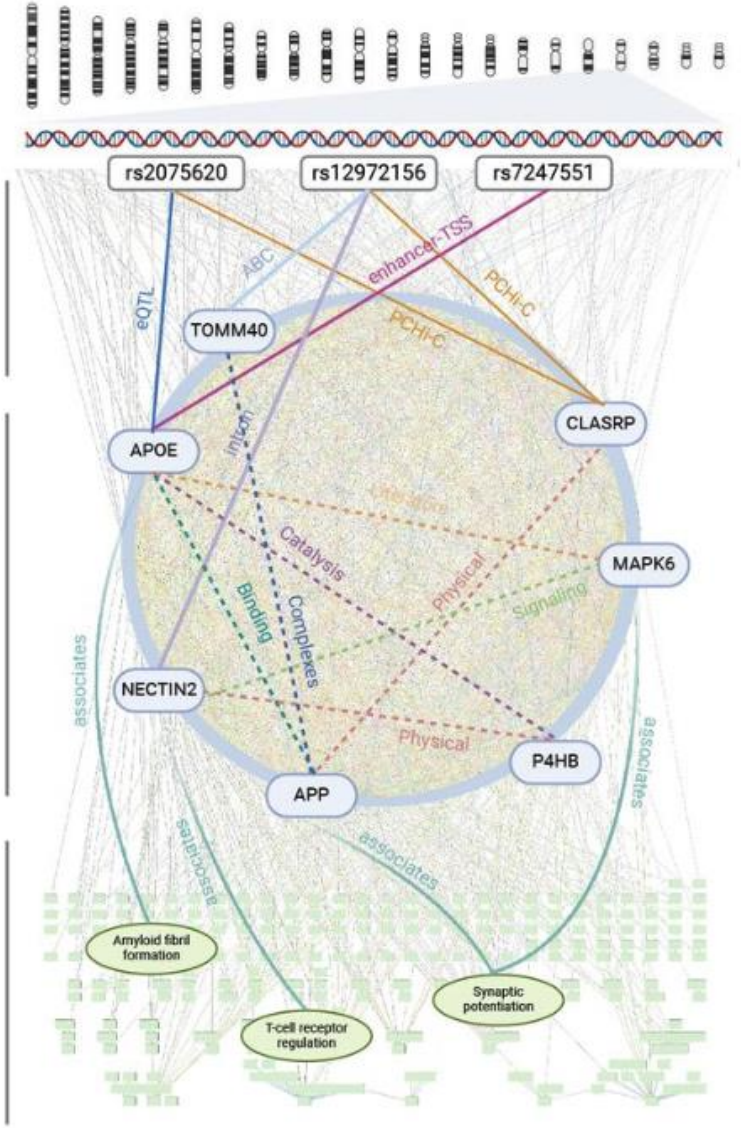
Small-cohort GWAS discovery with AI over massive functional genomics knowledge graph

Kexin Huang, Tony Zeng, Soner Koc, Alexandra Pettet, Jingtian Zhou, Mika Jain, Dongbo Sun, Camilo Ruiz, Hongyu Ren, Laurence Howe, Tom G. Richardson, Adrian Cortes, Katie Aiello, Kim Branson, Andreas Pfenning, Jesse M. Engreitz, Martin Jinye Zhang, Jure Leskovec

doi: https://doi.org/10.1101/2024.12.03.24318375



JHU Deep Learning Reading Group

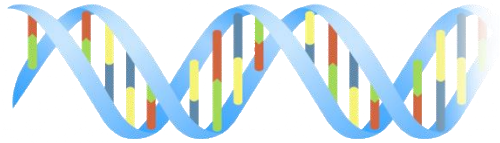
Feb 18, 2025

Outline

I. Introduction to GWAS and motivation of the problem
   I. What do we mean by functional graph?
   II. Why does this matter?
   III. How does it help? (simple example)
II. The method- a graph neural network + GWAS
III. Results
IV. Discussion

GWAS map disease-associated genetic variants

- Genome-wide association studies detect associations between phenotypes and genetic variants
- These help target regions of the genome related to particular diseases
- Detecting these associations limited by **N, effect size strength, rarity**

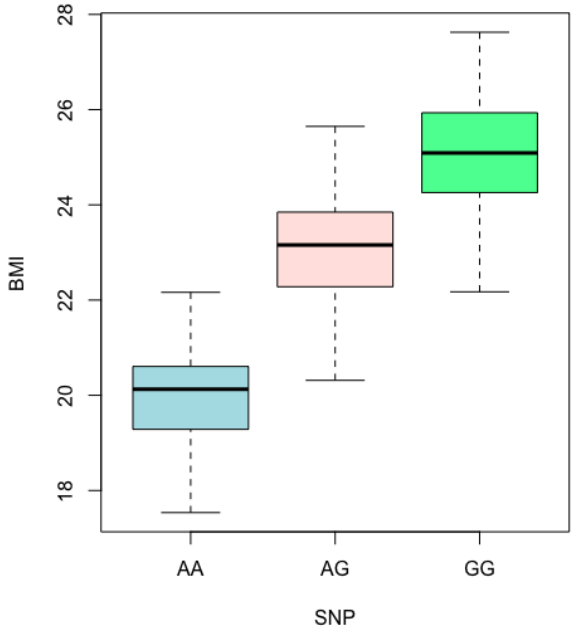


BMI
20
18.9
24
25
...

~

rs1234	rs1235	rs1236
AA	GT	TC
AG	GT	CC
GG	TG	TT
GA	TT	CT
...	...	...

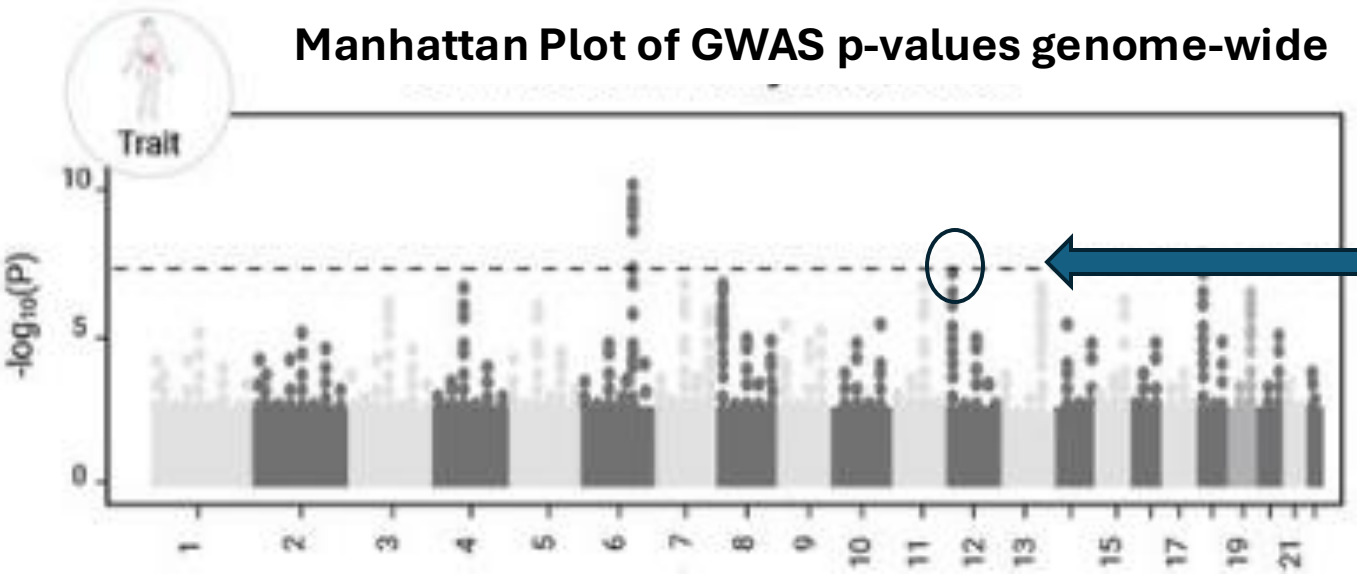
rs1234 correlates with BMI



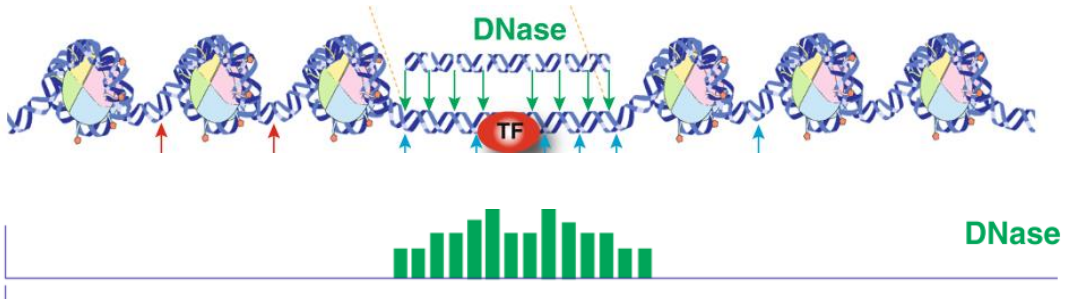
High testing burden + disease rarity limit GWAS

- Typically run genome-wide across *millions of variants*, so the standard for a "significant association" is $p \leq 5 \times 10^{-8}$
- Difficult to assemble a sufficiently large cohort to detect associations at this level for diseases which are very rare
- Existing work tries to leverage additional **functional genomic data** to improve variant detection

Manhattan Plot of GWAS p-values genome-wide

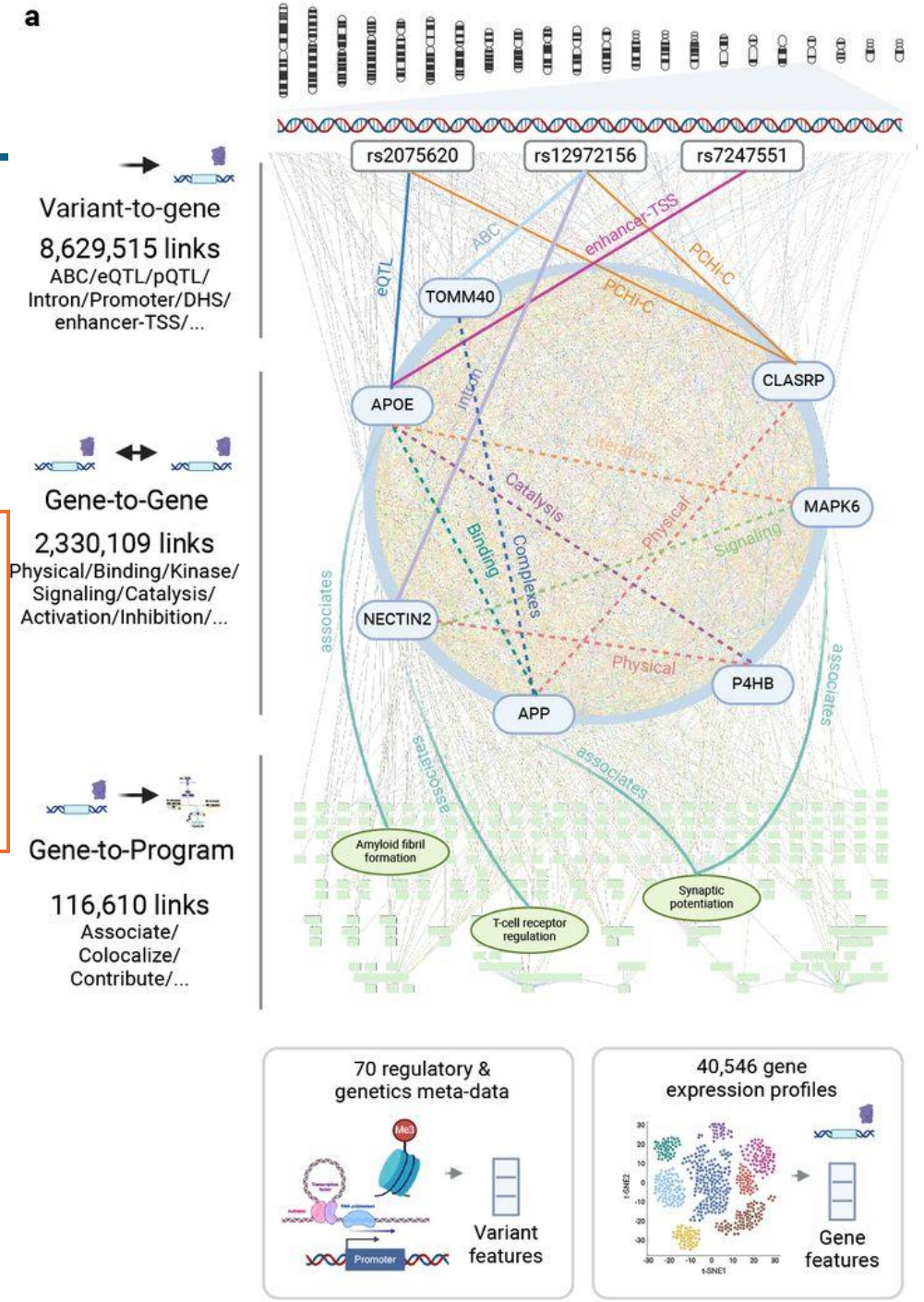


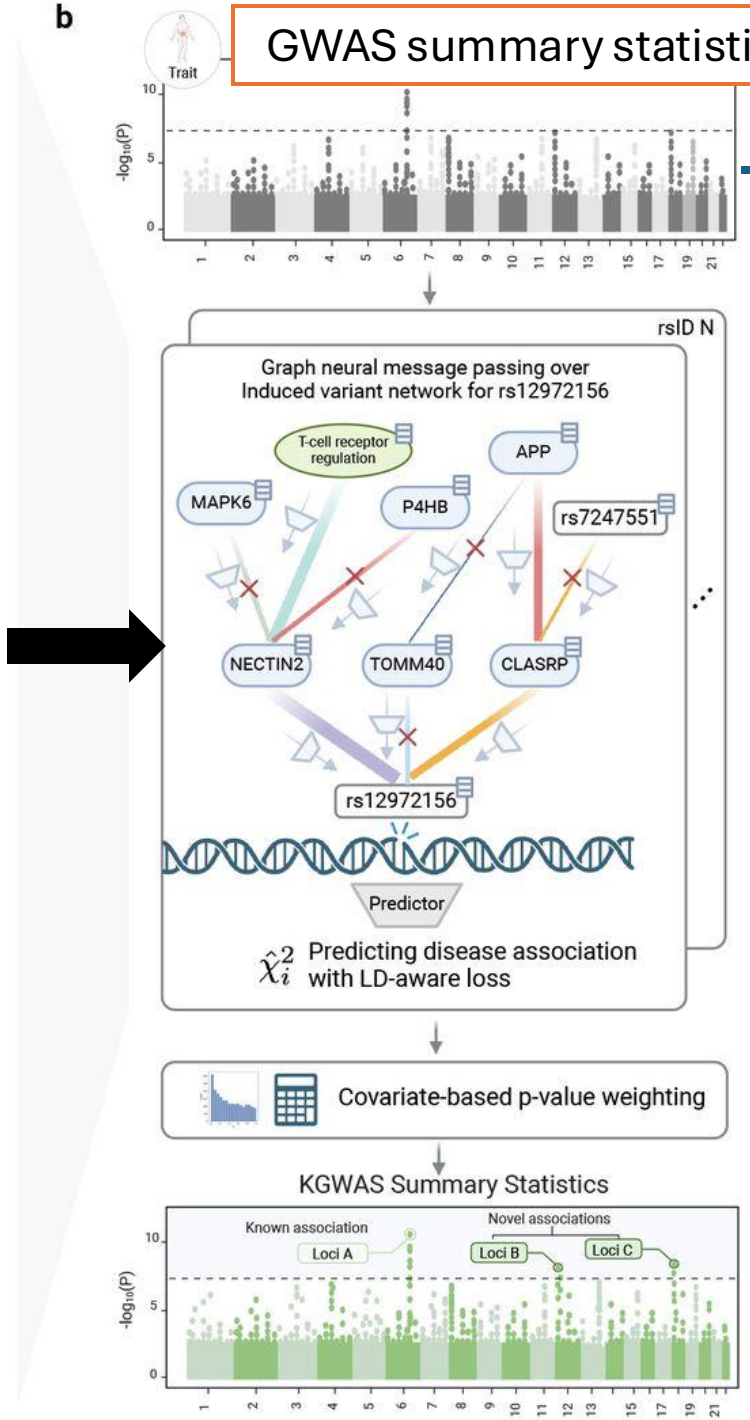
Accessible chromatin region + promoter



... but many annotations exist

Big ole' knowledge graph of genomic relationships (11M links)





