BRAUDE
College of Engineering, karmiel

Multi-Hop Feature Quality Estimation for Unsupervised Graph Representation Learning

Capstone Project Phase A - 61998

25-2-R-10

Supervisors:

Dr. Renata Avros

Prof. Zeev Volkovich

Submitters:

Gad Azriel

Lidor Kupershmid

BRAUDE College of Engineering, karmiel

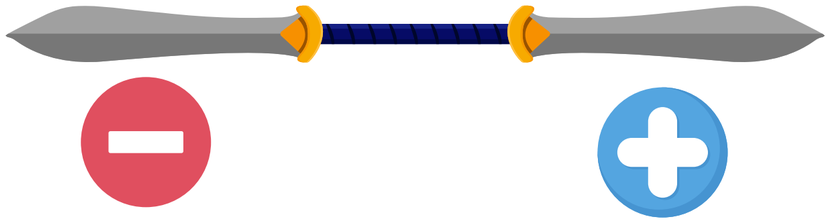
Challenges of Noisy Features in Graph-Based Models

Real-world systems, such as social networks and medical records, contain incomplete, inconsistent, and noisy data.
In graph-based models, noise propagates through the graph, affecting connected nodes.

Common sources of noise include:

- Unintentional user interactions
- Account sharing across multiple individuals
- Missing or inconsistent user preferences

Feature propagation is a **double-edged sword**



It amplifies noise across layers, causing error accumulation.

It denoises by smoothing and averaging signals.

BRAUDE
College of Engineering, karmiel

Problem Definition

Graph Neural Networks operate under the assumption that **node features are clean and complete.**
In practice, data **frequently contains noise, missing values, and inaccuracies.**
Social, biological, and financial graphs are often incomplete, inconsistent, and unreliable.
The research addresses this gap by designing a model that can:

- Estimate the reliability of node features
- Learn robust representations under noise

Challenges

Incomplete or Noisy Node Features

Real-world graphs often contain missing, outdated, or incorrect attributes.
Most models treat all features equally, amplifying noise and reducing accuracy.

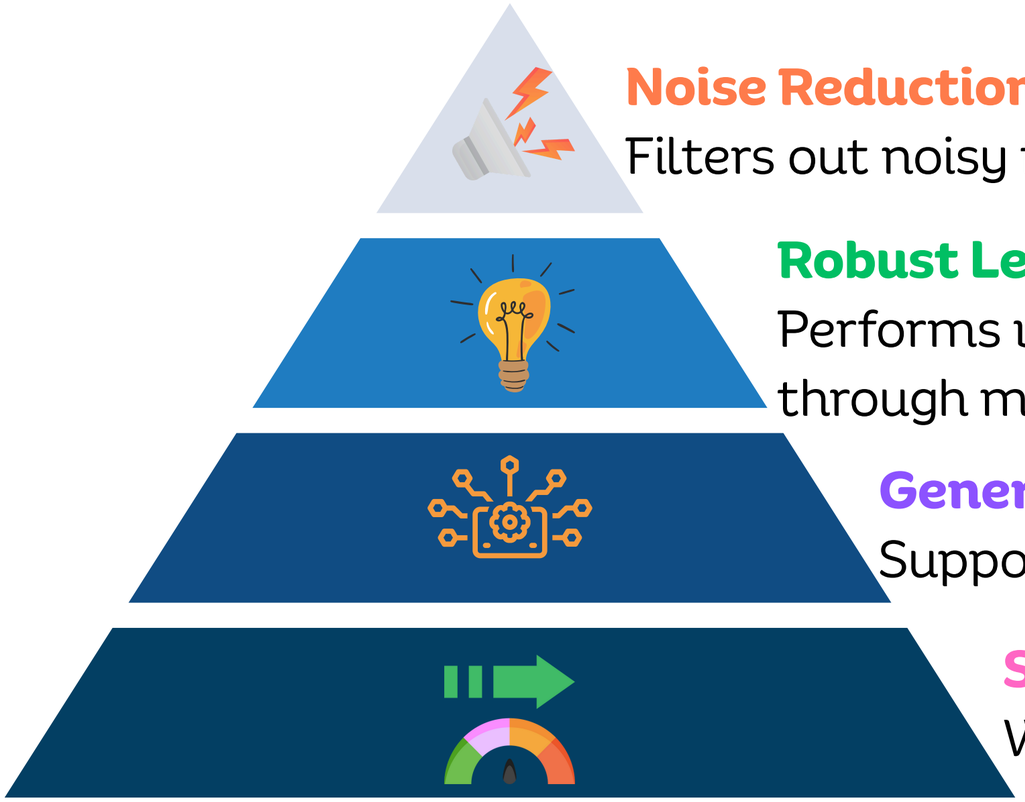
Adapting to Dynamic Graphs

Many GNNs assume a fixed structure.
They struggle to adapt to evolving graphs or generalise to unseen nodes, limiting their use in real applications.

