ChatGPT

Expert Review and Recommendations for the Hierarchical Heterogeneous GNN Architecture

Current Approach: The user's architecture is an innovative hierarchical heterogeneous GNN for multi-omics cancer subtyping. It builds **Level 1** patient-specific graphs (connecting mRNA, CNV, methylation, miRNA features within each patient via known gene mappings) and a **Level 2** inter-patient graph (connecting patients by shared pathway/gene alterations, with weighted edges for alteration frequency and pattern similarity). This design leverages prior biological knowledge (gene–target links, pathways) to integrate multi-omics data. Such use of knowledge-driven, multi-level graphs is well-aligned with emerging trends in multi-omics integration [1]. However, there are several methodological improvements and additions that could **enhance the model's performance, robustness, and biological insight**. Below, we provide detailed suggestions based on recent literature (e.g. **CtAE**, **M^2CGCN**, **OmicsCL**, etc.), organized by theme.

1. Advanced Training Objectives and Contrastive Learning

- **Incorporate Contrastive Multi-View Learning:** Adding a contrastive learning objective can greatly improve unsupervised representation quality. Recent methods like *M^2CGCN* and *OmicsCL* use contrastive losses to align and separate subtype-related patterns in the latent space. For example, M^2CGCN learns high-level patient features via a contrastive loss between omics views, significantly improving clustering of subtypes [2]. Similarly, OmicsCL introduces a **survival-aware contrastive loss** that encourages patient embeddings to reflect survival pattern similarities [3]. By sampling positive pairs (e.g. different augmentations of the same patient or patients with similar outcomes) and negative pairs (dissimilar patients), the model can be trained to **pull together embeddings of biologically similar patients and push apart others**, sharpening the separation of subtypes. Incorporating contrastive learning would likely enhance the current GNN's ability to discover clusters that are **both data-driven and clinically meaningful** (e.g. correlated with survival [3]).

- **Integrate Cluster Awareness or Pseudo-Labels:** Because the goal is unsupervised subtyping, one can introduce a **clustering objective** into the training loop. *M^2CGCN*, for instance, performs k-means on the high-level embeddings and then uses a "maximum matching" between cluster assignments and latent label predictions to reinforce clustering consistency [4]. This kind of approach (related to deep clustering frameworks) could be adapted to the hierarchical GNN – e.g. periodically clustering patient embeddings and using a loss that maximizes agreement of the GNN's output with these cluster assignments (or encourages separation of these clusters). Such a **self-supervision via pseudo-labels** helps drive the model to form well-separated subtype clusters beyond what a reconstruction or plain graph loss would do [4]. Likewise, incorporating **survival or clinical information as weak supervision** – for example, adding a Cox partial likelihood loss or ranking loss on the patient embeddings – could guide the model to encode prognostic signals (as done in OmicsCL's survival-aware training [3]). Even though the training remains *unsupervised* regarding subtype labels, these additions act as gentle "hints" to prefer embeddings that have real clinical relevance.

- **Hybrid Learning Objectives (Reconstruction + Contrastive):** It may be beneficial to **combine multiple learning objectives** to balance preserving information and learning discriminative features. One strategy is to train the Level-1 patient graphs in an **autoencoder fashion**: have the GNN encoder produce latent feature embeddings and use a decoder to reconstruct the original multi-omics features or graph structure. Indeed, M^2CGCN uses a **dual reconstruction loss** (one for node attributes, one for graph structure) on the low-level features for each omics, ensuring that the embeddings retain important information from each modality [2] [5]. Jointly, a contrastive or clustering loss on the Level-2 patient representations can ensure these embeddings are also useful for distinguishing subtypes. This *multi-task training* (reconstruction + contrastive) can prevent the model from simply fitting noise – the reconstruction term preserves intra-patient information, while contrastive/clustering terms ensure *inter-patient* distinctions are captured. Such combined objectives have been shown to outperform single-task training, especially on small, high-dimensional omics datasets [2] [6].

2. Omics-Specific Encoders and Feature Representation

- **Use Omics-Specific Encoding Networks:** Given the heterogeneity and scale of each omics, introducing dedicated encoders per omics type can improve feature extraction. Instead of feeding all features directly into the graph, consider **preprocessing each omic with a neural encoder (e.g. autoencoder or VAE)** to produce a lower-dimensional, denoised representation. This approach was used in models like HeteroGATomics, which employs a *multi-agent system for joint feature selection* and constructs separate feature networks for each omic [7]. By learning an informative compressed representation for mRNA, CNV, methylation, etc., the model can mitigate the "large *p*, small *n*" problem and reduce noise before graph integration [8] [9]. For example, one could train denoising autoencoders on each omic dataset to filter out technical noise and reduce dimensionality (many studies use PCA or autoencoders to this end [8] [10]). These encoders can be integrated into the GNN pipeline (learned jointly or pre-trained) so that each patient's node features are more compact, **enhancing the GNN's effectiveness** on high-dimensional data.