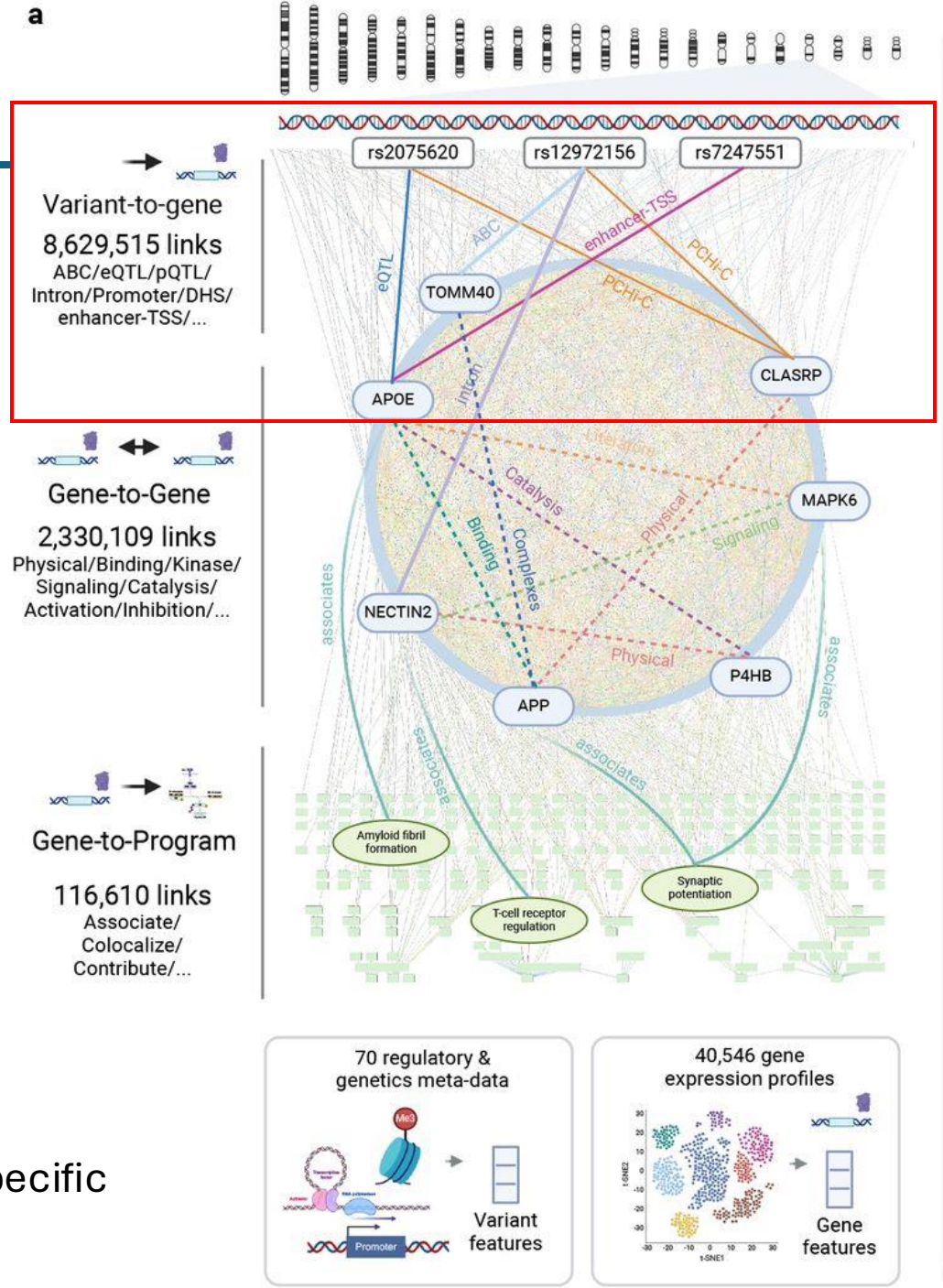
Graph neural network to estimate GWAS effects (χ^2)

Combine p-values from GWAS and GNN

updated summary statistics

Knowledge graph of genomic relationships (+ GNN!)

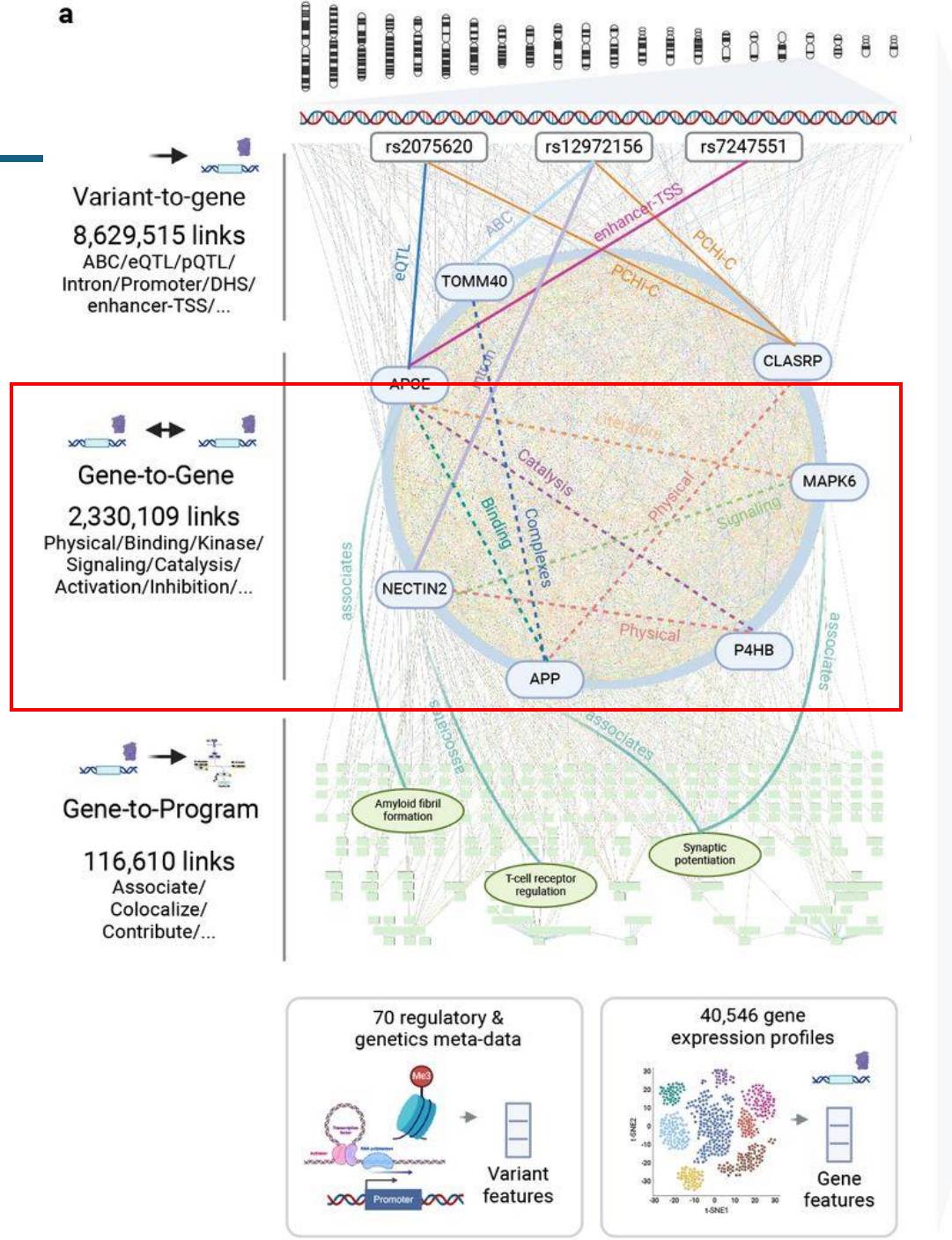
- Three* main categories:
  1) Variant-to-gene links (V2G)
     - 8.6 M links across 784K variants
     - 14 different data sources for these, including
       - Transcription start site variant
       - Promoter variant
       - DNAse-hypersensitive sites
       - eQTL
       - Activity-by-contact
       - pQTL
       - ensemble Variant-effect-prediction
       - Hi-C
       - etc…



*3 categories of relationships, but embeddings also include variant-specific annotations (e.g. coding variant MAF, baseline LD-score annotations)

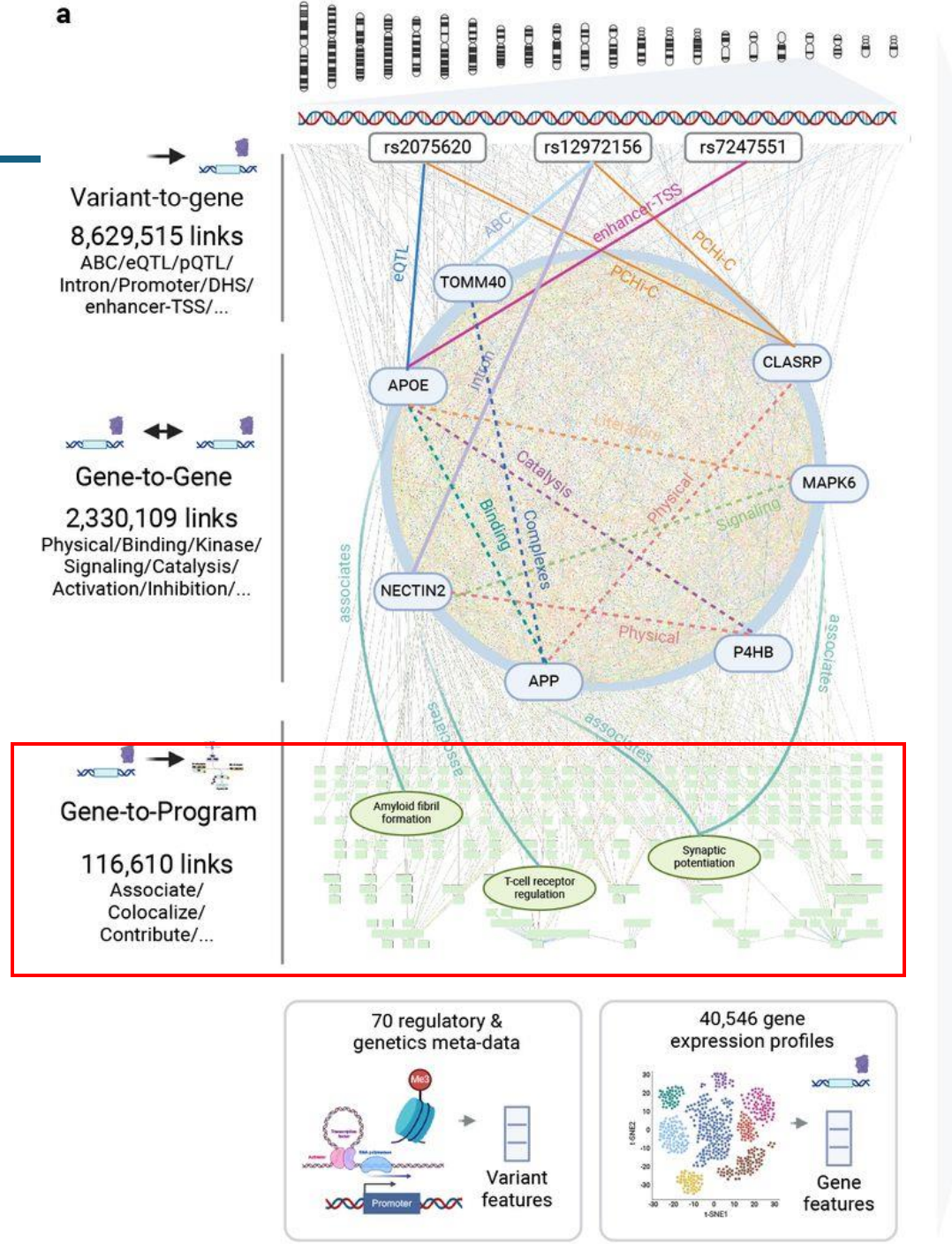
Knowledge graph of genetic relationships (+ GNN!)

- Three main categories:
  2) Gene-to-gene links
    - 2.3 M links across 19K protein-coding genes
    - 40 different gene-gene relationship types from STRING database, bioGRId, Database of Interacting proteins, e.g.
      - Gene-gene interaction in literature
      - Gene-gene physical association
      - Gene-gene binding
      - Gene-gene inhibition
      - Gene expression relationship



Knowledge graph of genetic relationships (+ GNN!)

- Three main categories:
  3) Gene programs
    - 117K links across 44K gene programs
    - Gene programs are predefined groups of genes with shared functions
      - Uses Gene Ontology (GO) annotations:
      - GO Cellular Component, e.g.
        - cytoskeleton
      - GO Molecular function, e.g.
        - transporter activity
        - insulin receptor activity
      - GO Biological process, e.g.
        - DNA repair
        - cytosine biosynthetic process



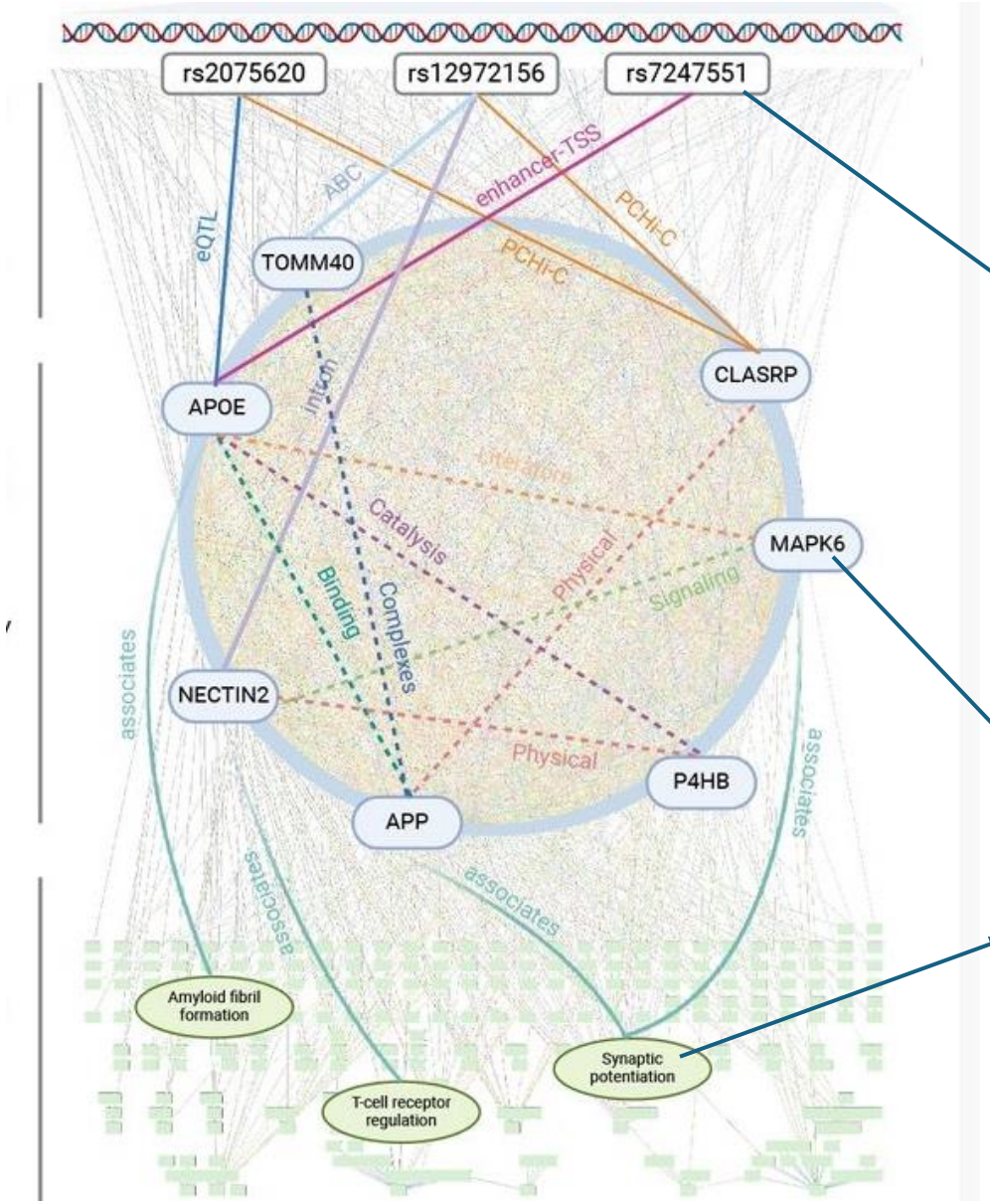
Encoding of network edges

- $e_{i,j,r} = (v_i, v_j, r)$, where
  - v_i: "source node"
  - v_j: "target node"

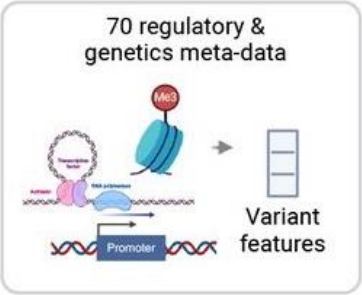
  $v_i, v_j \in \mathcal{V}$, the vertex set (containing all nodes)

  - r: relationship type (e.g. SNP i is an eQTL of gene j)
    - $r \in \mathcal{T}_{\mathrm{r}}$, the set of all relationship types

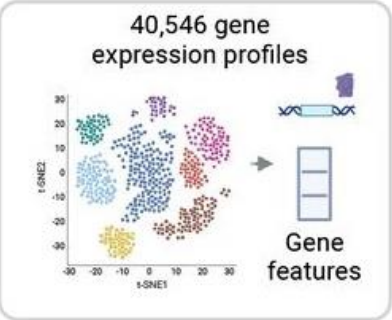
Each entity (variant, gene, gene program) is a node in the graph



- Each node on the graph is **initialized** as an embedding, which is updated during training.
- These are initialized as follows:
- **Variant nodes:** initialized as a vector of 70 SNP specific annotations (baseline LD annotations. from LD score regression)
  - e.g. coding, conservation score, DHS, Enhancer, methylation data, intron, TFBS, CpG content, recombination rate, MAF, etc.



