the number of nodes), global efficiency, total final knowledge, and clustering coefficient, for the GLIDE-optimised graphs, in comparison with several other graph types. The suffix indicates the number of nodes in the graph, e.g., 'er-n20' indicates an Erdős–Rényi random graph with 20 nodes. Figures are the mean of 5 independent random seeds, with std. dev. in parentheses.

F Formal Explanation of Uniform Coverage of Subject Areas

Following the notation of the paper, let X_{ij} be the knowledge score of agent i in subject area j, and consider a fixed agent u and subject s.

We compute the expected effect of a marginal increase in X_{us}. For the agent itself, this marginal increase is simply the identity:

$$\frac{\partial b_j}{\partial b_j} = 1.$$

For $k \neq i$,

$$\frac{\partial}{\partial b_j} \max(0, c_{ki} - b_j)\, A_{kj} = \begin{cases} 0 & b_j > c_{ki} \\ -A_{kj} & \text{otherwise.} \end{cases}$$

This can be written using an indicator variable as

$$\frac{\partial}{\partial b_j} \max(0, c_{ki} - b_j)\, A_{kj} = -A_{kj}\, \mathbb{1}(b_j < c_{ki}).$$

The expectation of an indicator variable is equal to the probability of the event it indicates, and since $A_{kj} \geq 0$, the probability $P(b_j < c_{ki})$ is increasing in c_{ki}, and so $-A_{kj}\, P(b_j < c_{ki})$ is decreasing in c_{ki}:

$$\frac{\partial}{\partial c_{ki}} - A_{kj}\, \mathbb{1}(b_j < c_{ki}) = \frac{\partial^2}{\partial b_j\, \partial c_{ki}} \max(0, c_{ki} - b_j)\, A_{kj} < 0.$$

The gradient of c_{ki} with respect to the focus weight F_{ij} is 1 for all i, j, so by the chain rule,

$$\frac{\partial^2}{\partial F_{ij}\, \partial F_{kj}} \max(0, c_{ki} - b_j)\, A_{kj} < 0 \qquad \forall\, k \neq i,$$

i.e., the off-diagonal entries of the Jacobian are negative. This shows that, for each j, the F_{ij} compete with one another: for a given agent, whichever j currently has the lowest F_{ij} ($i \neq l$), will exhibit the highest gradient, encouraging each agent to focus on a subject that is not already being focused on, which pushes the mean of all focus weights across agents toward uniformity across subjects.

G Robustness to Alternative Training Parameters

In this section, we show the effect of varying the values of the training parameters for GLIDE. The three such parameters mentioned in Section 3 are the method of knowledge transfer (max across outgoing edges), the initialisation of knowledge scores and the initialisation of capacities.

Both the initialisation parameters of capacities, which is N(20,1), and the knowledge scores at each iteration, which is U(1,1.3), are scale invariant, meaning we could multiply each by a constant and training dynamics would be unchanged (up to a learning rate scale in the latter case). Thus, they each have one free parameter, and the noise ratio for capacities is 1/20. The table below investigates sensitivity to these as well as changing the update rule to a blend of mean and max of neighbours.

The results show that GLIDE is robust to varying the knowledge score initialisation: changing from 0.3 (as reported in the paper) to 0.2 or 0.5 does not substantively affect the final score, and changing to 0.4 actually improves the final score. For the mean of the capacities initialisation, GLIDE performance is more sensitive. It degrades slightly on changing from 20 (as reported in the paper) to 25 or 30, though still remains significantly above all comparison methods. When changed to 15, it degrades more substantially. There it falls slightly below one of the comparison methods, MAC, though is still substantially better than all others. The shape of the graphs, which informs the discussion and analysis in the paper, is the same under all of these settings: sparse, low global efficiency and low clustering coefficient.

Changing the neighbour update rule, gives very similar performance, still outperforming any alternative methods.

	knowledge-score	avg-degree	global efficiency	cc	cc-watts-strogatz
GLIDE main (k-init 0.3, cap 20)	6.24×10^8 $\pm2.43\times10^7$	0.18 ±0.01	0.36 ±0.01	3.72 ±0.21	0.32 ±0.03
k-init 0.2	7.63e+08	0.21	0.38	2.55	0.61
k-init 0.5	7.89e+08	0.20	0.36	2.54	0.77
cap 15	1.47e+08	0.95	0.46	2.50	1.00
cap 25	7.24e+08	0.11	0.15	1.75	0.24
cap 30	7.03e+08	0.10	0.20	1.65	0.23
mean-max	7.95e+08	0.38	0.29	2.15	0.58

Table 5: Final knowledge score in GLIDE-discovered graphs for various values of the noise in knowledge initialisation (k-init), and in capacities across nodes (cap). The bottom row, 'mean-max' changes the aggregation to be the sum of the mean and max across neighbours cf. main results which use max only.

	knowledge-score	avg-degree	global efficiency	cc	cc-watts-strogatz
cont	6.24×10^8 $\pm2.43\times10^7$	0.18 ±0.01	0.36 ±0.01	3.72 ±0.21	0.32 ±0.03
thresh0.1	7.91e+08	0.19	0.34	2.56	0.78
thresh0.2	7.88e+08	0.19	0.34	2.56	0.74
thresh0.3	7.76e+08	0.19	0.34	2.56	0.69
thresh0.4	7.72e+08	0.19	0.34	2.56	0.65
thresh0.5	7.71e+08	0.19	0.34	2.55	0.60
thresh0.6	7.69e+08	0.19	0.34	2.53	0.54
thresh0.7	7.68e+08	0.19	0.34	2.50	0.50
thresh0.8	7.69e+08	0.19	0.34	2.45	0.44
thresh0.9	7.69e+08	0.19	0.34	2.43	0.42

Table 6: Effect of varying the discretisation threshold.

H Measuring the Effect of Discretisation

The approach in GLIDE is to optimise edge weights in continuous space and then threshold to give discrete (binary) edges as desired. The main experiments use 0.5 as the threshold. Here, we present results using different threshold values to measure whether the system is sensitive to this choice of threshold.

Table 6 repeats the main results with varying thresholds and also reported the score for the continuous, non-thresholded version as used during training. The average degree reduces with increasing threshold but the overall knowledge score remains broadly the same.

These results suggest that the network learns to put continuous edge weights either close to 1 or close to 0 so the thresholding makes little difference to the overall knowledge discovery rate.