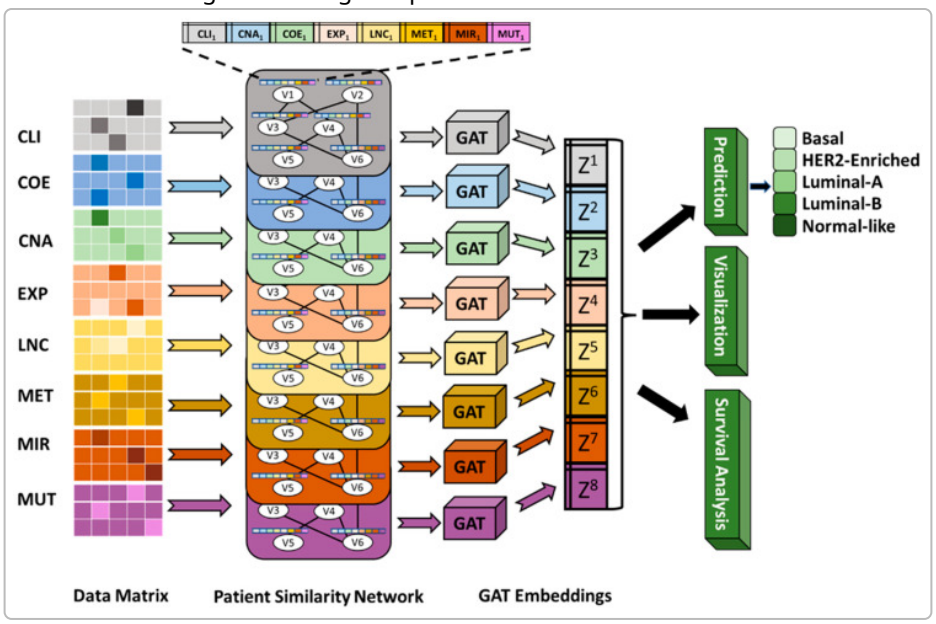
- **Preserve Modality-Specific Signals:** It's important that integration does not completely wash out signals unique to each omics. Recent methods emphasize learning **both shared and specific representations** for each modality. For instance, the MOCSS model (Chen *et al.*, 2023) trains separate autoencoders to capture joint (cross-omics) factors versus individual-omics factors, with constraints to keep them orthogonal [11]. This ensures that, say, a DNA methylation-driven subtype signal is not overshadowed by mRNA patterns common to all patients. In the current graph design, one improvement could be to allow **omics-specific latent factors** in the patient embeddings – e.g. part of each patient's representation could be specific to one data type. Mechanisms like **multi-head attention or gating per modality** could achieve this: the model learns how much each omic contributes to a patient's final embedding. This way, if one omic (e.g. miRNA) is noisy or less informative, the model can down-weight it, whereas a strongly informative omic can have more influence. Preserving some modality-specific components aligns with the idea of not ignoring "individual information within each omics level" during integration [12] [2]. In practice, the Level-1 heterogeneous graph could be extended so that each omics layer has an independent embedding pipeline before they are merged at patient nodes. This **fusion-free multi-view approach** (processing each omic separately up to some point) was a key design in M^2CGCN and HeteroGATomics [13] [14], and helps retain the unique structure of each data type.

- **Denoising and Robust Feature Learning:** The user currently uses all features (after removing >20% missing) without additional selection. This comprehensive approach maximizes data usage but can introduce a lot of noise and redundant signals. To tackle this, consider incorporating techniques for **robust feature learning** such as Contractive or Denoising Autoencoders on the input features. The *SubtypeCtAE* framework, for example, uses a **Contractive Autoencoder (CtAE)** to learn features that are *less sensitive to small input perturbations*, thereby denoising the omics data [15]. By adding a contractive penalty (Jacobian regularization) or noise injection during encoding, the model can learn stable representations that aren't dominated by experimental noise. The authors of CtAE found this improved the downstream clustering and yielded higher C-index and more significant survival differences than using raw features [16]. In the GNN context, one could apply similar regularization to node feature encoders or even to the GNN embeddings (e.g. adding Gaussian noise to features in each epoch as data augmentation). **Feature selection** can also be guided by external knowledge: CtAE followed its autoencoder by a **Cox regression-based feature selection**, choosing features significantly associated with survival for integration [17]. The user's pipeline could emulate this by either filtering input features that show little variance or correlation with outcomes, or by adding a sparsity constraint/attention such that only the most predictive nodes strongly influence the patient embedding. Overall, introducing **feature selection and denoising steps** will combat the high dimensionality and noise, leading to cleaner inputs for the graph model. This addresses a known challenge that many integration methods handle by heavy pre-filtering; doing so in a learned way can be more effective than a naive variance filter [18].