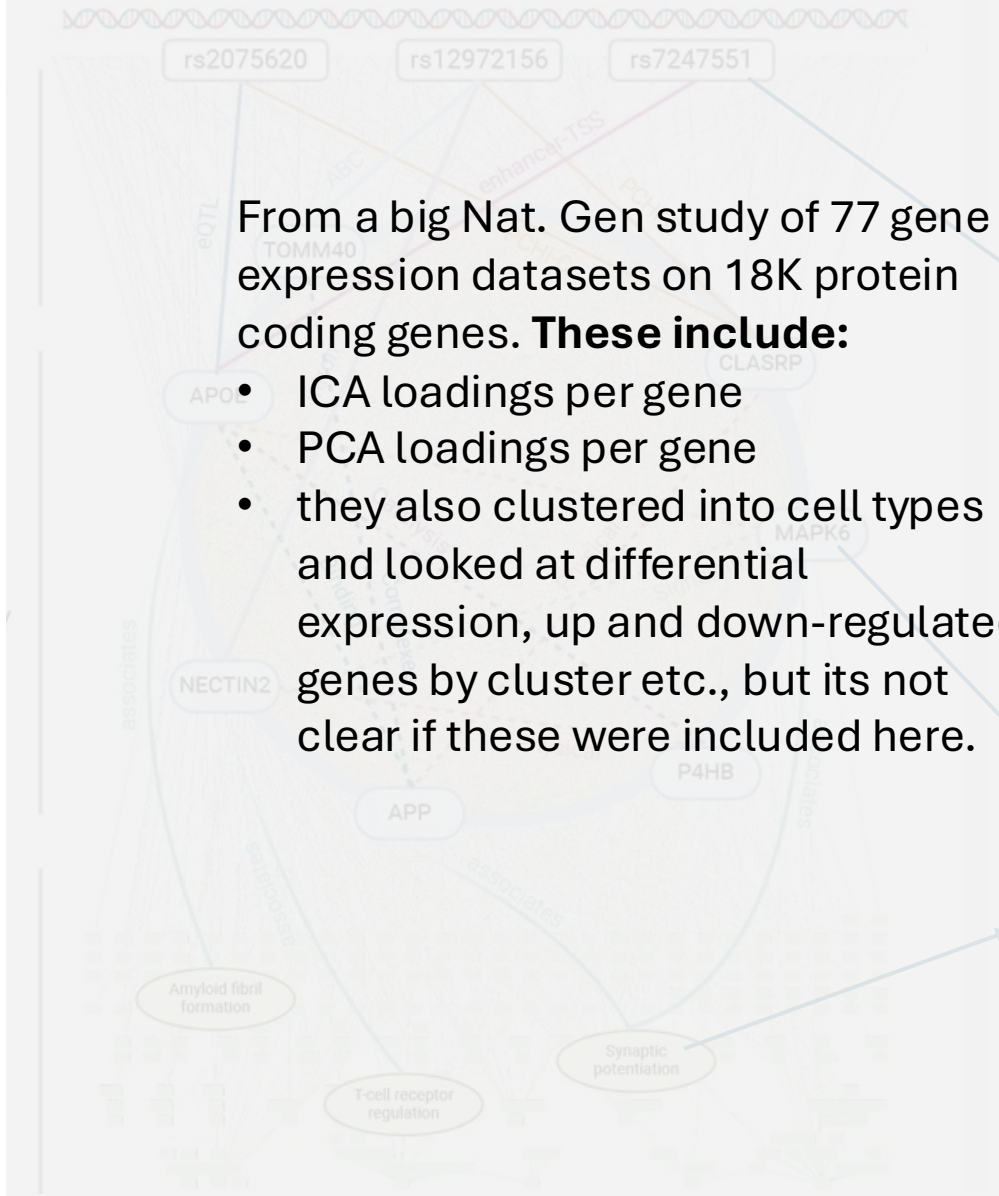
- **Gene nodes:** initialized with 40,546 gene annotations from published scRNA-seq data (“PoPS features”)



- **Gene program nodes:** randomly initialized

Node i embedding initialization notation: $\boldsymbol{h}_i^{(0)}$

Each entity (variant, gene, gene program) is a node in the graph



From a big Nat. Gen study of 77 gene expression datasets on 18K protein coding genes. **These include:**

- ICA loadings per gene
- PCA loadings per gene
- they also clustered into cell types and looked at differential expression, up and down-regulated genes by cluster etc., but its not clear if these were included here.

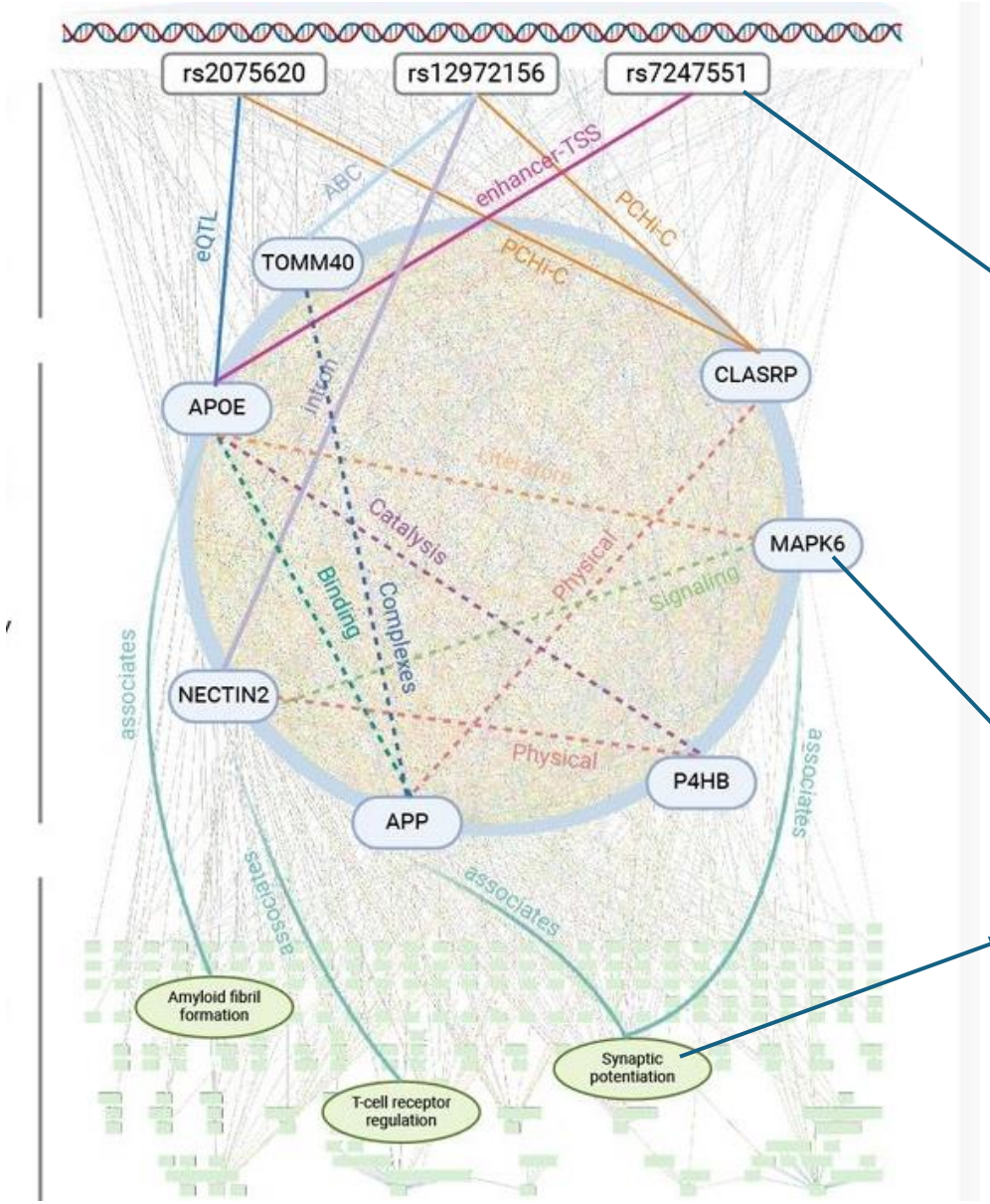
- Each node on the graph is **initialized** as an embedding, which is updated during training.
- These are initialized as follows:
- **Variant nodes:** initialized as a vector of 70 SNP specific annotations (baseline LD annotations. from LD score regression)
  - e.g. coding, conservation score, DHS, Enhancer, methylation data, intron, TFBS, CpG content, recombination rate, MAF, etc.

- **Gene nodes:** initialized with 40,546 gene annotations from published scRNA-seq data (“**PoPS features**”)

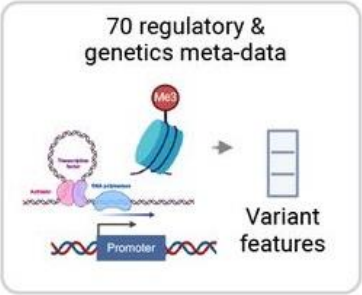
- **Gene program nodes:** randomly initialized

Node i embedding initialization notation: $h_i^{(0)}$

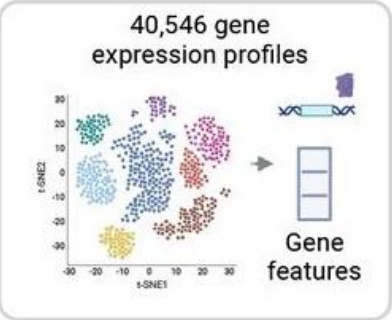
Each entity (variant, gene, gene program) is a node in the graph



- Each node on the graph is **initialized** as an embedding, which is updated during training.
- These are initialized as follows:
- **Variant nodes:** initialized as a vector of 70 SNP specific annotations (baseline LD annotations. from LD score regression)
  - e.g. coding, conservation score, DHS, Enhancer, methylation data, intron, TFBS, CpG content, recombination rate, MAF, etc.



- **Gene nodes:** initialized with 40,546 gene annotations from published scRNA-seq data (“PoPS features”)

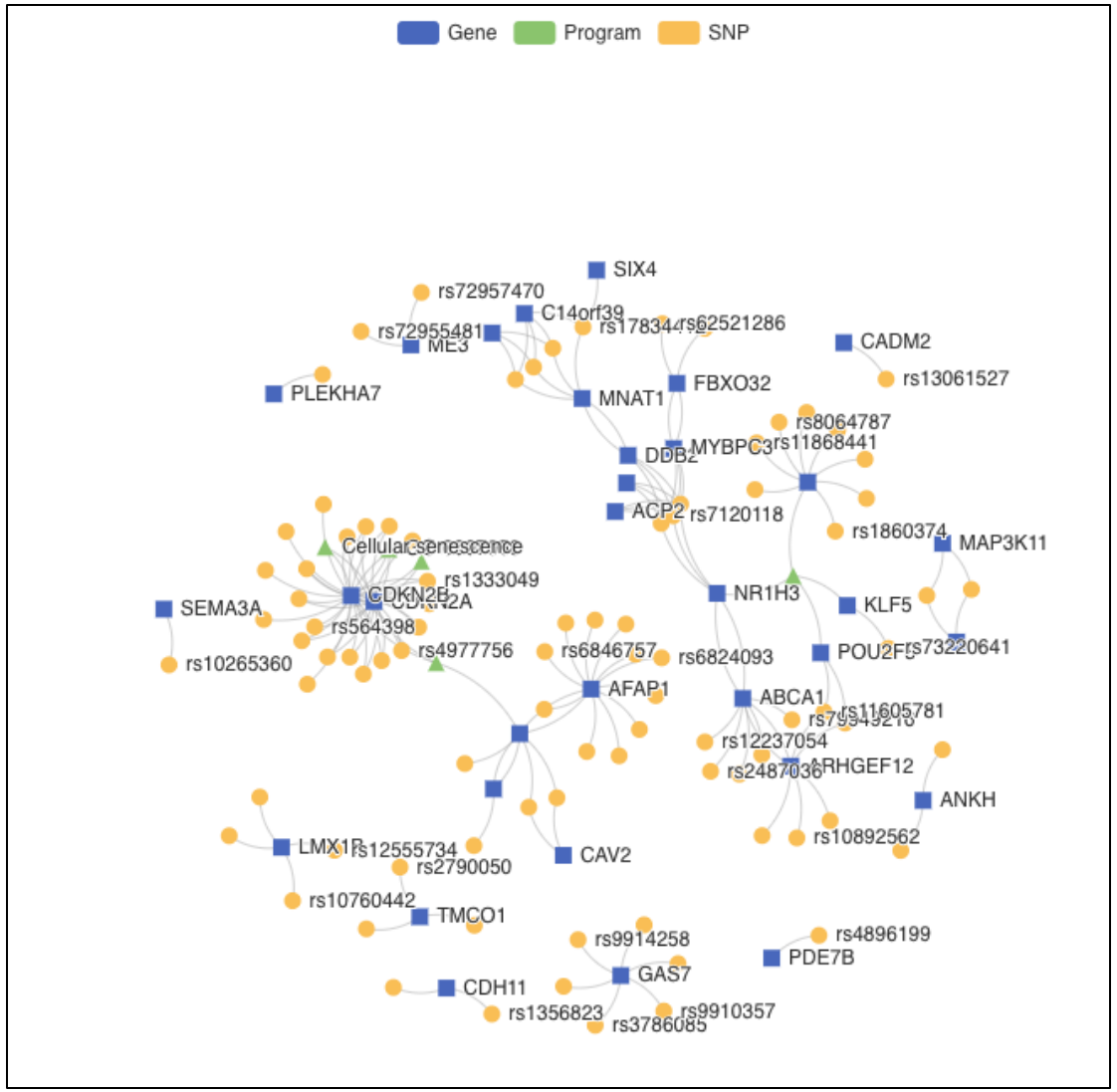


- **Gene program nodes:** randomly initialized

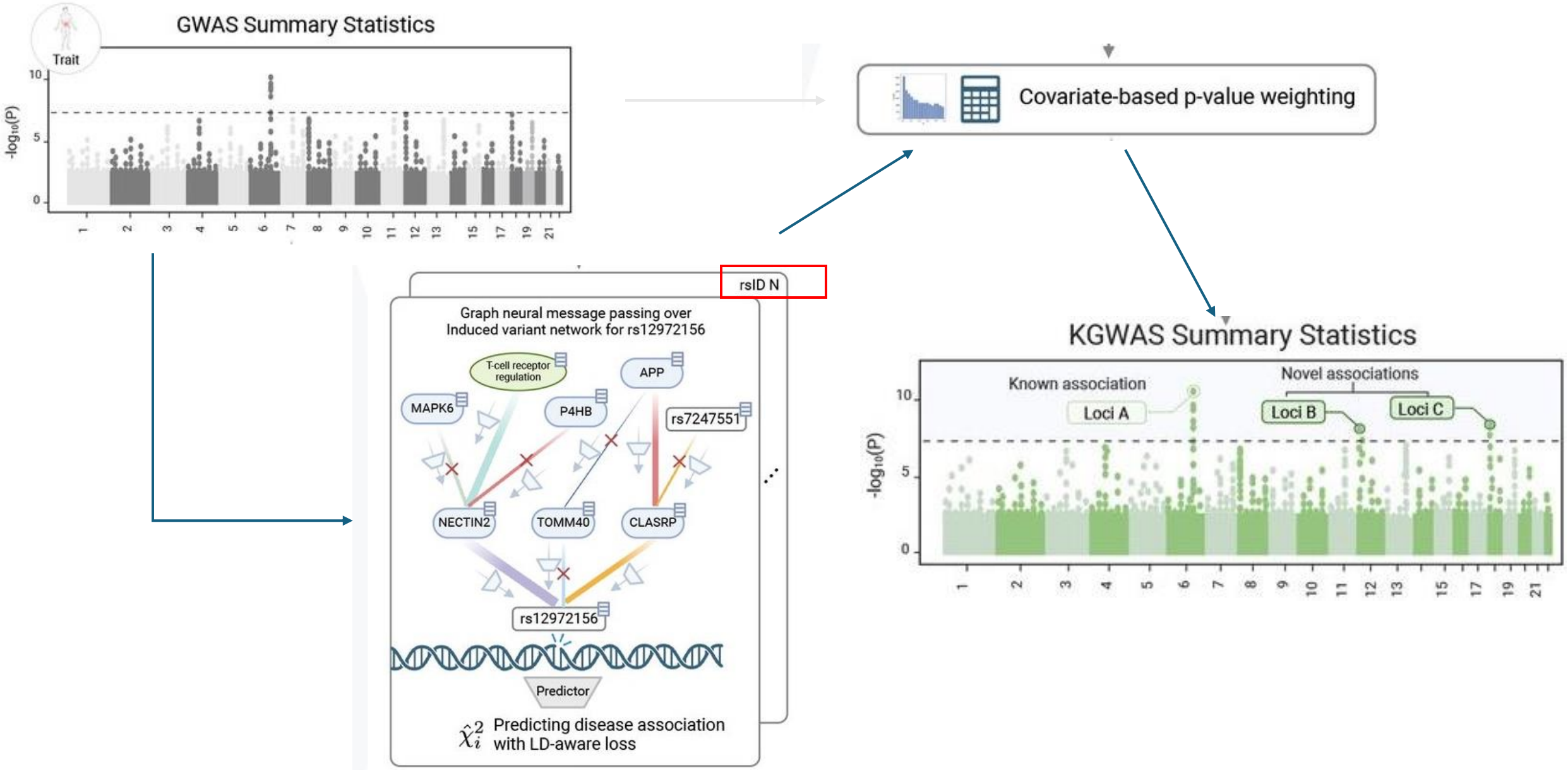
Node i embedding initialization notation: $\boldsymbol{h}_i^{(0)}$

