Predicting the effect of drug combinations: an Active Learning Approach

Structure of the Dataset

A graph $G = (V, E)$ where the vertices are either drugs or proteins: $V = V_d \cup V_p$.

- If a vertex $v \in V_d$ is a drug, it is associated with a feature representation $x^v \in \mathbb{R}^k$ containing the *Morgan fingerprint* of the drug. We generated fingerprints with $nBits = 1024$ and $radius = 4$
- If a vertex $v \in V_p$ is a protein, there is no feature available. We set the feature representation $x^v = 0_k$
- The edges E represent drug-drug interactions, drug-protein interactions and protein-protein interactions. We can consider these edges to be of three types $E = E_{dd} \cup E_{dp} \cup E_{pp}$.
- Each edge is associated with a feature vector $x^e \in \mathbb{R}^4$. If an edge corresponds to a drug-drug interaction, $e \in E_{dd}$, x^e contains the 4 drug synergy scores, otherwise $x^e = 0_4$.

Proposed pipeline

Let us restrict to only one the drug synergy scores for now (*e.g.* HSA)

Model

Initialize node embeddings $h_0^v \in \mathbb{R}^n$. A typical choice for n would be $n \approx 10$. Another choice can be to initialize embeddings based on the eigenvectors of the Laplacian operator.

Initialize matrices $M_{p \leftrightarrow p}$, $M_{d \to p}$, $M_{p \to d}$, M_x. In what follows, σ denotes any activation function such as ReLU or Sigmoid. λ would be chosen < 1.

for T time steps, iterate:

if $v \in V_d$:

$$h_{t+1}^v = \sigma(M_x x^v + \sum_{\omega \in N(v) \cap V_p} M_{p \to d} h_t^\omega) + \lambda h_t^v$$

Note: we could use concatenation instead of sum between node and neighbour's messages

if $v \in V_p$:

$$h_{t+1}^v = \sigma(\sum_{\omega \in N(v) \cap V_d} M_{d \to p} h_t^\omega + \sum_{\omega \in N(v) \cap V_p} M_{p \leftrightarrow p} h_t^\omega) + \lambda h_t^v$$

After T time steps:

Predict the drug synergy score with a function $f_\phi(h_T^{v1}, h_T^{v2}, x^{v1}, x^{v2})$. We can choose something simple at first: $f_\phi = h_T^{v1} G h_T^{v2}$. Note: in the case where G has negative eigenvalues, drugs with opposite embeddings will have a high predicted synergy score.

Notes:

- If T is big, it may not be tractable to backprop through all time steps. We should investigate how people have dealt with this in practice, maybe there are readily applicable solutions in *pytorch geometric*.
- We can add constraints on the matrices to make the update rule a contractive map. The embeddings will then converge to the unique fixed point of the contractive map (independent of the initialization of the embeddings). See *Banach's fixed point theorem*
- We can choose T dynamically, and choose to stop when we witness that the embeddings h have almost converged

Active Learning

Let Q denote the set of drug pairs that we have queried so far. We can choose this set to be non empty at first, and *pretrain* the model with these pairs.

Bayesian Learning:

We want to get N samples of predicted scores for each drug pair by introducing noise in the model and performing several forward passes (in the spirit of Monte Carlo dropout). Two proposed ways to do this:

- 1 forward pass in the GNN + N forward passes in f_ϕ with different dropout configurations
- N forward passes in the GNN with different random initializations of the embeddings h.

Acquisition function

We use *Expected Improvement*

Let s_+ be the best score *seen* so far among all drug pairs. For each drug pair dd, we get N samples of predicted synergy scores $s_{dd}^1, \ldots, s_{dd}^N$.

The Expected Improvement for the drug pair dd is:

$$EI = \mathbb{E} \max(s_{dd} - s_+, 0)$$

It can be approximated by:

$$\hat{EI} = \frac{1}{N} \sum \sigma(s_{dd}^i - s_+)$$

where σ denotes the ReLU function.

Then we query the pairs with highest (one at a time or several at a time) Expected Improvement $\hat{EI}$. The query at iteration j is Q_j.

We expend Q with the newly queried pairs: $Q \leftarrow Q \cup Q_j$ and train the model with the newly queried pairs Q_j or with all previously queried pairs Q.

Note that we have to restrict our queries to the set of pairs for which we actually have access to ground truth scores.

References

- MPNN: link
- Review on GNNs: link
- Decagon: link
- Bayesian Optimization: link
- Graph Laplacian: link