BRAUDE
College of Engineering, karmiel

MQE and Its Impact on Graph Learning

Multi-hop Feature Quality Estimation (MQE) refers to a method for assessing the reliability of features in graph data through multi-hop propagation.
By evaluating feature quality across multiple layers, MQE enhances the robustness of machine learning models operating on noisy or incomplete networks.



Noise Reduction

Filters out noisy features through multi-hop analysis

Robust Learning

Performs well on incomplete or corrupted graph data features through multi-hop analysis.

Generalization

Supports transferable learning across different graph domains.

Scalable Inference

Works efficiently on large, real-world graphs

MQE

MQE Step-by-Step Workflow

Preproccessing
Loading Graph and preprocessing

Graph structure augmentation
Use k-Nearest Neighbors to connect similar nodes

Multi-hop Propagation
Aggregate information across neighborhood layers

Quality Estimation
Measure feature stability using variance across hops

Preprocessing

Prior to feature quality evaluation, the graph structure, node features, and neighborhood scope for information aggregation must be established.

1

Graph (G = V, E):
A data structure consisting of a set of nodes (V) and edges (E) representing the relationships between nodes.

2

Feature Matrix
A matrix where each node is represented by a vector of features describing its properties.

3

Propagation Depth (L)
The number of neighborhood hops considered during feature aggregation.

BRAUDE
College of Engineering, karmiel

Graph Structure Augmentation

Motivation for Graph Augmentation

Real-world graphs are often incomplete or misleading connections may be missing, weak, or outdated. Before we propagate information, we want to strengthen the structure so that feature aggregation is based on meaningful relationships.

Graph Augmentation Using k-Nearest Neighbors

The k-Nearest Neighbors (kNN) algorithm is applied to connect nodes that are similar in feature space, regardless of their original connectivity.

For each node

Find the k most similar nodes based on features.

Add edges between them

Importance of Graph Augmentation

- Combines structural and semantic connections
- Fixes missing links and improves neighborhood quality
- Prepares the graph for more reliable multi-hop propagation

BRAUDE
College of Engineering, karmiel

Multi-Hop Propagation

Every node learns not just from itself but from its neighborhood.

Multi-Hop Propagation

Each node aggregates information from its L-hop neighborhood, including both immediate and higher-order neighbors.

Importance of Multi-Hop Propagation

- Captures both local and global structural information
- Reinforces connections between related nodes
- Enhances learning in the presence of incomplete or missing features

BENEFITS

MQE doesn't just propagate blindly. It tracks feature consistency across hops to learn which information to trust and which to ignore.

Multi-hop propagation expands context, but noise from a single neighbor can spread and affect many nodes.

Feature Quality Estimation

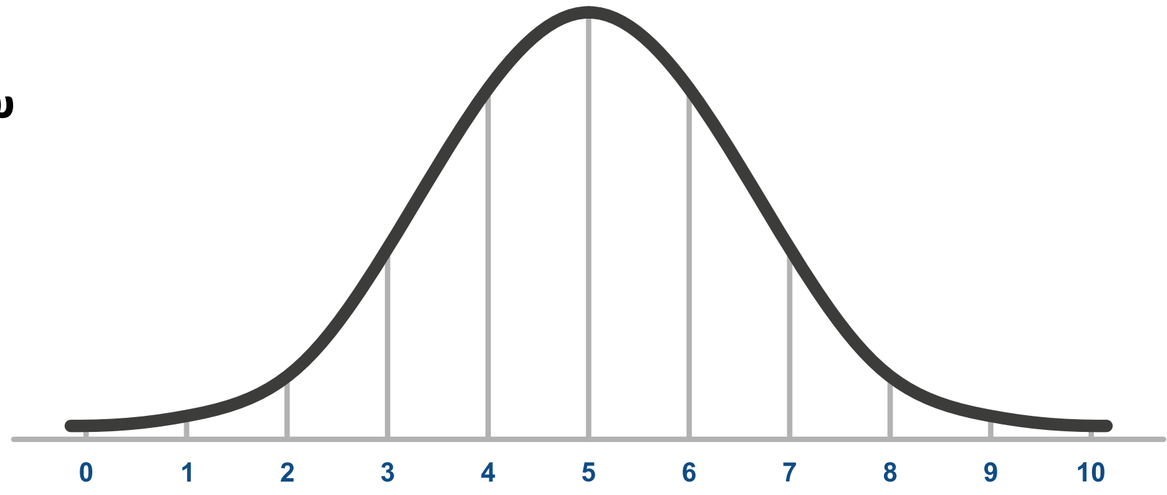
After multi-hop propagation**, MQE evaluates the reliability of each feature by estimating its trustworthiness.**