Check out their network here

- **See this in action here:**
  - https://kgwas.stanford.edu/#tab-8708-1



Integrating this network results in new discoveries



Procedure for network training

1. **Initialization**
   1. Specify input node embedding for each node and edge relationships
2. **Propagate** relationship-specific neural messages
3. **Aggregate** local network neighborhood information
4. **Update** network target node embedding (to minimize prediction loss of χ^2 statistic)

*repeat 2-4 over **L** layers of propagation*

Graph neural network LD-aware loss function

- Minimizing the following loss function:

$$\mathcal{L} = \sum_{i \in \mathcal{S}} \frac{1}{l_i} * \frac{1}{(N h_g^2 l_i / M + 1)^2} |\hat{\chi}_i^2 - \chi_i^2|^2,$$

"LD-score"- estimates the amount of genetic variation tagged by variant i (usually due to LD)

This scaling is based on a ton of existing work, ***but very important b/c helps account for the correlation structure which exists among genomic variants due to linkage disequilibrium***

Sample size of GWAS summary statistics

Heritability constant of summary statistics

Total # of variants in reference LD panel

Prediction from NN for variant *i*

From provided GWAS data

Graph neural network LD-aware loss function

- Minimizing the following loss function:

$$\mathcal{L} = \sum_{i \in \mathcal{S}} \frac{1}{l_i} * \frac{1}{(N h_g^2 l_i / M + 1)^2} |\hat{\chi}_i^2 - \chi_i^2|^2,$$

Because $\widehat{\chi^2}$ is a prediction based on network priors, not an association discovery statistic, there is no sense of a p-value or false discovery control

Prediction from NN for variant *i*

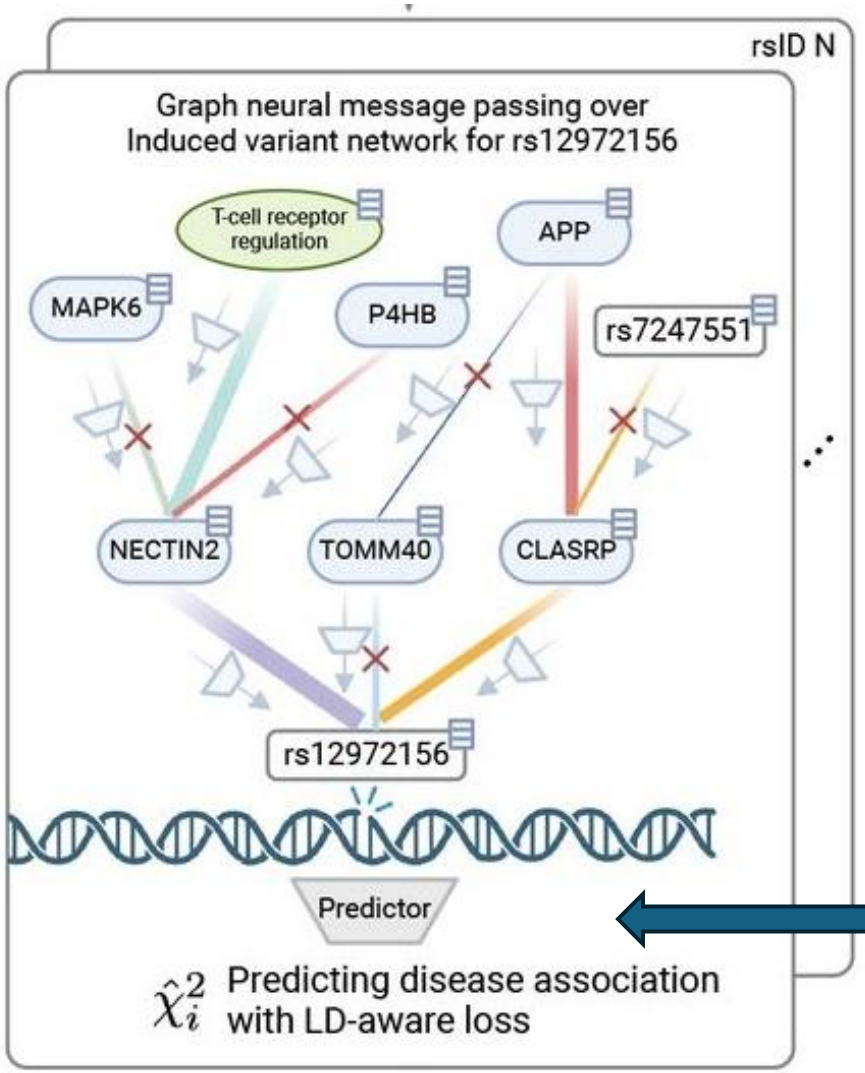
This $\widehat{\chi^2}$ is used to derive a p-value using a weighting framework that controls for false discovery

TL;DR:

1. Stratify SNPs in bins based on $\widehat{\chi^2}$
2. Estimate proportion of null (π_0) and true (π_1) associations in bin using a cubic spline on the p-value histogram
3. Weight p-value by a bin-normalized weight (ratio of π_0 and π_1 per bin)
4. Calibrate estimated p-values by original GWAS p-values by applying a rank-preserving scaling factor

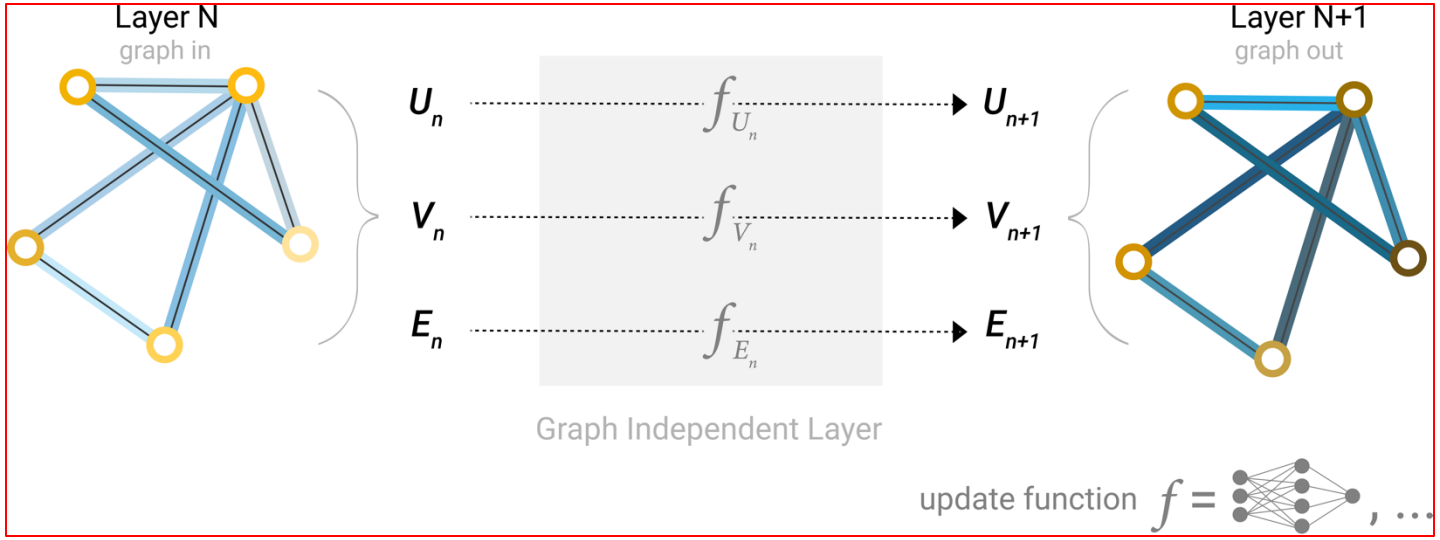
(see the paper for more details) to ensure that # of discoveries is the same in some p-value range

Prediction of χ_i^2 from variant embedding $h_i^{(L)}$



L: the total number of "**layers of propagation**" across the network

- Instead of having "layers" of your DNN nodes through which information propagates, in GNNs we refer to "layers of propagation"
- e.g. the N+1 layer is the graph updated after message passing between nodes has occurred on the N later



"we first applied a prediction head on the variant embedding $h_i^{(L)}$"

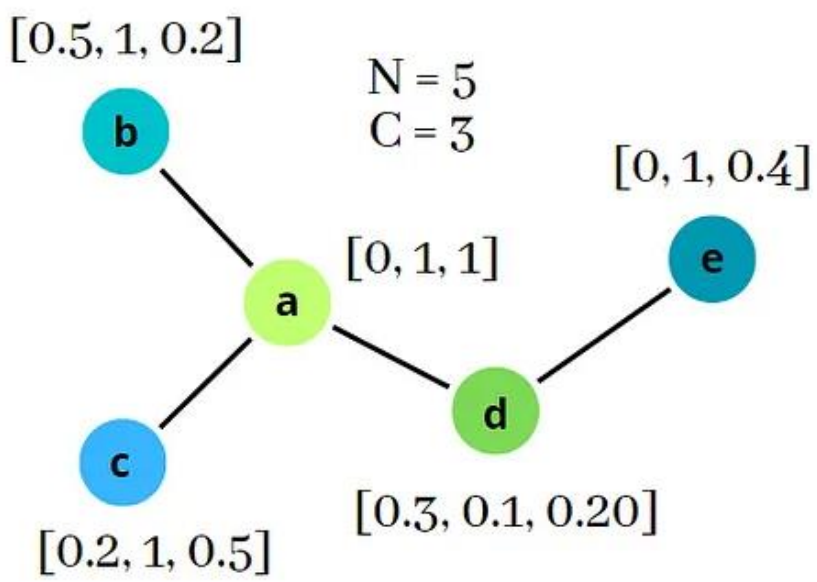
Image from "A gentle introduction to graph neural networks", distill.pub

GNNs built on aggregate and update/combine functions

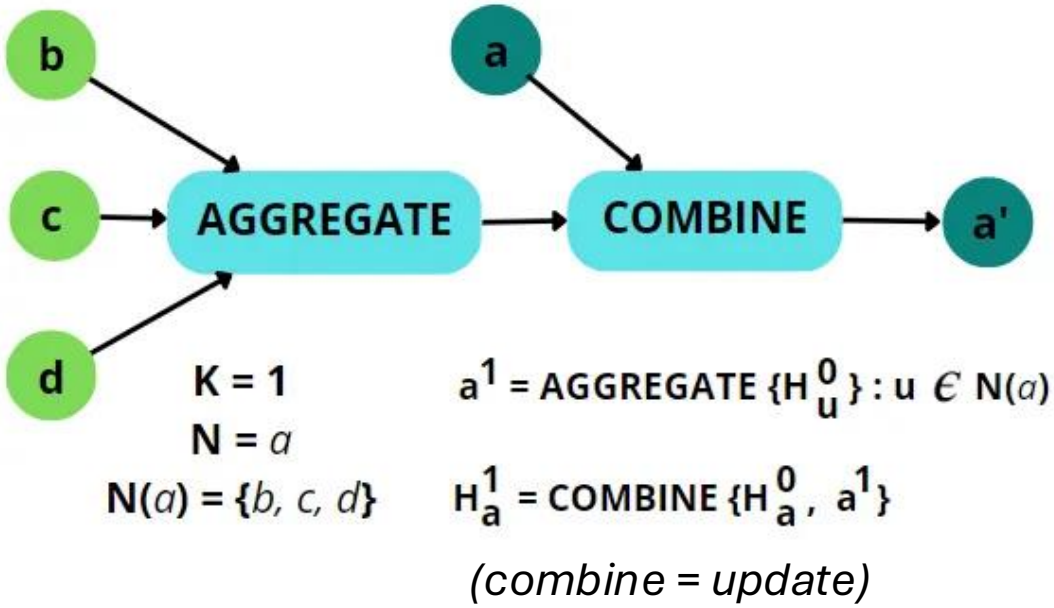
- **AGGREGATE-** combine information from surrounding neighbor nodes (order invariant)
- **UPDATE-** Use aggregated information from neighbors to update target node

$$X \in R^{N \times C}$$

0	1	1
0.5	1	0.2
0.2	1	0.5
0.3	0.1	0.20
0	1	0.4



embedding for node a



From "Introduction to Graph Neural Networks: An illustrated guide (Medium.com)

GNNs built on aggregate and update/combine functions

- **AGGREGATE-** combine information from surrounding neighbor nodes (order invariant)
- **UPDATE-** Use aggregated information from neighbors to update target node

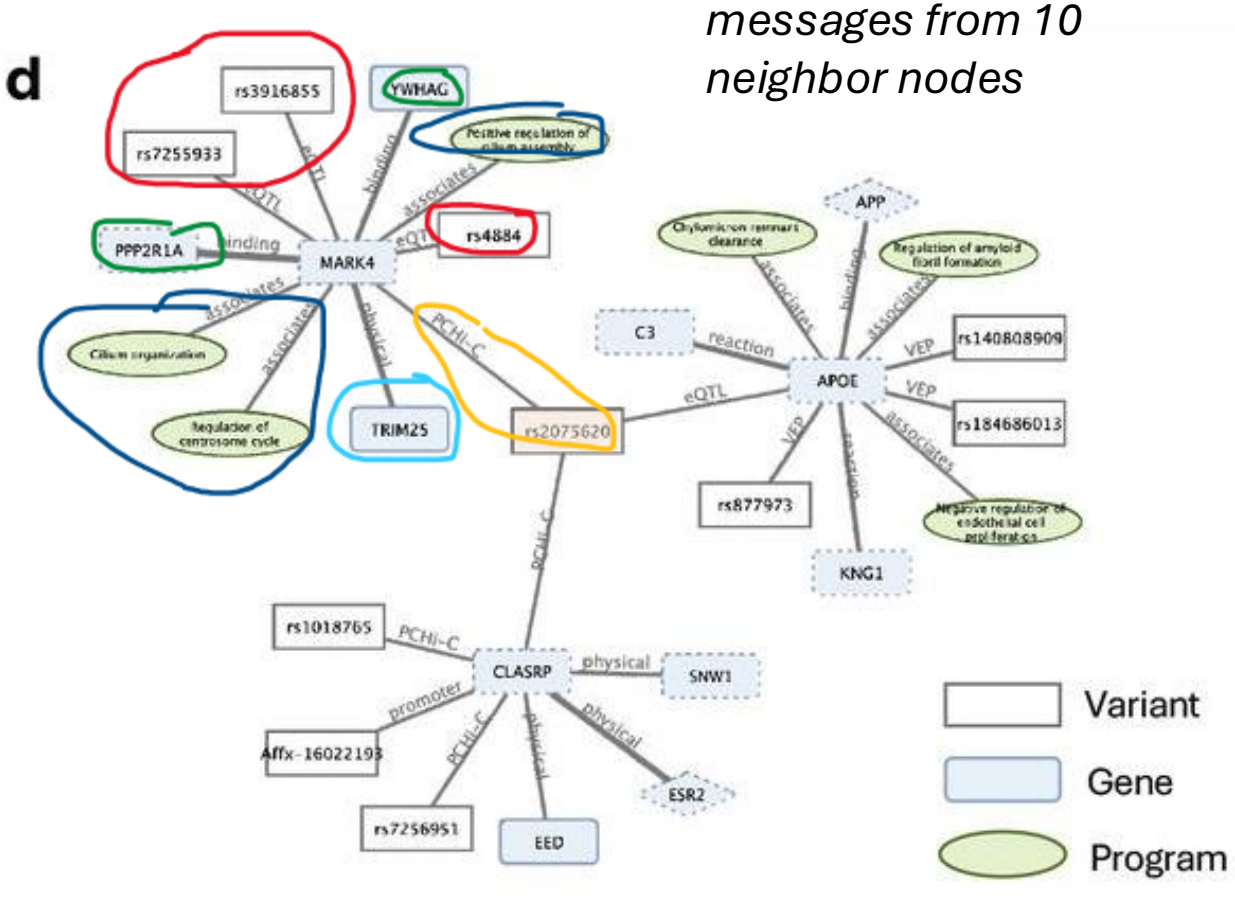
AGGREGATE

weighted average of neighbor messages

$$\tilde{\boldsymbol{m}}_{r,i}^{(l)} = \frac{1}{|\mathcal{N}_r(i)|} \sum_{j \in \mathcal{N}_r(i)} \alpha_{i,j,r}^{(l)} \boldsymbol{m}_{r,j}^{(l)}$$

- r: relationship type
- i: target node
- $m_{r,j}^{(l)}$: "message" from neighbor node j of relationship type r at layer l
- $\mathcal{N}_r(i)$: the set of nodes neighboring i of relationship type r
- $\alpha_{i,j,r}^{(l)}$: attention term (coming up soon!)

e.g. MARK node has 5 different aggregated messages from 10 neighbor nodes



From "Introduction to Graph Neural Networks: An illustrated guide (Medium.com)

GNNs built on aggregate and update/combine functions

- **AGGREGATE-** combine information from surrounding neighbor nodes (order invariant)
- **UPDATE-** Use aggregated information from neighbors to update target node

AGGREGATE

weighted average of neighbor messages

$$\widetilde{\boldsymbol{m}}_{r,i}^{(l)} = \frac{1}{|\mathcal{N}_r(i)|} \sum_{j \in \mathcal{N}_r(i)} \alpha_{i,j,r}^{(l)} \boldsymbol{m}_{r,j}^{(l)}$$

- r: relationship type
- i: target node
- $m_{r,j}^{(l)}$: "message" from neighbor node j of relationship type r at layer l
- $\mathcal{N}_r(i)$: the set of nodes neighboring i of relationship type r
- $\alpha_{i,j,r}^{(l)}$: attention term (coming up soon!)