3. Enhancing Graph Design with Biological Knowledge

- **Intra-Patient Graph Improvements (Level 1):** The current intra-patient heterogeneous graphs connect different omics features based on gene mappings (e.g. miRNA–target gene, CpG–TSS, gene–CNV). To further enrich this graph, the user could integrate additional *biological interaction networks*. For example, adding **gene–gene edges** (protein–protein interactions or co-expression links) would create a more connected graph of mRNA nodes, so that the GNN can propagate signals between functionally related genes in the same patient. If two genes frequently interact or are in the same pathway, linking them in the patient graph could help the model capture pathway-level activation patterns. Similarly, linking miRNAs that co-regulate targets, or methylation sites that lie in the same region or regulate the same gene, could be beneficial. This essentially incorporates a prior knowledge graph (from sources like KEGG, Reactome, BioGRID, etc.) into each patient's graph. It leverages the idea of **knowledge-driven priors**, which has been highlighted as an effective direction in recent GNN cancer studies [1]. Another idea is to introduce **"hub" nodes to represent pathways or gene sets**: e.g. a node for a specific pathway that connects to all member genes in that patient (if those genes are altered). This would create a hierarchical graph *within* the patient – genes connect up to pathway nodes, which could then connect to other omic features (like a CNV node might connect to a pathway if any gene in that pathway has a CNV). Such hierarchical modeling could make it easier for the GNN to pool information at a pathway level. In fact, *hypergraph* approaches (where a hyper-edge connects a group of features) have been used to capture higher-order relationships in multi-omics [19]. For instance, HyperTMO (a hypergraph-based model) can connect multiple omics features in one hyper-edge to signify a joint involvement in a biological process [19]. Implementing a hypergraph or a two-tier node system (features and pathway nodes) in Level 1 could improve the model's ability to recognize coordinated aberrations in the same pathway within a patient.

- **Inter-Patient Graph Enhancements (Level 2):** Constructing the patient–patient similarity network is a critical step, and there is room to make it more data-driven and multi-faceted. Currently, edges are weighted by shared pathways or gene alterations, frequency, pattern similarity, and clinical relevance. One improvement is to adopt a **multi-view graph** approach for patient similarities. Instead of one combined network, you could build **separate patient graphs for different alteration types or omics** and then integrate them. For example, one graph where edges reflect *gene mutation/CNV overlap*, another where edges reflect *mRNA expression profile correlation*, another for *methylation pattern similarity*, etc. This is analogous to what the recent *MOGAT* framework does: it constructs multiple patient similarity networks (one per data type, including clinical) and applies GAT on each, producing embeddings that are later fused [20] [21]. This separation prevents dominance of one data type and allows the model to assign modality-specific importance to patient relationships. You could then either fuse these graphs into one heterogeneous patient-level graph with different edge types, or learn from each graph and combine the learned patient embeddings (e.g. by concatenation or attention fusion). **Figure: Multi-Graph Integration:** For example, the MOGAT method builds separate graphs for clinical (CLI), co-expression (COE), copy number (CNA), mRNA (EXP), lncRNA, methylation (MET), miRNA (MIR), and mutation (MUT) data, applies a GAT to each, and then combines the resulting embeddings for prediction



. Each omics view thus contributes a neighborhood structure among patients, and attention mechanisms in GAT weigh the most important neighbors within each view. Adopting a similar multi-view strategy, the user could ensure that **shared pathway mutations** form one view, while **expression profile similarity** (perhaps via correlation or Euclidean distance in expression PCA space) forms another, etc. This would enrich the inter-patient connectivity beyond just pathway overlaps.