The core idea

Each propagated feature is modeled as a **Gaussian distribution:**

- Mean (μ) → what the feature value probably is
- Variance (σ^2) → how stable or noisy it seems

This allows the model to reason not just about the value but about **how confident it should be** in using it.



Assessing Feature Stability

If a feature stays **stable across hops**, it is variance is low → It's **reliable**

If a feature **fluctuates a lot**, variance is high → it's **noisy and risky**

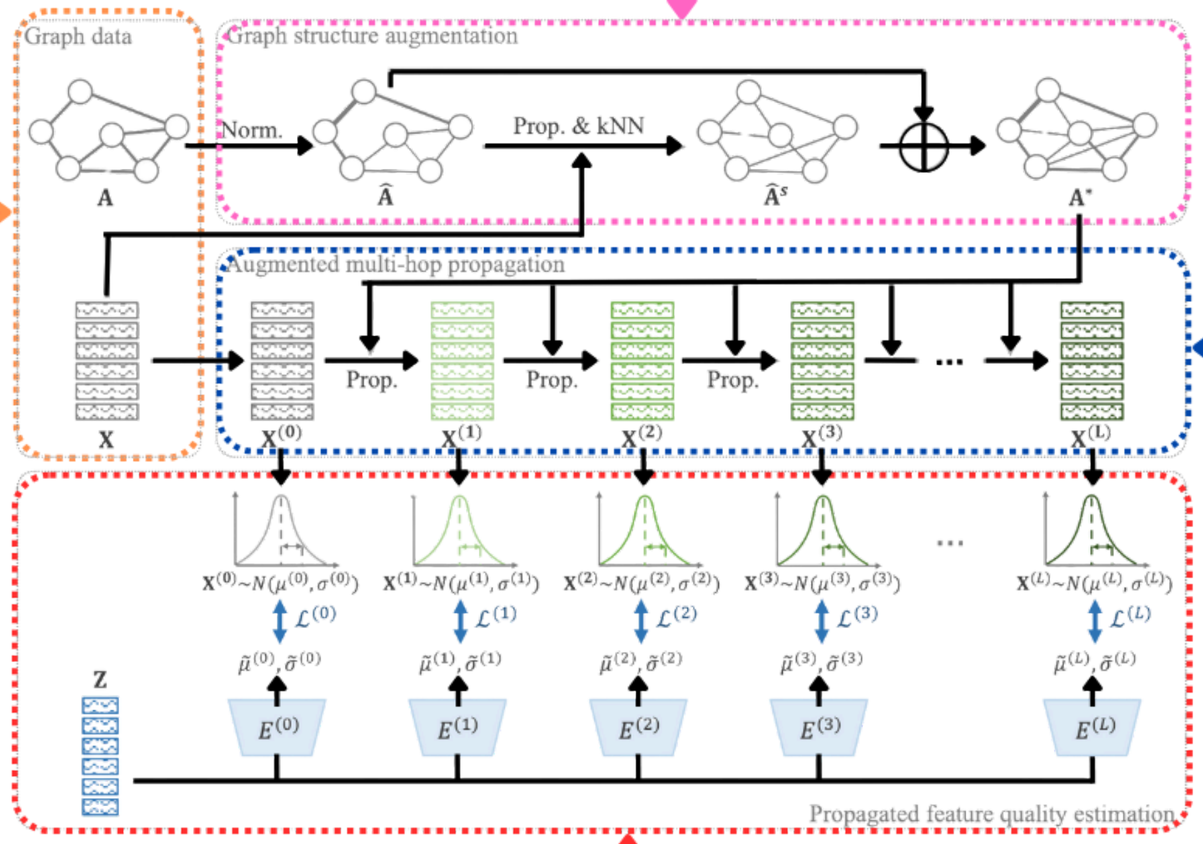
BRAUDE
College of Engineering, karmiel

Normalize an adjacency matrix and use **k-Nearest Neighbors** (kNN) to find similar data points.