UPDATE

add to current embedding

$$\boldsymbol{h}_i^{(l)} = \boldsymbol{h}_i^{(l-1)} + \sum_{r \in \mathcal{T}_R} \widetilde{\boldsymbol{m}}_{r,i}^{(l)}$$

$h_i^{(l-1)}$: Node i embedding at previous layer
$\mathcal{T}_R$: set of all relationship type r
$\widetilde{m}_{r,i}^{(l)}$: weighted average message of all neighbors of i having relationship type r

This process of aggregating and updating for each node is repeated across L layers of propagation, until we reach $h_i^{(L)}$

From "Introduction to Graph Neural Networks: An illustrated guide (Medium.com)

What are neighbor "messages" ($\boldsymbol{m}_{r,j}^{(l)}$)?

- KGWAS learns a relationship-specific weight matrix to convert the embedding into its corresponding "message":

$$\boldsymbol{m}_{r,j}^{(l)} = \boldsymbol{W}_{r,M}^{(l)} \boldsymbol{h}_j^{(l-1)}$$

- e.g. the outbound message from node j at layer l, with respect to relationship r (to node i) based on the previous layer's embedding
- M indicates $\boldsymbol{W}$ is the weight matrix for message passing for relationship type **r**

ATTENTION

These messages are used to assign edge-specific attention weight ($e_{i,j,r}^{(l)}$), quantifying the importance of a given edge for message-passing

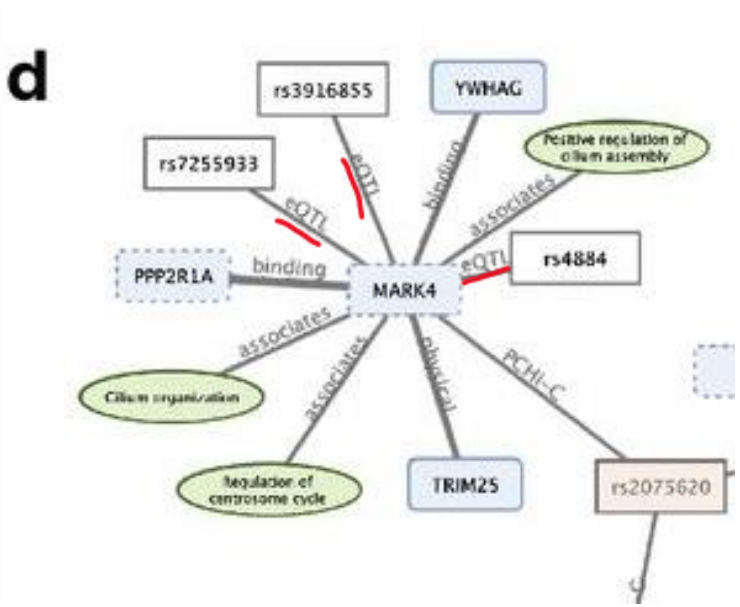
Edge attention aids interpretability

- Edge-specific attention score
  - $e_{i,j,r}^{(l)} = LeakyRelu(\boldsymbol{W}_{r,A}^{(l)} (\boldsymbol{m}_{r,i}^{(l)} || \boldsymbol{m}_{r,j}^{(l)}))$

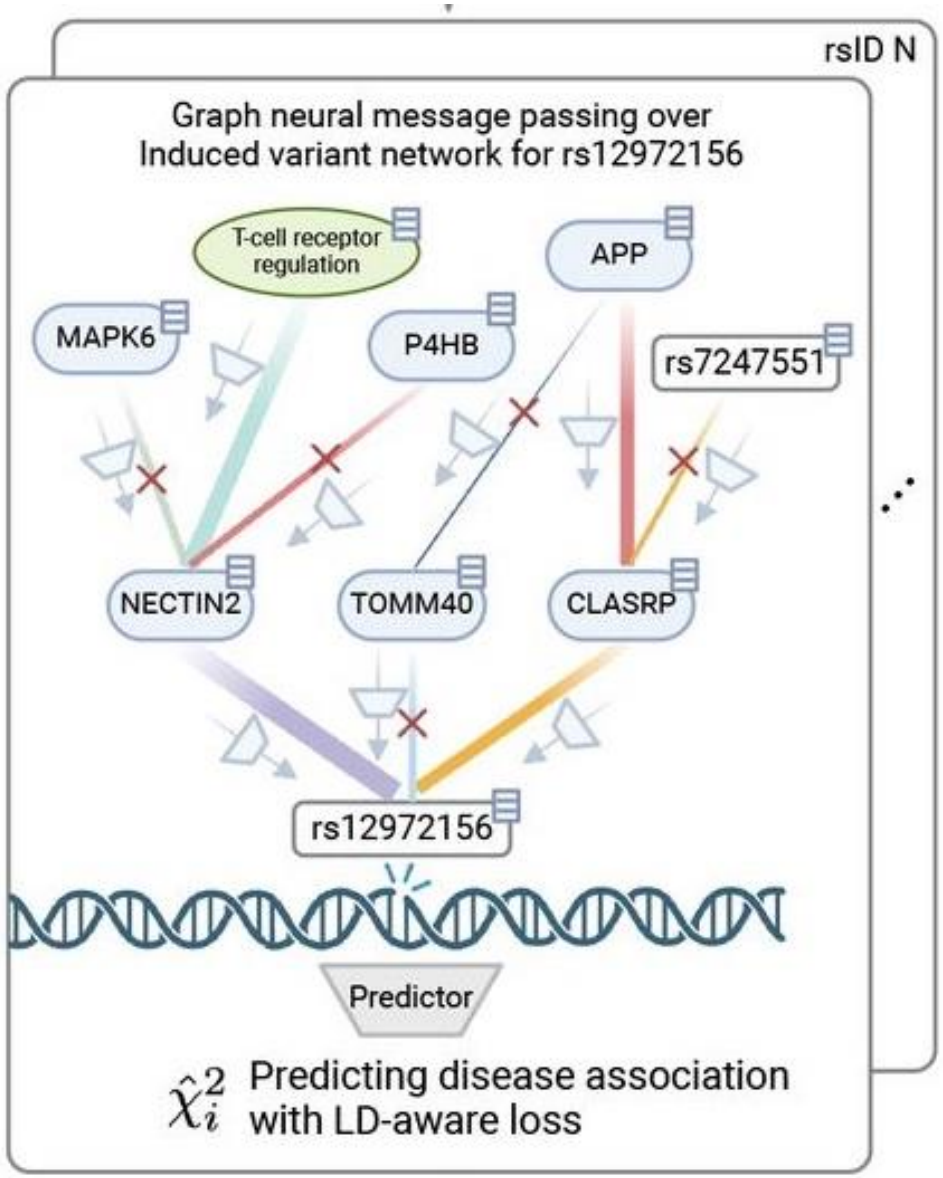
- A indicates $\boldsymbol{W}$ is the weight matrix for ATTENTION for relationship type **r**
- || indicates concatenation of message vectors

- **These scores are normalized to give a weight for a given relationship type across all edges with that relationship:**

$$\alpha_{i,j,r}^{(l)} = \frac{\exp\left(e_{i,j,r}^{(l)}\right)}{\sum_{k \in \mathcal{N}_r} \exp(e_{i,k,r}^{(l)})}$$



Model structure- in summary



$$\boldsymbol{h}_i^{(0)} \quad \boldsymbol{h}_j^{(0)} \longrightarrow \boldsymbol{m}_{r,j}^{(l)} = \boldsymbol{W}_{r,M}^{(l)} \boldsymbol{h}_j^{(l-1)}$$

$$e_{i,j,r}^{(l)} = LeakyRelu(\boldsymbol{W}_{r,A}^{(l)} (\boldsymbol{m}_{r,i}^{(l)} || \boldsymbol{m}_{r,j}^{(l)}))$$

$$\alpha_{i,j,r}^{(l)} = \frac{\exp\left(e_{i,j,r}^{(l)}\right)}{\sum_{k \in \mathcal{N}_r} \exp(e_{i,j,r}^{(l)})}$$

$$\widetilde{\boldsymbol{m}}_{r,i}^{(l)} = \frac{1}{|\mathcal{N}_r(i)|} \sum_{j \in \mathcal{N}_r(i)} \alpha_{i,j,r}^{(l)} \boldsymbol{m}_{r,j}^{(l)}$$

$$\boldsymbol{h}_i^{(l)} = \boldsymbol{h}_i^{(l-1)} + \sum_{r \in \mathcal{T}_R} \widetilde{\boldsymbol{m}}_{r,i}^{(l)}$$

$$\mathcal{L} = \sum_{i \in \mathcal{S}} \frac{1}{l_i} * \frac{1}{(N h_g^2 l_i / M + 1)^2} |\hat{\chi}_i^2 - \chi_i^2|^2,$$

Empirical tests of KGWAS performance

1. **Does it detect the signal we want it to?**
2. **Does it actually boost power when N is low in GWAS?**
    1. Evaluations in real GWAS on subsampled cohorts vs full cohort
3. **Does the knowledge graph actually make a difference?**
4. **Is the benefit of KGWAS knowledge graph greater or less than embeddings from foundation gene models (ESM, Enformer)?**

Empirical tests of KGWAS performance

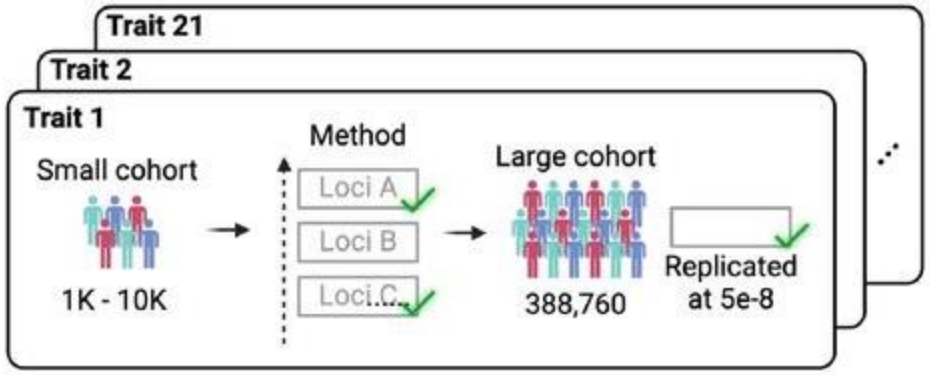
1. **Does it detect the signal we want it to?**
    1. Simulations that show that KGWAS is:
        1. Well-calibrated false positive rate in null simulations
        2. detects more true associations than standard GWAS or similar methods in causal simulations
        3. KGWAS is well-calibrated with better power across trait heritability and causal variant settings
2. **Does it actually boost power when N is low in GWAS?**
    1. Evaluations in real GWAS on subsampled cohorts vs full cohort
3. **Does the knowledge graph actually make a difference?**
4. **Is the benefit of KGWAS knowledge graph greater or less than embeddings from foundation gene models (ESM, Enformer)?**

Empirical tests of KGWAS performance

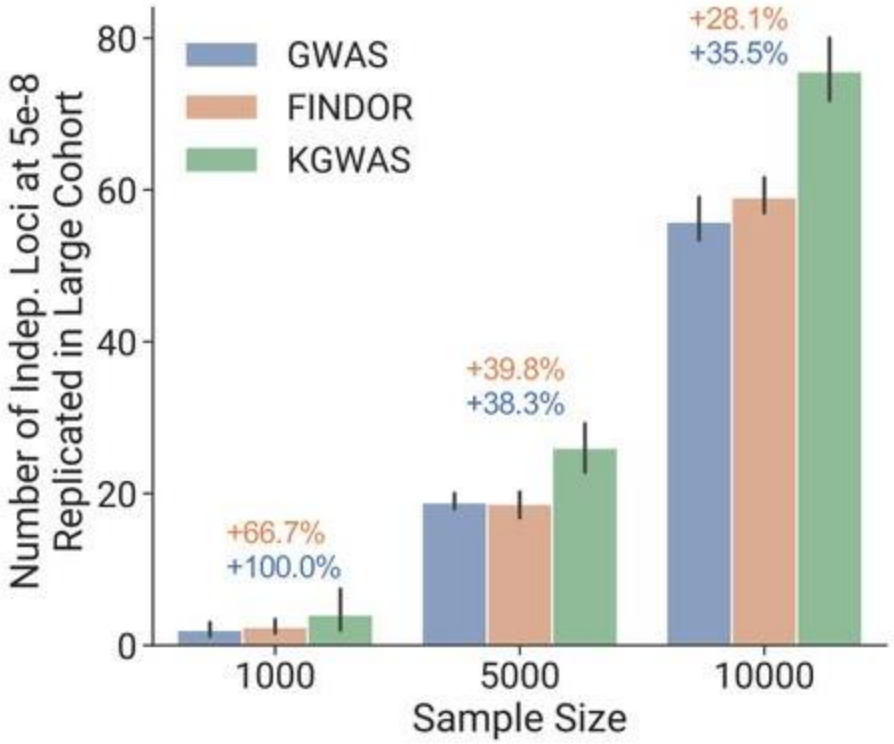
1. **Does it detect the signal we want it to?**
2. **Does it actually boost power when N is low in GWAS?**
   1. Evaluations in real GWAS on subsampled cohorts vs full cohort
3. **Does the knowledge graph actually make a difference?**
4. **Is the benefit of KGWAS knowledge graph greater or less than embeddings from foundation gene models (ESM, Enformer)?**

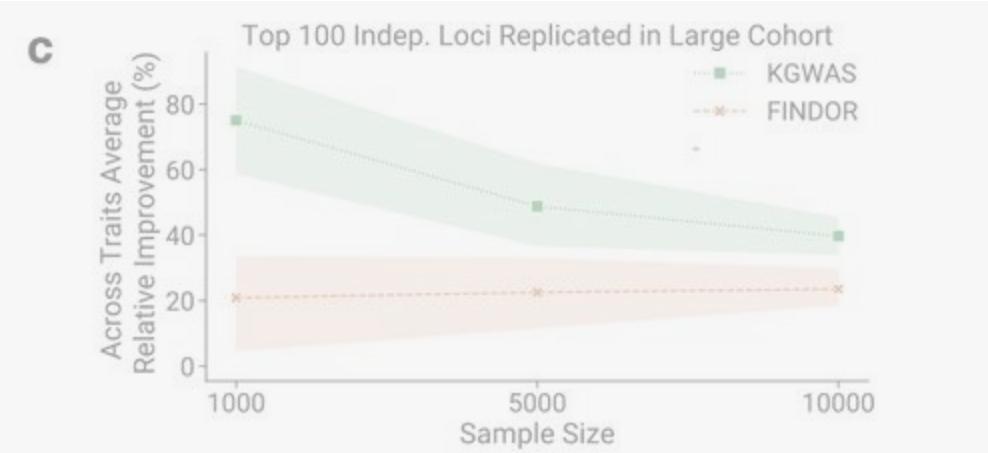
How does KGWAS boost power to detect associations?