- **Learnable or Dynamic Edge Weighting:** Whether using a single patient graph or multi-graphs, it's beneficial to let the model *learn* the importance of connections rather than relying solely on preset weights. Graph Attention Networks (GAT) already help by assigning attention coefficients to neighbors, which addresses one known issue of GCN-based methods (i.e. not differentiating neighbor importance [21] [22]). With GAT at Level 2, edges that connect very similar patients should automatically get higher attention weights, whereas spurious connections will be down-weighted

[21]. Beyond this, the user could implement a **learning mechanism for edge weights** themselves. For instance, use the patient feature embeddings to compute a similarity (like a learned metric) that updates the adjacency matrix iteratively – a form of *graph structure learning* where the graph is refined during training. Some recent contrastive GNN approaches include modules to refine the graph structure by comparing learned embeddings (reinforcing true connections and dropping unlikely ones), which could be considered. Additionally, ensure the graph connections are **pruned for noise**: perhaps only keep the top *k* most similar patients for each patient (k-NN graph) rather than fully connecting everyone who shares any pathway. This can avoid situations where a rare artifact (e.g. two patients sharing a single low-frequency mutation) creates a link that confuses the model. Using a threshold on the edge weight or statistically determining significant overlaps will keep the network more robust. In summary, the inter-patient graph would benefit from a more nuanced construction – potentially **multi-relational edges** (different edge types for different shared alterations) and a **semi-supervised graph refinement** strategy. Classic methods like *Similarity Network Fusion (SNF)* might inspire here: SNF takes multi-omic similarity matrices and **fuses them via an iterative diffusion process**, effectively weighting patient–patient links that are consistently strong across data types [23]. The user could mimic this by initializing the patient graph as a weighted sum of several similarity measures (pathway, gene, expression, etc.) or by actually performing SNF to get an integrated patient similarity network to use as Level 2 input.

- **Include Clinical and Demographic Features:** Since the question mentions clinical relevance in edge weights, it's worth explicitly stating that *clinical data* (age, stage, tumor grade, etc.) can be incorporated either as node features or as additional patient similarity criteria. If not already done, one could add an edge between patients sharing a clinical characteristic (e.g. same stage) or simply include these variables in computing patient similarity. Some frameworks treat clinical variables as an additional modality in integration [20]. Because clinical factors can heavily influence outcomes, including them may improve the subtype identification (though one must be cautious not to have the model trivialize subtypes by a clinical factor if the goal is to find molecular subtypes). At the very least, adding clinical info as node features on the patient nodes in Level 2 could allow the GNN to consider them in its embedding – potentially enhancing the survival signal in clusters.

4. Handling Omics Imbalance and Noisy Data

- **Modality Imbalance:** In multi-omics data, different layers can have vastly different feature counts, value scales, and signal-to-noise ratios. The current approach treats all features equally after preprocessing, which could lead to **dominance of high-dimensional or high-variance omics** (e.g., gene expression with 20k features might overwhelm miRNA with <1k features). To address this, consider balancing techniques such as **feature scaling per omic and loss weighting**. Ensure that each omic's feature set is appropriately normalized (z-scores or unit-variance) so that the GNN doesn't focus on one modality just because of scale. You can also introduce weights in the loss function or attention mechanism to give under-represented modalities a fair contribution. For example, in a heterogeneous GNN, one could apply a **modality-level attention**: the model learns a weight for "mRNA vs CNV vs methylation vs miRNA" for each patient or each layer of the network, indicating which modality is most informative for that context. This idea is supported by the success of attention-based integration frameworks: they automatically learn to focus on the most relevant inputs [21] [22]. By doing so, if one omic is particularly noisy for a patient, the model can rely on other omics instead (dynamic modality weighting).

- **Noise and Outlier Robustness:** Removing features with >20% missing data and applying KNN imputation is a good start to handle technical noise. Beyond that, **advanced imputation or noise modeling** could improve data quality. For instance, a simple improvement is to use a model-based imputer (like a VAE that estimates missing values with uncertainty) instead of KNN, though this may be outside the GNN itself. Within the GNN, techniques like **dropout** on node features and edges during training can make the model resilient to noise – essentially simulating missingness or unreliable measurements and forcing the model to learn redundancies. The contractive/denoising autoencoder approach mentioned earlier also directly tackles noise in features [15]. Moreover, one can identify and down-weight outlier patients or features: e.g. if a patient has an extremely unique profile that might just be technical artifact, the model could recognize this through a higher reconstruction error or lower similarity to others, and one could reduce its influence (perhaps by an outlier score threshold). In unsupervised learning, **imbalanced subtype sizes** (one subtype having very few patients) is analogous to class imbalance in supervised learning. This is rarely addressed in clustering algorithms, and indeed many methods can ignore small clusters [24]. To mitigate this, ensure that the clustering evaluation is not solely driven by the largest groups. One practical step is to perform *consensus clustering* or repeat clustering multiple times to see if small clusters persist, thereby giving confidence they are true subtypes. If a subtype is suspected to be small, you could also use a method like **oversampling in latent space** for contrastive learning (e.g. augment minority group patients more often as positives). While not straightforward without labels, careful analysis of learned embeddings (e.g. t-SNE plots) can reveal if the model is compressing minority patterns into larger clusters. If so, adjusting model capacity or using a **temperature parameter in contrastive loss** can sometimes help separate subtle differences by making the loss more sensitive to smaller distances.