The kNN algorithm then identifies the 'k' closest neighbours based on distance, aiding tasks **like classification and clustering.**

Adjencay Matrix (A)

Feature Matrix (X)



Perform multi-hop propagation on the augmented graph A*: Use the **augmented adjacency matrix A*** and propagate features for L hops:
$X^{(0)}=X$, $X^{(\ell)}=A \star X^{(\ell-1)}$, $\ell=1,2,\ldots,L$
This gives multi-hop features that are propagated with respect to the augmented graph structure.

MQE evaluate the quality of the propagated features, Checking whether the standard deviation is high or low based on this information, **MQE determines** whether it can **predict** the **features of a specific node** or whether it is **unable to do so (due to high standard deviation)**

BRAUDE
College of Engineering, karmiel

Making MQE Scalable: Inductive Power with GraphSAGE

Limitations of the Original MQE

- Requires access to the **entire graph** during training and inference --> Can't handle **new or unseen nodes**
- Struggles with **dynamic graphs** where nodes and edges change over time.
- **Runtime** on big graphs is challenging because they are large, dynamic, and constantly evolving.

Why GraphSAGE

- GraphSAGE (**Sample and Aggregate**) enables **inductive** representation learning
- Instead of processing the full graph, it **samples a fixed-size neighborhood**
- Works efficiently on **large, evolving graphs** → and generalizes to **unseen nodes**

MQE is powerful but not **scalable** on its own.
GraphSAGE makes MQE **practical, efficient, and future-proof.**

BRAUDE College of Engineering, karmiel

Embedding with GraphSAGE

Scalable, inductive, and neighbor-aware node representation learning.

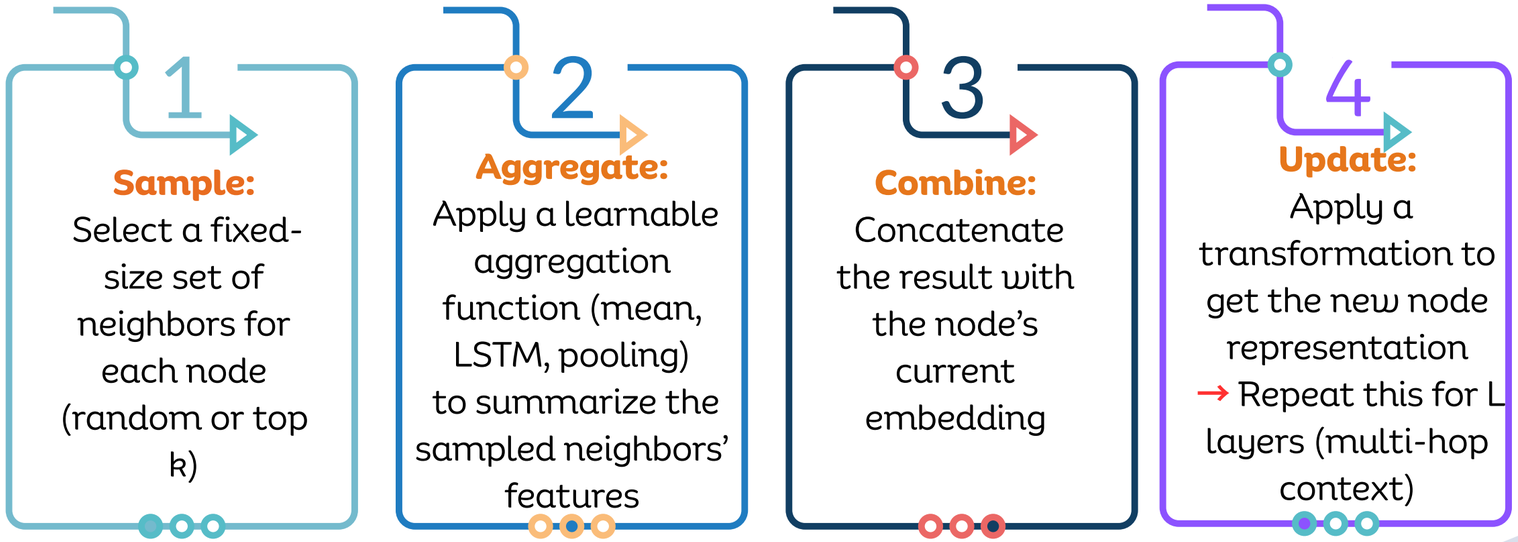
GraphSAGE as an Alternative to Full Propagation

Traditional GNNs require the entire graph to compute embeddings which breaks when graphs are huge, changing, or include unseen nodes.