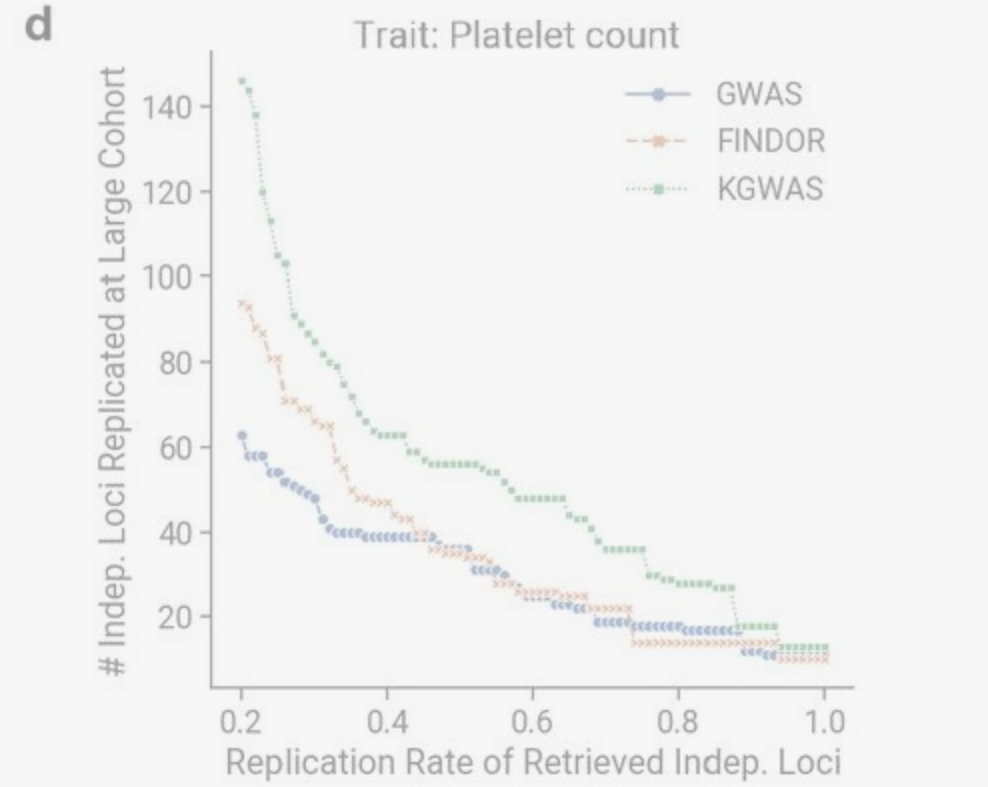
Test: Run KGWAS on a subsampled GWAS cohort, and compare detected associations to the full cohort:



1) KGWAS produced many more replicated discoveries than vanilla GWAS (at P <5e-08)



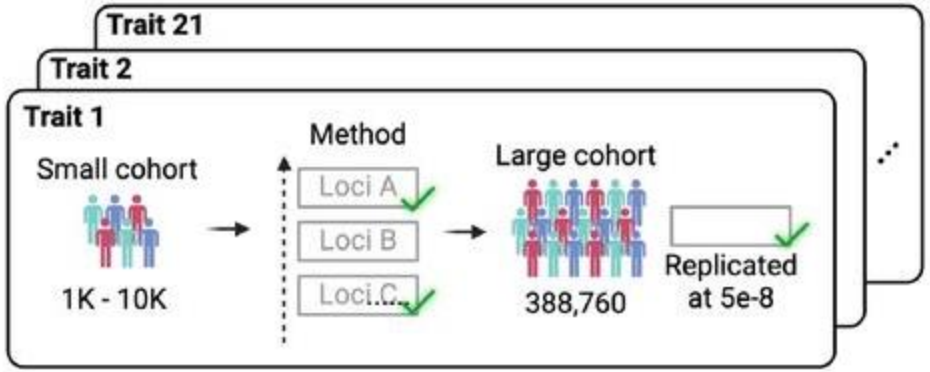


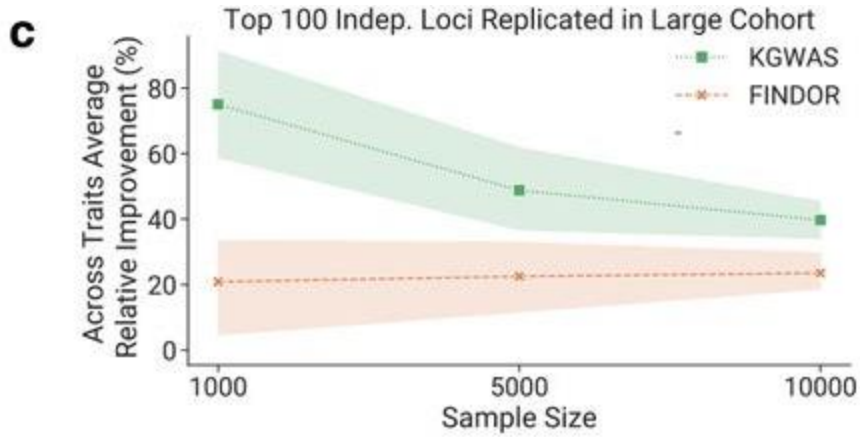


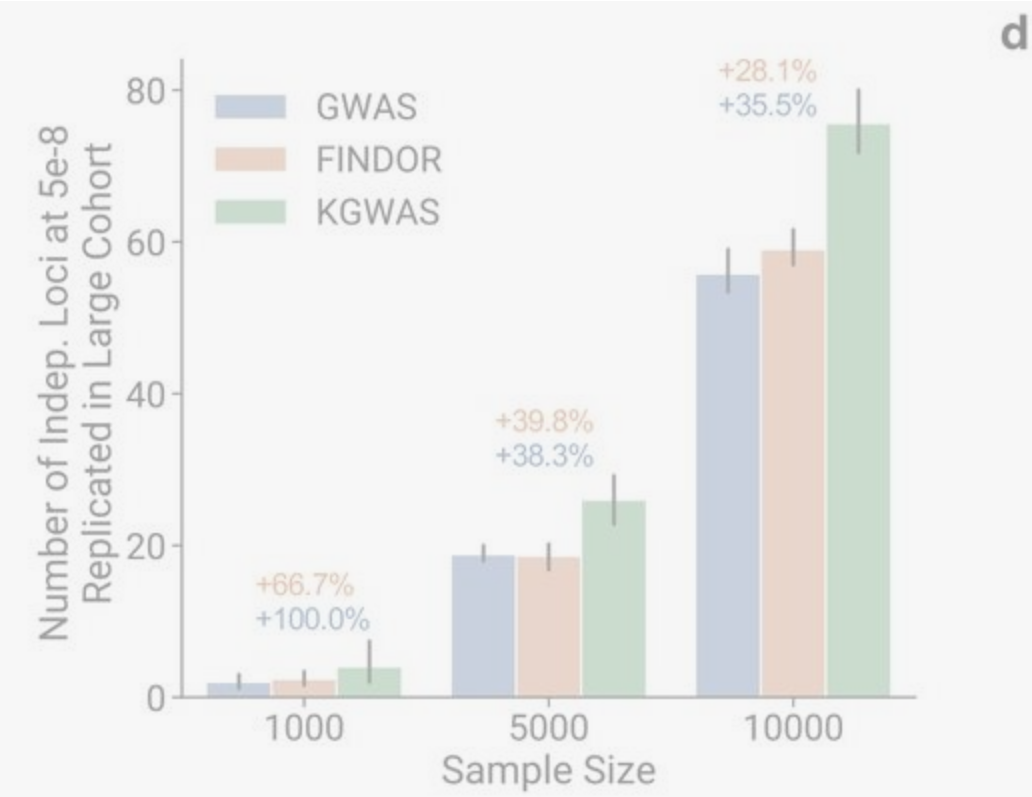
How does KGWAS boost power to detect associations?

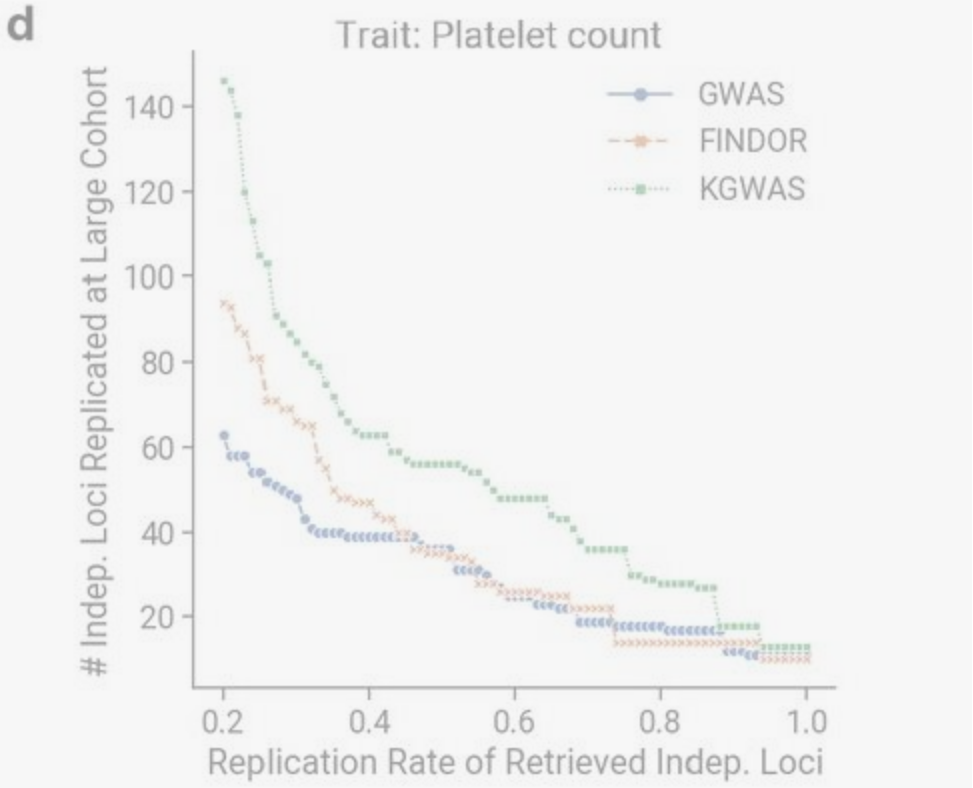
Test: Run KGWAS on a subsampled GWAS cohort, and compare detected associations to the full cohort:

2) This effect is consistent across multiple traits when looking at top 100 SNPs (panel c)



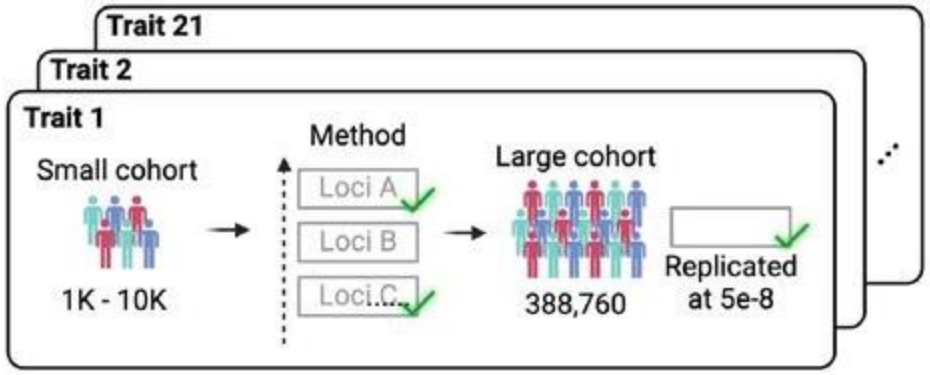


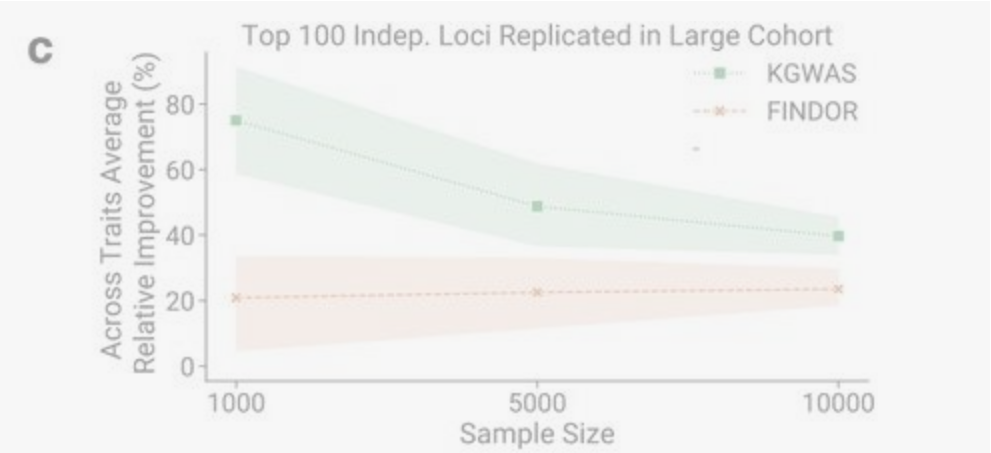




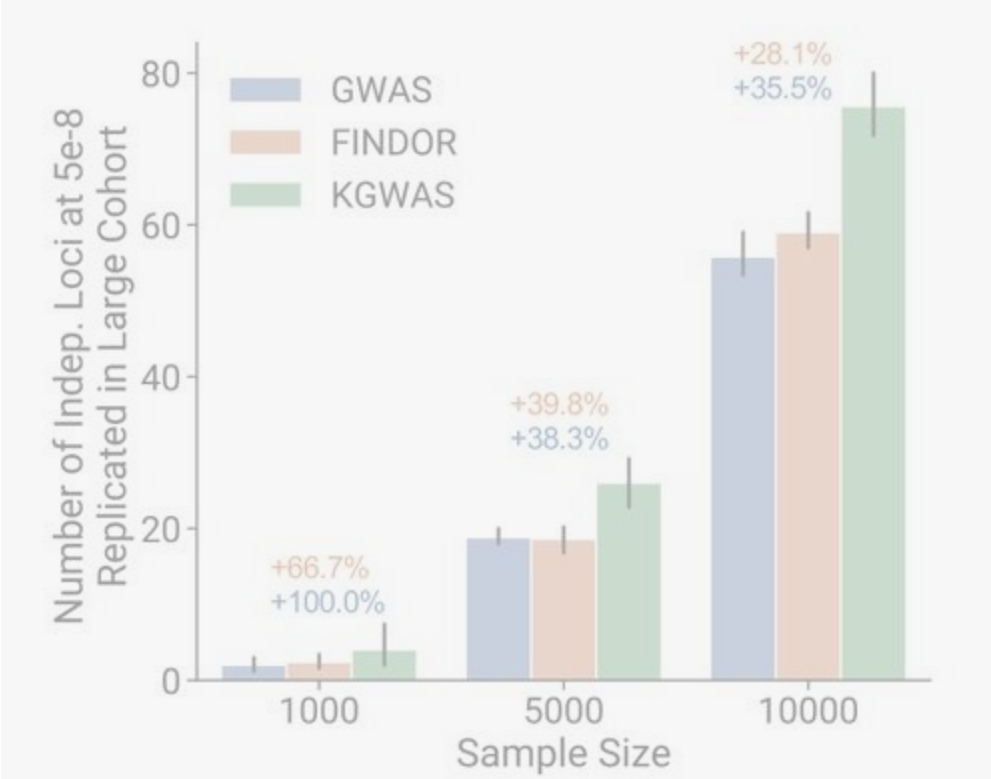
How does KGWAS boost power to detect associations?

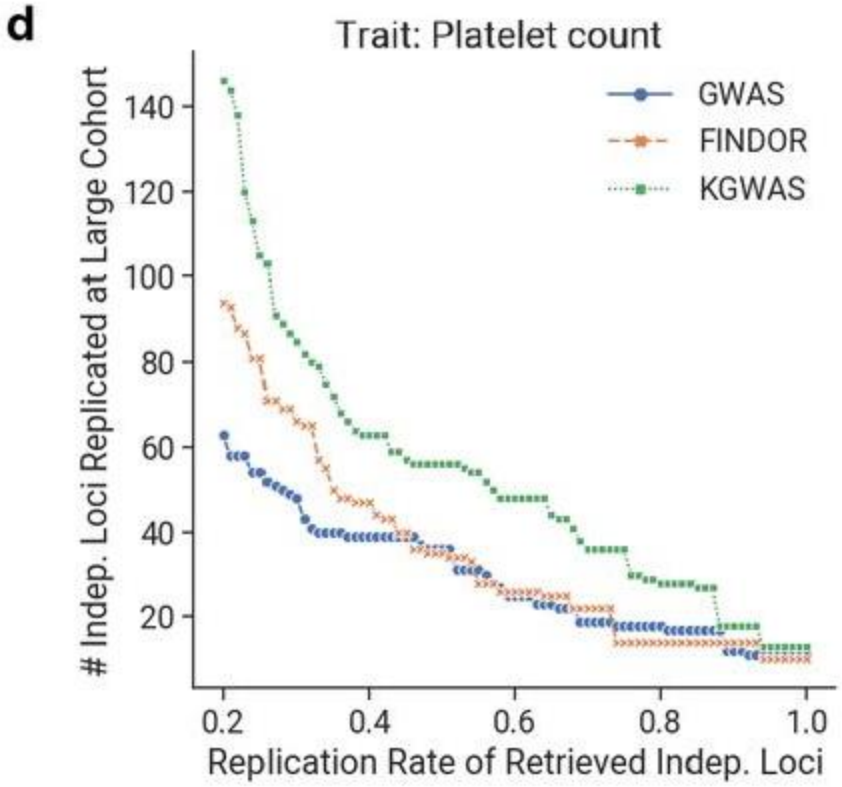
Test: Run KGWAS on a subsampled GWAS cohort, and compare detected associations to the full cohort:





3) This boost in performance was consistent across different different p-value thresholds