- **Evaluation Feedback Loop:** Finally, use the evaluation metrics (C-index, p-values) as part of the model selection. If certain settings yield better survival separation (higher C-index, lower log-rank p-value), treat that as feedback to tune the model. For example, OmicsCL demonstrated that explicitly optimizing for survival-related structure in embeddings improved concordance with survival [3]. In the user's case, if adding a survival-based loss is too involved, simply choose the model epoch/parameters that maximize unsupervised clustering quality and survival distinction on a validation set. This will implicitly handle noise/imbalance by favoring models that find the more meaningful patterns rather than overfitting everything.

In summary, the architecture can be enhanced by **modern training techniques (contrastive multi-view learning, joint reconstruction losses)**, **better feature handling (omics-specific encoders, feature selection, denoising)**, and **refinements in graph construction (multi-relational patient graphs, knowledge-driven edges, attention-based weighting)**. These improvements address known gaps such as ignoring modality-specific information, relying on hand-crafted graphs without learning, and sensitivity to noisy high-dimensional data [18] [24]. By integrating these ideas from recent studies – e.g. *M^2CGCN*'s multi-level contrastive learning, *CtAE*'s robust feature extraction, *OmicsCL*'s survival-aware objective, and others – the user's model could achieve more **accurate, biologically insightful, and robust** cancer subtyping results. Each suggested addition comes with a trade-off in complexity, but the literature shows that these are fruitful directions to explore for state-of-the-art performance in multi-omics integration.

References: Recent works that inspired these suggestions include contrastive graph frameworks for subtyping [2] [3], autoencoder-based integration with survival filtering [15] [17], attention-driven multi-omics GNNs [21] [22], and surveys highlighting the importance of heterogeneous graphs and hybrid models

in multi-omics cancer research [1] [18]. These can be consulted for detailed methodologies that the user may adapt to further improve their hierarchical GNN architecture.

---

[1] [19] Graph Neural Networks in Multi-Omics Cancer Research: A Structured Survey
https://www.arxiv.org/pdf/2506.17234

[2] [4] [6] [12] [13] Multi-view multi-level contrastive graph convolutional network for cancer subtyping on multi-omics data - PubMed
https://pubmed.ncbi.nlm.nih.gov/39899598/

[3] [2505.00650] OmicsCL: Unsupervised Contrastive Learning for Cancer Subtype Discovery and Survival Stratification
https://arxiv.org/abs/2505.00650

[5] Multi-view multi-level contrastive graph convolutional network for cancer subtyping on multi-omics data - PMC
https://pmc.ncbi.nlm.nih.gov/articles/PMC11789786/

[7] [2408.02845] Heterogeneous graph attention network improves cancer multiomics integration
https://arxiv.org/abs/2408.02845

[8] [9] [10] [11] Autoencoders with shared and specific embeddings for multi-omics data integration | BMC Bioinformatics | Full Text
https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-025-06245-7

[14] [Literature Review] Heterogeneous graph attention network improves cancer multiomics integration
https://www.themoonlight.io/en/review/heterogeneous-graph-attention-network-improves-cancer-multiomics-integration

[15] [16] [17] Robust feature learning using contractive autoencoders for multi-omics clustering in cancer subtyping - PubMed
https://pubmed.ncbi.nlm.nih.gov/39577512/

[18] [24] MOTGNN: Interpretable Graph Neural Networks for Multi-Omics Disease Classification
https://arxiv.org/html/2508.07465v1

[20] [21] [22] MOGAT: A Multi-Omics Integration Framework Using Graph Attention Networks for Cancer Subtype Prediction - PubMed
https://pubmed.ncbi.nlm.nih.gov/38474033/

[23] M2GGCN | PDF | Cluster Analysis | Gene Expression
https://www.scribd.com/document/901463747/M2GGCN