GraphSAGE solves this by:

- 📌 **Sampling a small, fixed number of neighbors** at each layer
- 📌 Learning how to **aggregate and combine** their information efficiently
- 📌 Operating in an **inductive way** meaning it works even on **nodes not seen during training**
- 📌 Ideal for dynamic, large-scale, and real-world graphs.

How Does It Work?



BRAUDE College of Engineering, karmiel

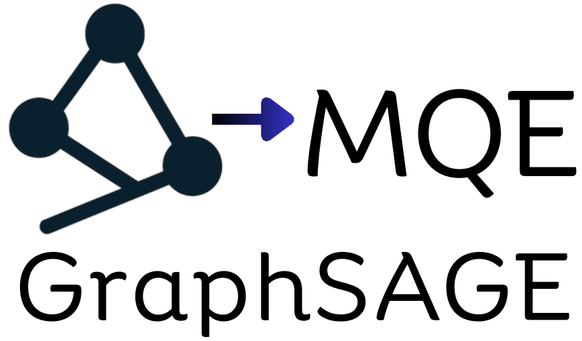
Integrating GraphSAGE into MQE

Key Changes with GraphSAGE Integration

Instead of full-matrix propagation, GraphSAGE applies **neighbor sampling to aggregate multi-hop features.**

Each node collects information through:

- Sampling a fixed number of neighbors
- Applying learnable aggregation functions
- Performing layer-wise updates



Core Components Preserved After GraphSAGE Integration

- **Aggregation still spans** up to L hops using sampled neighborhoods
- Features at each hop are **modeled as Gaussian distributions**
- Mean and variance are **estimated to assess feature quality**

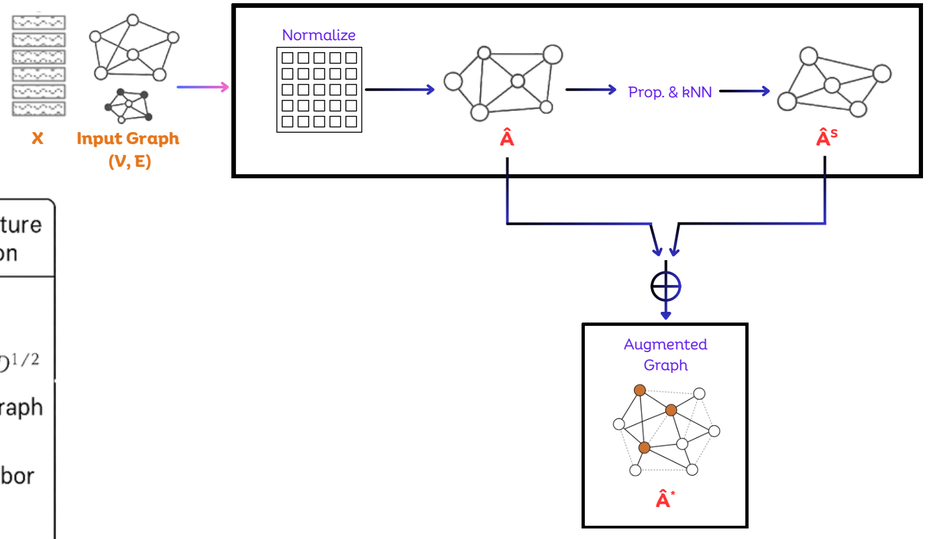
Benefits of Integrating GraphSAGE into MQE

- **Inductive learning:** Supports generalization to unseen nodes during inference
- **Scalability:** Efficient for large-scale and dynamic graphs
- **Robustness:** Continues to filter out unreliable features through quality estimation

GraphSAGE gives MQE the scalability it lacked,
while MQE gives GraphSAGE the uncertainty-awareness it never had.