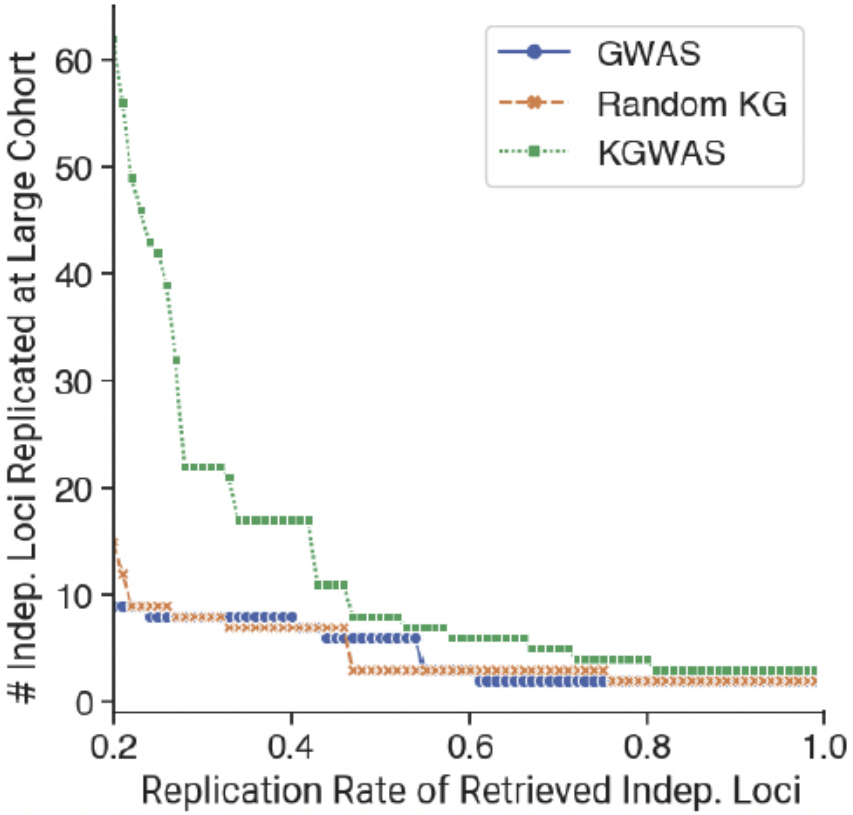
Empirical tests of KGWAS performance

1. **Does it detect the signal we want it to?**
2. **Does it actually boost power when N is low in GWAS?**
   1. Evaluations in real GWAS on subsampled cohorts vs full cohort
3. **Does the knowledge graph actually make a difference?**
4. **Is the benefit of KGWAS knowledge graph greater or less than embeddings from foundation gene models (ESM, Enformer)?**

KG indeed contributes to performance

Test: replaced KG with a "random" permuted KG and see how that performs compared to true KG (on phenotype IGF-I

Result: Performance boost is primarily driven by the real knowledge graph



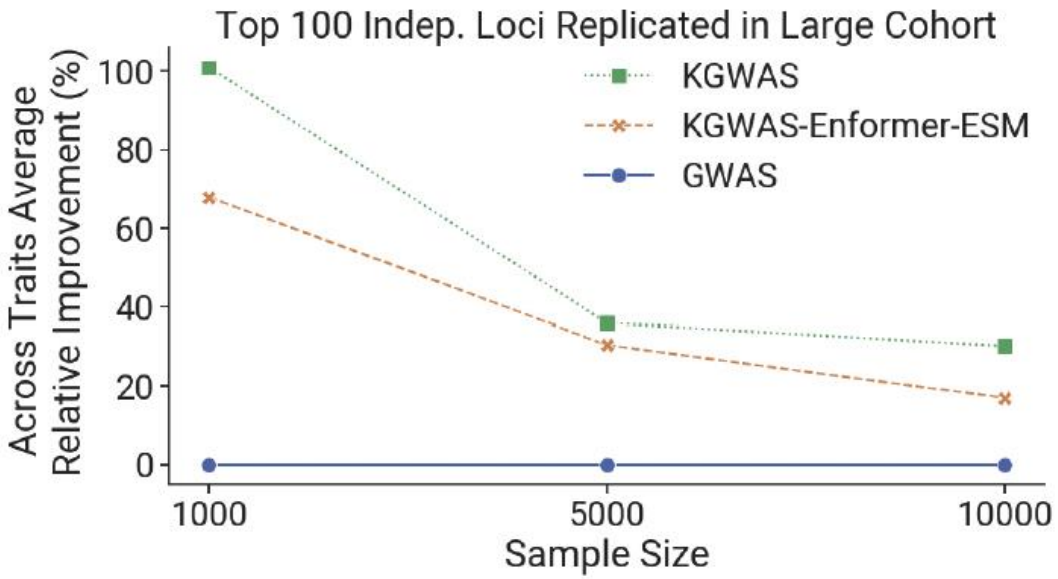
Supplementary Figure 7: Impact of using a random KG. We randomize the knowledge graph by randomly permuting edges for every edge type and then performing a subsampling analysis on IGF-1. We observe that by randomizing the KG, the performance degrades to base GWAS, showing that prior knowledge in the KG drives the most performance improvement and the structure in the KG is essential.

Empirical tests of KGWAS performance

1. **Does it detect the signal we want it to?**
2. **Does it actually boost power when N is low in GWAS?**
   1. Evaluations in real GWAS on subsampled cohorts vs full cohort
3. **Does the knowledge graph actually make a difference?**
4. **Is the benefit of KGWAS knowledge graph greater or less than embeddings from foundation gene models (ESM, Enformer)?**

KGWAS embeddings perform better in this framework than foundation model embeddings

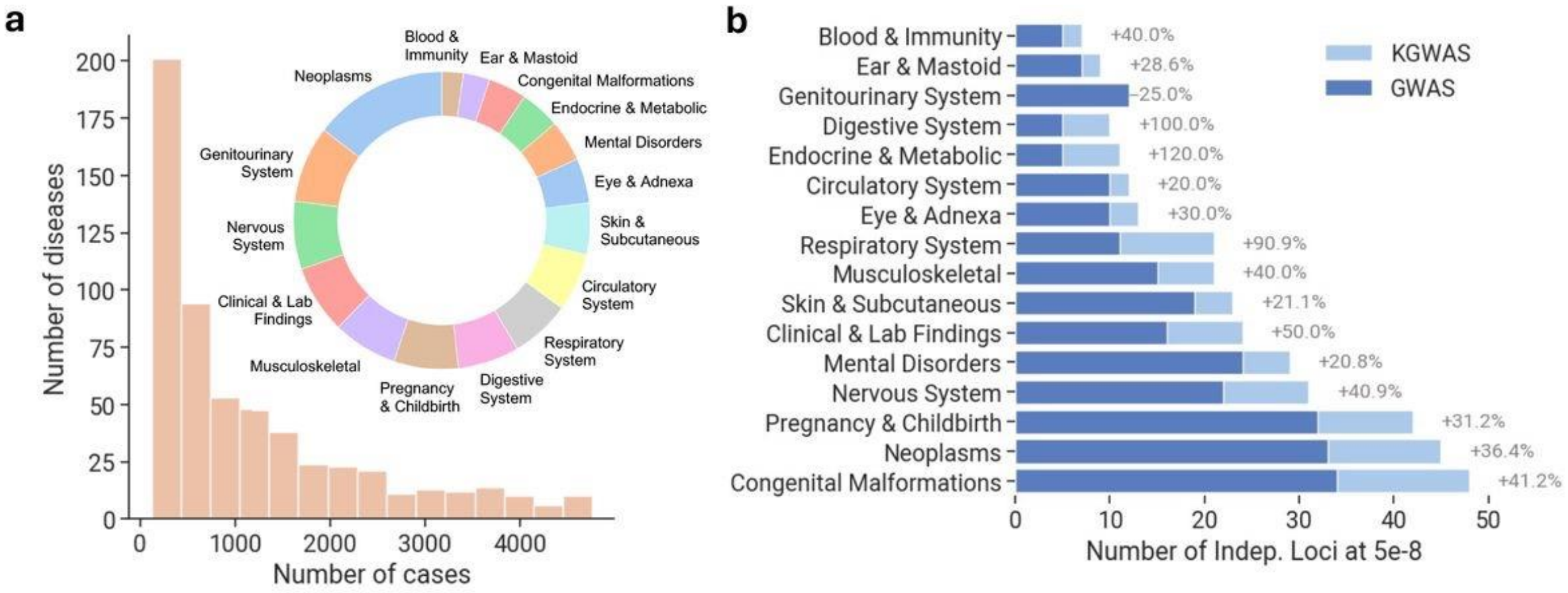
- **Test:** replace variant embeddings with those from *enformer* and gene embeddings with protein embeddings from ESM
- **Result:** This is better than GWAS, but not as good as KGWAS
- **Author's explanation:**
  - Gene embeddings: Gene-co expression (scRNA-seq) is more informative for GWAS than protein structure information (ESM)
  - Variant embeddings: BaselineLD annotations are "orthogonal" to functional genomics network data, while enformer captures functional genomics information



Supplementary Figure 9: Performance of using alternative node embeddings. We switched out the initialized embedding of KGWAS from scRNA-seq profiles to ESM embedding and baselineLD features to enformer embedding and then reported the subsampling analysis across three sample sizes. We observe that it has consistent improvement over base GWAS but underperforms compared to KGWAS. We suspect that it is because in human genetics discovery, gene co-expression patterns captured by scRNA-seq is most informative compared to protein structure information in ESM. For the enformer embedding, it is largely capturing functional genomics information, which overlaps with the functional genomics KG. In contrast, a variant-level baselineLD feature provides more orthogonal information.

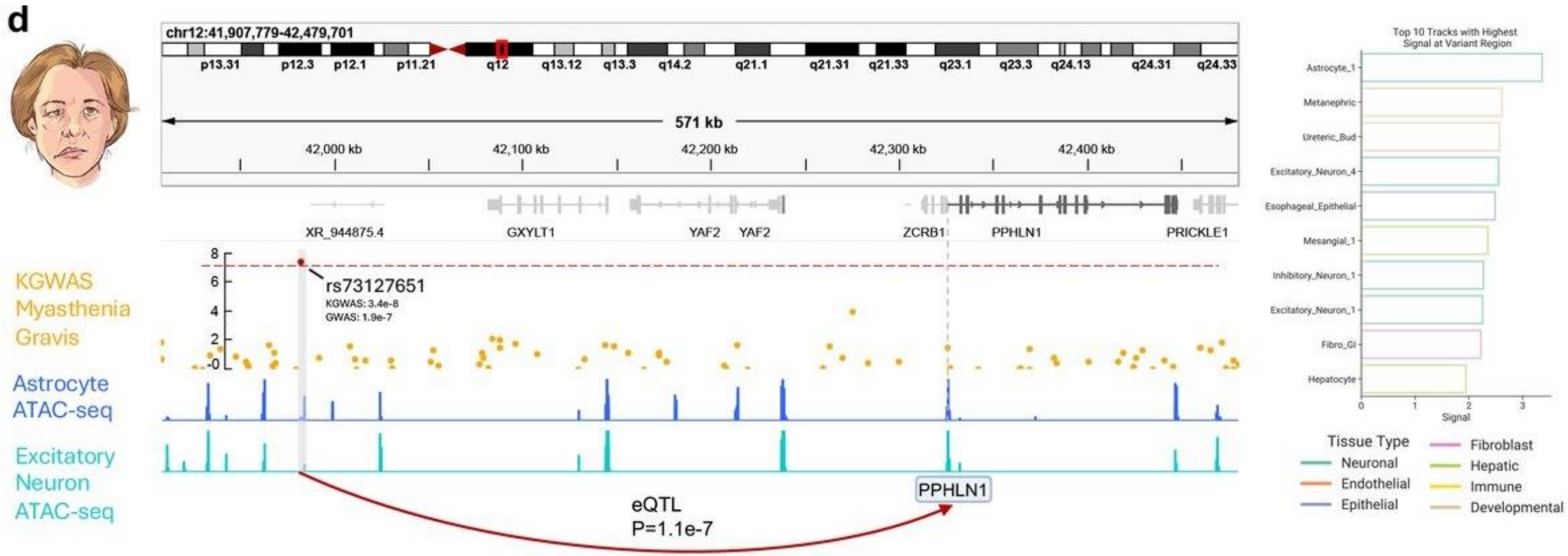
KGWAS detects novel associations across 554 uncommon diseases

- ”Uncommon disease”: N cases < 5K in UKB
  - Includes 141 rare disease with N cases < 300



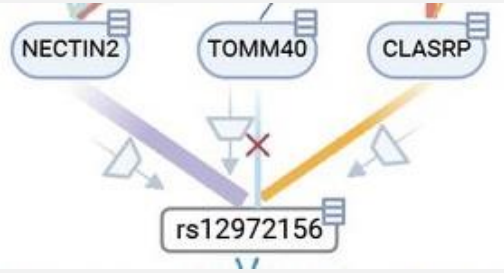
KGWAS yields significant associations for Myasthenia Gravis

- MG a rare autoimmune disorder affeecing neuromuscular junctions
- Affects (20/100,000 people)
- novel association: rs73127651



Interpreting GANN through attention scores

Goal: identify the most useful/informative relations between nodes



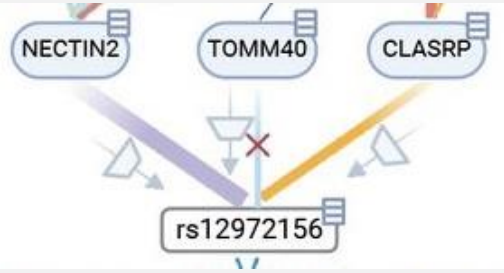
Attention score

$$\alpha_{i,j,r}^{(l)} = \frac{\exp\left(e_{i,j,r}^{(l)}\right)}{\sum_{k \in \mathcal{N}_r} \exp(e_{i,j,r}^{(l)})}$$

Relative score on each edge with relationship type r, scaled to reflect the importance within a certain set of relationships ($\mathcal{N}_r$)

Interpreting GANN through attention scores

Goal: identify the most useful/informative relations between nodes



Attention score

$$\alpha_{i,j,r}^{(l)} = \frac{\exp\left(e_{i,j,r}^{(l)}\right)}{\sum_{k \in \mathcal{N}_r} \exp(e_{i,j,r}^{(l)})}$$

Relative score on each edge with relationship type r, scaled to reflect the importance within a certain set of relationships ($\mathcal{N}_r$)

Challenge:

- multiple relationship types, and attention scores are based on per-relationship weight.
- These scores have different distributions across relationships, so **not directly comparable** when prioritizing relationships

Solution: scale to find edge importance z-score

1) $$z_{i,j,r} = \frac{\alpha_{i,j,r} - \mu_r}{\sigma_r}$$

"relation-wise z-score"

μ_r:weighted average for relation r

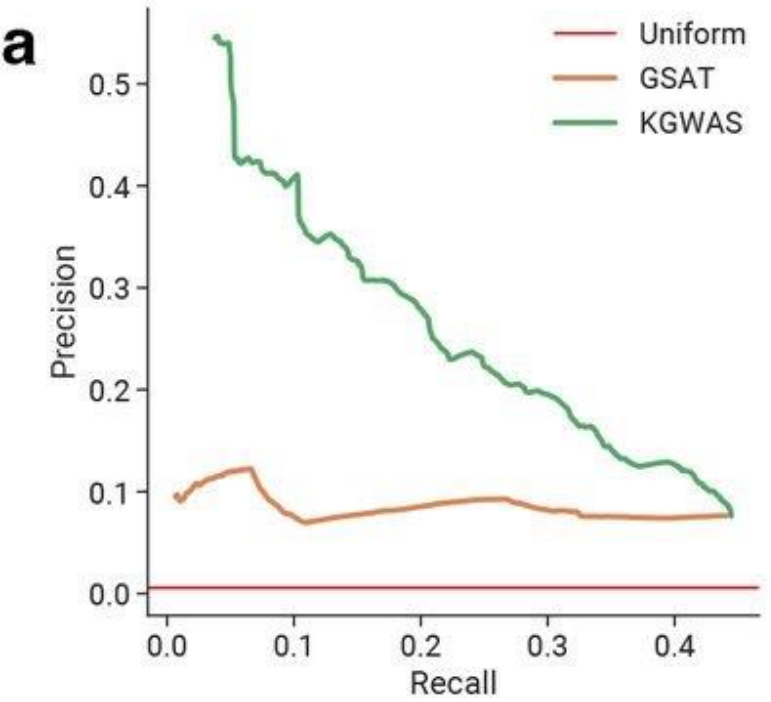
σ_r: standard deviation for relation r

2) $$z_{i,j} = \max_{r \in \mathcal{J}} z_{i,r,j}$$

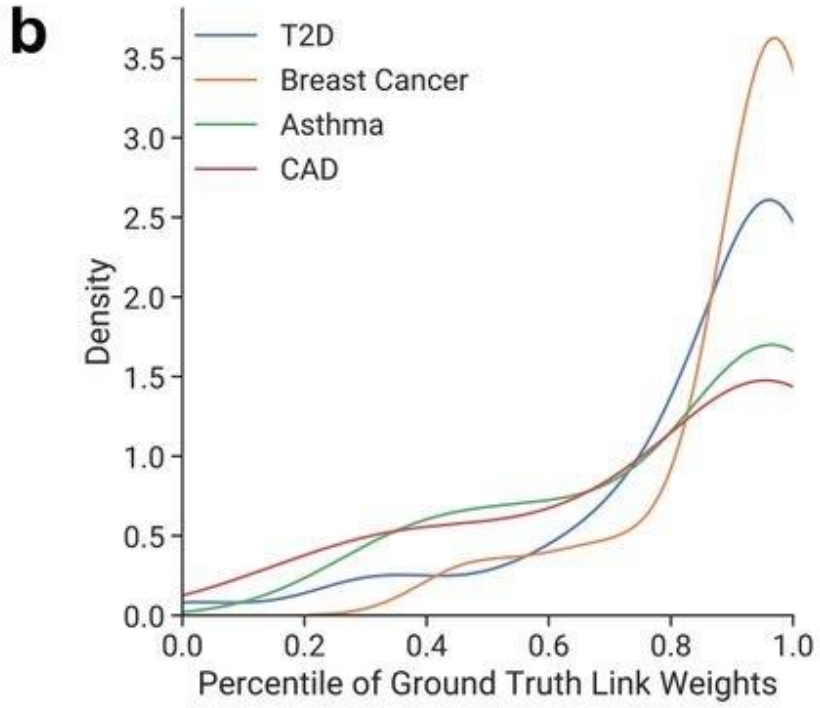
"edge importance"

Evaluating the utility of attention z-scores

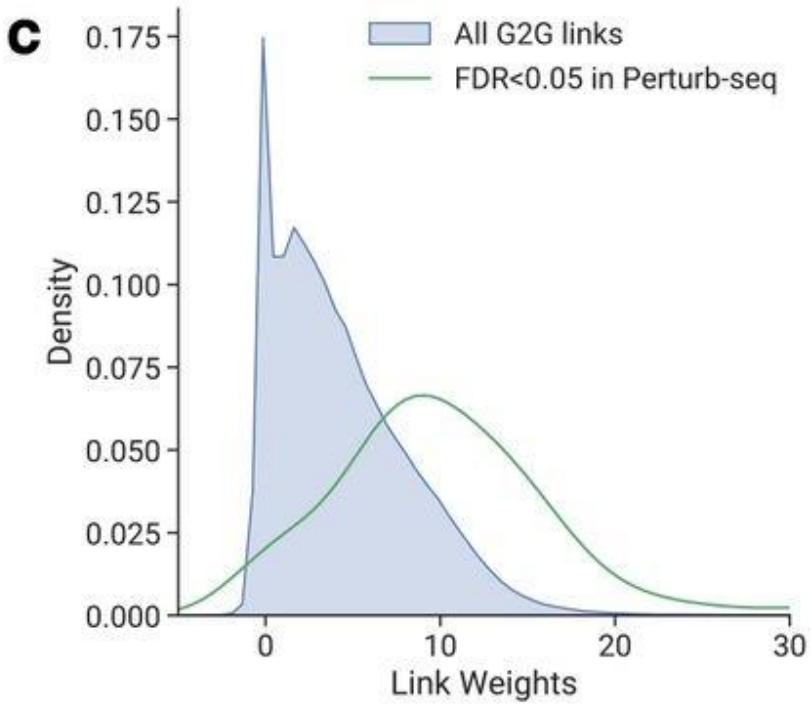
KGWAS Network interpreter



Simulations evaluating V2G, G2G, and G2P links

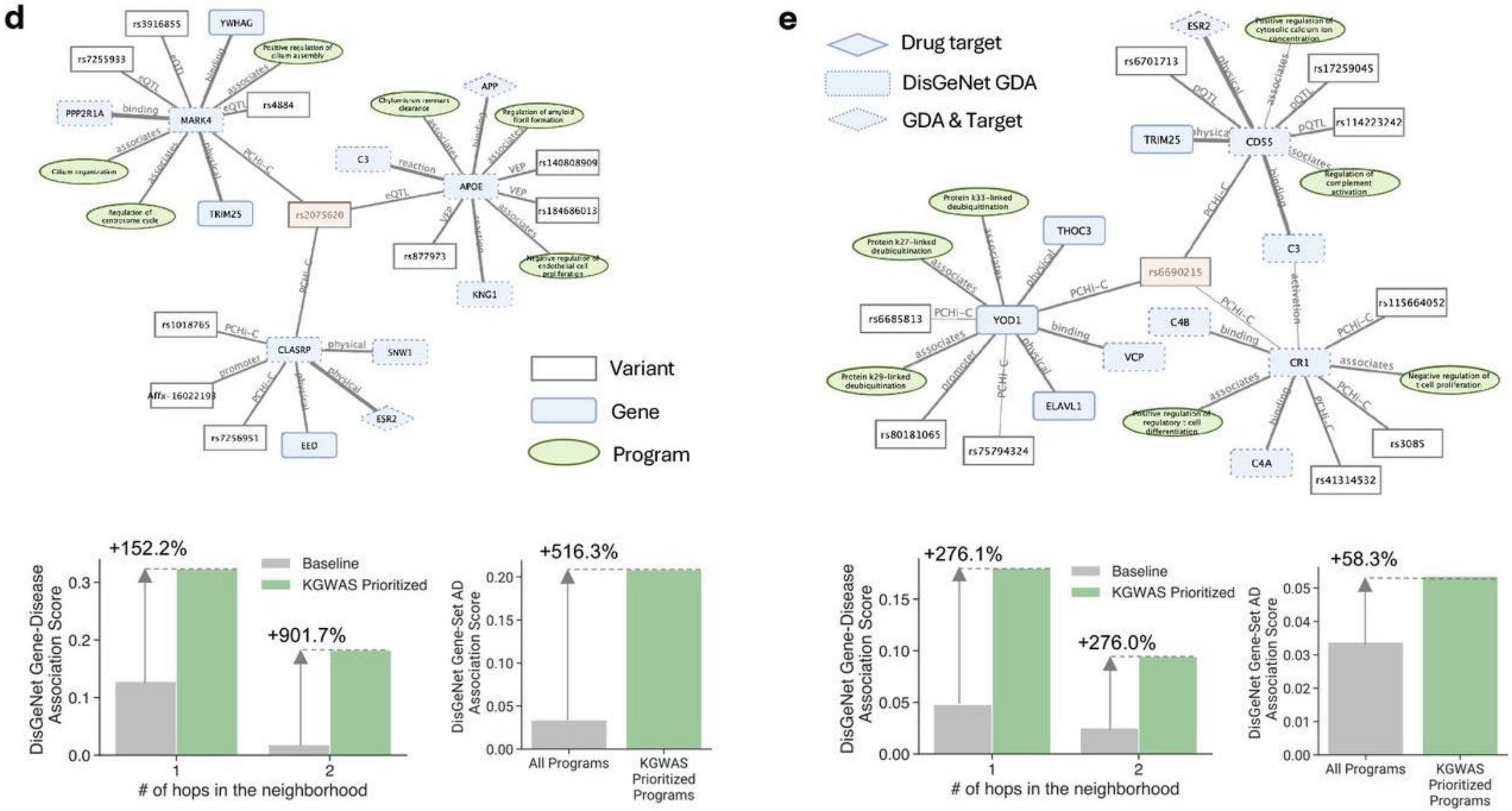


V.S. Open Targets "ground truth" disease-specific V2G links



G2G links in mean corpuscular hemoglobin versus perturb-seq test on hematopoietic cells

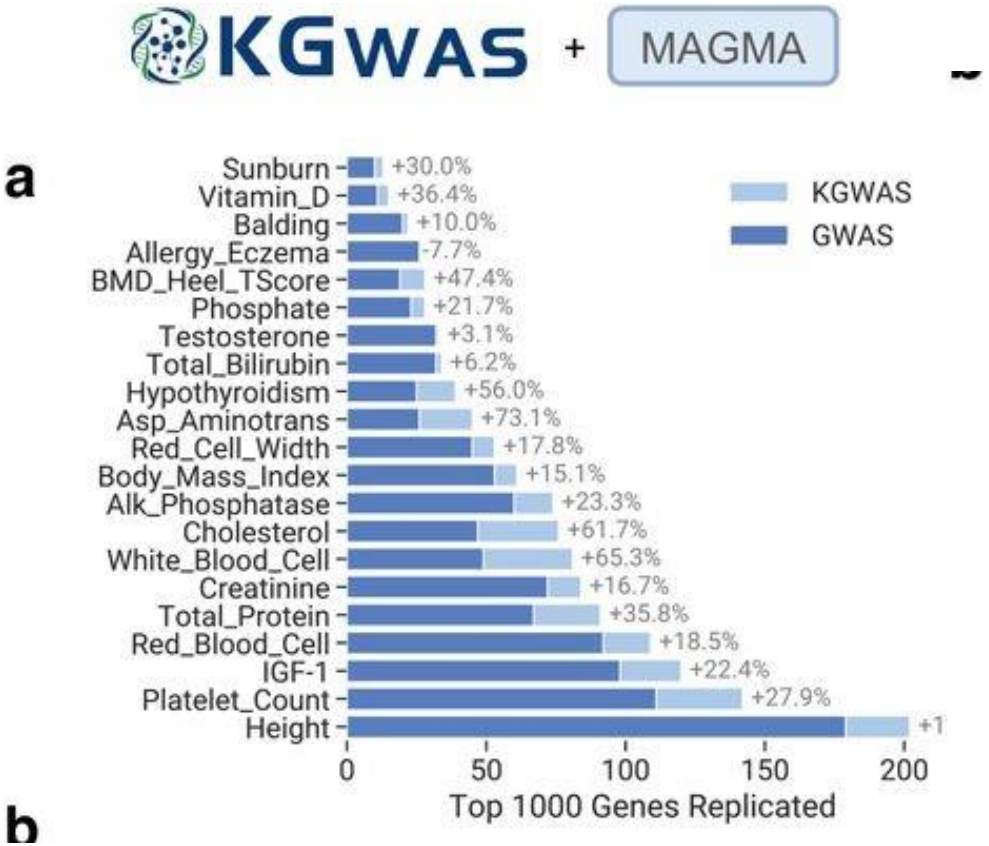
Novel and plausible findings in AD: + 2 new loci



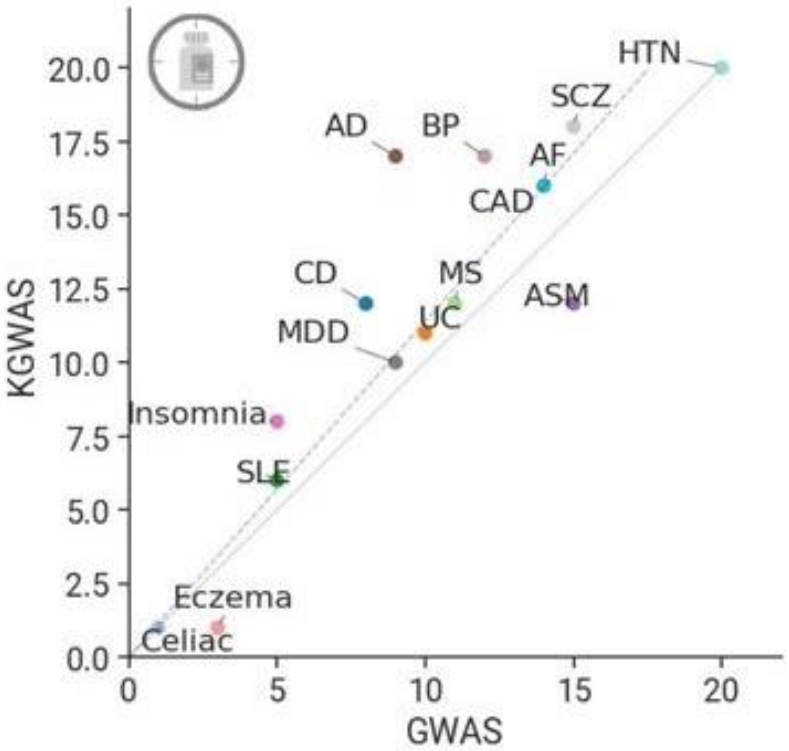
KGWAS can improve post-GWAS analysis

MAGMA: widely used procedure for prioritizing genes from GWAS summary statistics using

Test: use a downsampled cohort (N=1000) to estimate GWAS, and see how many genes you replicate with the full GWAS



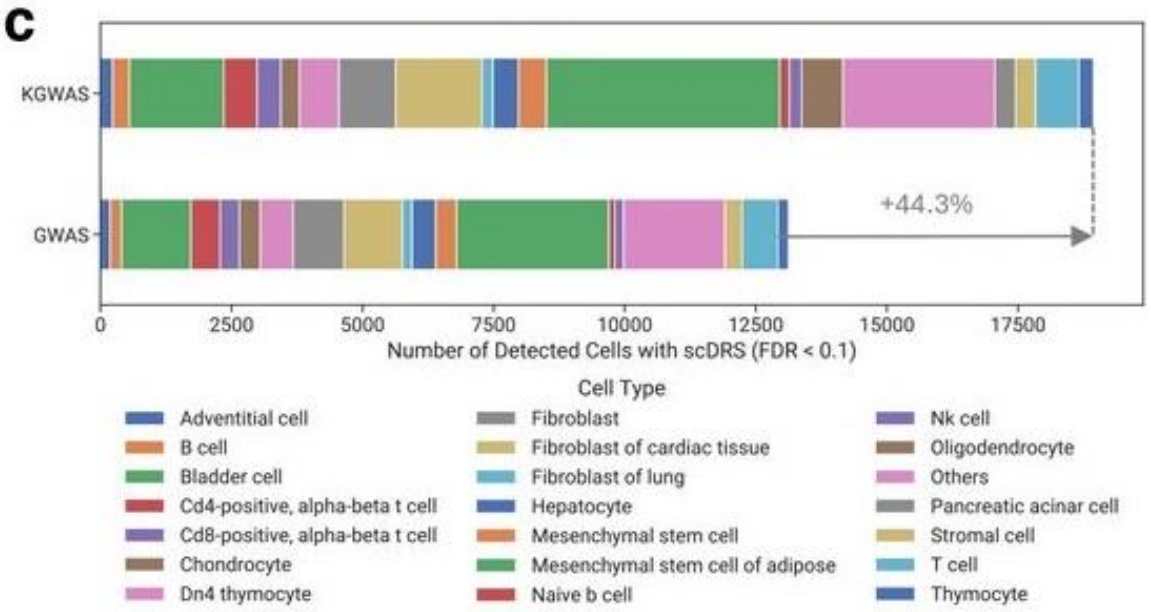
Number of drug target genes



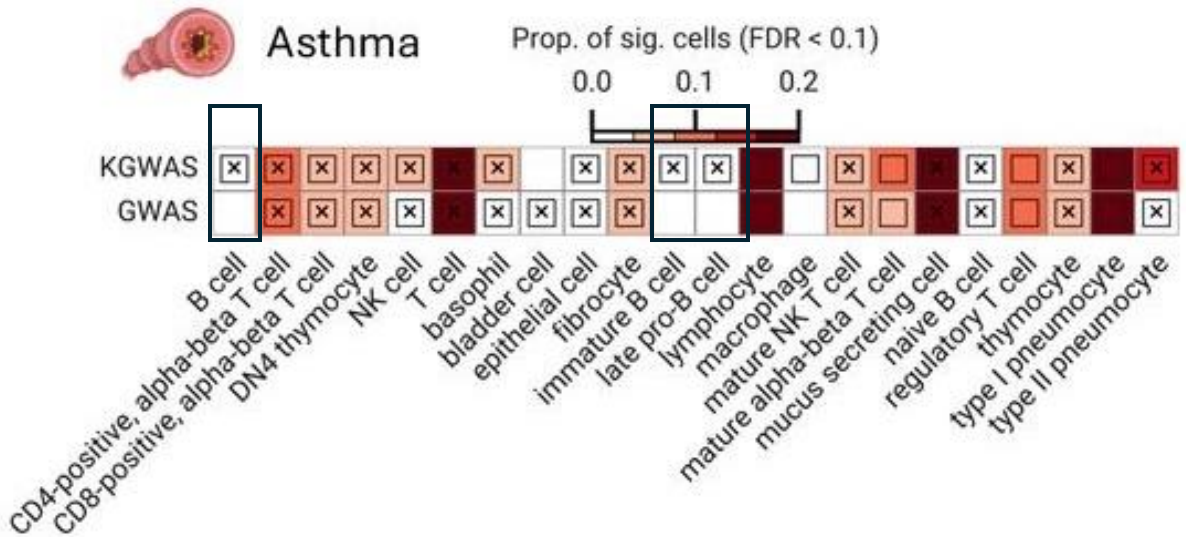
KGWAS can improve post-GWAS analysis

scDRS: identifies cell types with high expression on disease-associated genes from GWAS (prioritize disease-relevant cell types)

Test: evaluate at 93 heritable UKBB diseases, using Tabula Muris scRNA-seq data (mouse)



Cell types detected across 93 heritable diseases



In asthma, KGWAS identifies B-cells, which have ea known role in asthma pathogenesis

Discussion and possible shortcomings

- Some things they mention:
  - Input data limited on SNPs: likely not causal variants being targeted.
  - Input data is focused on common variants
  - Only gives p-values, so directional effects aren't available

- Some of my concerns:
  - Ambiguity about how the gene-level embeddings are initialized
  - Possible risk of double-dipping might effect result interpretability, e.g.
    - Use GWAS to train and learn χ^2, then use it again to scale p-values
  - Open Targets relationships are input, and then they tests against this in their evidence of biological relevance section
  - No provided evidence that they are appropriately accounting for LD effects with objective function ?
  - Comparison against standard GWAS doesn't account for cost of model complexity