BRAUDE
College of Engineering, karmiel

Graph Structure Augmentation



1. Input Initialization

- Graph G (V, E)
- Feature matrix $X\,(n \times f)$
- Set $h^0) = X$
- Param: hops L

2. Graph Structure Augmentation

- Normalize adjacency: $\ddot{A} = D^{-1/2}\,A^{\cdot}\,D^{1/2}$
- Build k-NN graph on X
- Use for neighbor sampling in GraphSAGE

Repeat for $l = 1$ to L

3. GraphSAGE Feature Propaggation

Repeat for $l = 1$ to L:
For each node i

1. Aggregate: $h^{i)^n} =$ AGG(h^{n-1}))
2. Concat with $h^{(l-1)}$
3. Transform: $h^{(l)} = \sigma(\mathrm{W}^{(r}$)

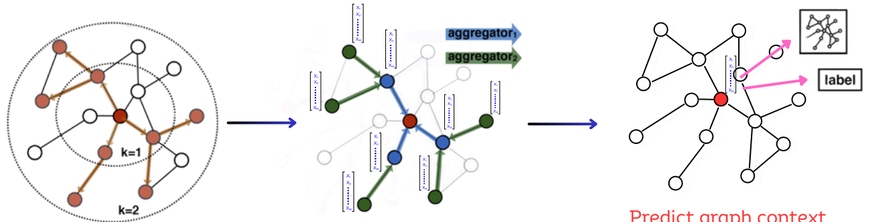
Output: $h^{(i)} \ldots h(l)$

4. Feature Quality Estimation

- Predict: $\mu(l)$, $\sigma^{(l)}$ from latent z
- Model: $h(l) \sim N$
- Loss: Gaussian NLL
- Suppress high-σ features

Final Output: Meta-repr $Z\,(n \times z)$

GraphSAGE-based multi-hop propagation

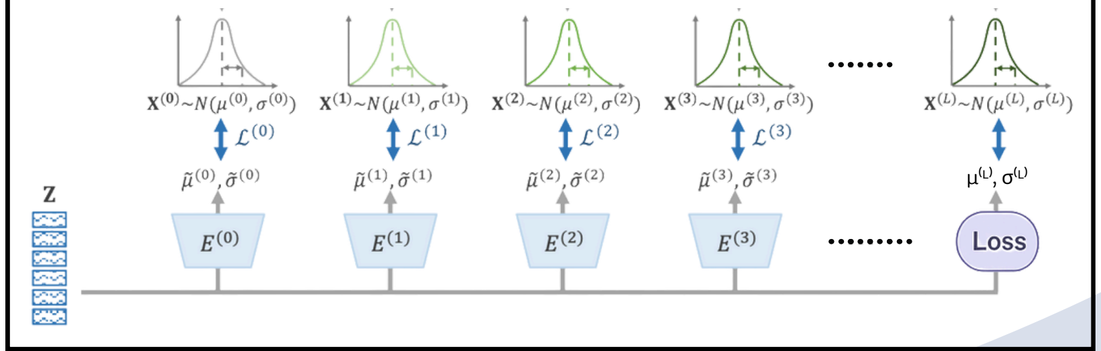


Sample Neighborhood

Aggregate feature information from neighbors

Predict graph context and label using aggregated information

Propagated feature quality estimation



BRAUDE College of Engineering, karmiel

Evaluation Plan: Measuring the Value of MQE + GraphSAGE

Objectives

- Measure accuracy, robustness, and scalability
- Test performance under increasing noise levels
- Compare original MQE vs MQE + GraphSAGE vs baselines

Datasets

- Cora, CiteSeer, PubMed - citation networks
- Amazon Computers, Amazon Photos - product co-purchase graphs
- Simulate noise
- Gaussian noise
- Uniform noise
- Feature dropout / corruption

Evaluation Metrics

Metric	Purpose
Accuracy	Overall performance on classification
NLL (Negative Log-Likelihood)	Quality of feature estimation
Robustness Score	Performance drop under increasing noise
Runtime (sec/epoch)	Efficiency on large graphs
Generalization Score	How well it handles unseen nodes

Goal: Show that MQE + GraphSAGE achieves both robustness under noise and scalability in real-world settings outperforming other models.

Testing Plan:
Comparing Original MQE vs. MQE + GraphSAGE

Metric	Description	Expected Superior Variant
Classification Accuracy	Predictive quality on clean data	Both
Robustness to Noise	Performance degradation under feature noise	MQE (Standalone)
Runtime	Training and inference time	MQE + GraphSAGE
Scalability	Efficiency on large graphs	MQE + GraphSAGE
Inductive Generalization	Handling unseen nodes	MQE + GraphSAGE

! By combining MQE with GraphSAGE, we preserve MQE's feature quality estimation while gaining **scalability, inductive capability, and efficiency for real-world deployment.**

BRAUDE
College of Engineering, karmiel

THANK YOU!