5-2022

Learning semantically rich network-based multi-modal mobile user interface embeddings

Meng Kiat Gary ANG
Singapore Management University, gary.ang.2019@phdcs.smu.edu.sg

Ee-peng LIM
Singapore Management University, eplim@smu.edu.sg

Learning Semantically Rich Network-Based Multi-Modal Mobile User Interface Embeddings

GARY ANG*, Singapore Management University
EE-PENG LIM, Singapore Management University

Semantically rich information from multiple modalities - text, code, images, categorical and numerical data - co-exist in the user interface (UI) design of mobile applications. Moreover, each UI design is composed of inter-linked UI entities which support different functions of an application, e.g., a UI screen comprising a UI taskbar, a menu and multiple button elements. Existing UI representation learning methods unfortunately are not designed to capture multi-modal and linkage structure between UI entities. To support effective search and recommendation applications over mobile UIs, we need UI representations that integrate latent semantics present in both multi-modal information and linkages between UI entities. In this article, we present a novel self-supervised model - Multi-modal Attention-based Attributed Network Embedding (MAAN) model. MAAN is designed to capture structural network information present within the linkages between UI entities, as well as multi-modal attributes of the UI entity nodes. Based on the variational autoencoder framework, MAAN learns semantically rich UI embeddings in a self-supervised manner by reconstructing the attributes of UI entities and the linkages between them. The generated embeddings can be applied to a variety of downstream tasks: predicting UI elements associated with UI screens, inferring missing UI screen and element attributes, predicting UI user ratings, and retrieving UIs. Extensive experiments, including user evaluations, conducted on datasets from RICO, a rich real-world mobile UI repository, demonstrate that MAAN out-performs other state-of-the-art models. The number of linkages between UI entities can provide further information on the role of different UI entities in UI designs. However, MAAN does not capture edge attributes. To extend and generalize MAAN to learn even richer UI embeddings, we further propose EMAAN to capture edge attributes. We conduct additional extensive experiments on EMAAN, which show that it improves the performance of MAAN and similarly out-performs state-of-the-art models.

CCS Concepts: • **Computing methodologies** → **Neural networks**; • **Human-centered computing** → **User interface management systems**; • **Information systems** → **Multimedia and multimodal retrieval**.

Additional Key Words and Phrases: network embedding, mobile application user interface, unsupervised retrieval, self-supervised learning, multi-modal, user interface design

1 INTRODUCTION

The increasing maturity of mobile application design/development practices means that a mobile UI designer/developer can easily refer to existing UI repositories as part of the development process. Such repositories include multi-modal information - network structures consisting of UI screens and UI elements of mobile applications with one or more links between them, visual information from UI screen images, codes associated with UI elements, textual description of the associated applications, categorical application genres and numerical application ratings. The

*Contact author

Authors' addresses: Gary Ang, Singapore Management University, Singapore, gary.ang.2019@phdcs.smu.edu.sg; Ee-Peng Lim, Singapore Management University, Singapore, eplim@smu.edu.sg.

2160-6455/2022/5-ART $15.00
https://doi.org/10.1145/3533856

RICO dataset [6, 28] is one such repository. It includes data from more than 9,000 Android applications. Such repositories are extremely valuable when we search relevant UI objects for design reference or for replacing some existing UI objects as part of the UI development process. However, manually sifting through UI objects in large repositories is not feasible. A naive retrieval system based on keywords or genre would not be adequate as such features are sparse and do not capture rich semantics in UI interface designs.

Latent representations, also known as embeddings, can be developed to better capture underlying similarities. Embeddings capture semantics of data by projecting semantically similar data close together in the embedding space. However, most approaches for generating embeddings only capture information from a single modality. Existing multimedia/multi-modal approaches to represent UI screen images also do not capture the non-Euclidean nature of network structures linking UI interface components together. Hence, a *network-based embedding approach that captures multi-modal information*, i.e. an approach that generates embeddings that captures both network structure and information from multiple modalities is needed. The state-of-the-art attributed network embedding models nevertheless generate embeddings that capture structural network information, but *not information from multiple modalities and node types.*

On the left of Figure 1, we show an overview of the UI dataset. Two types of nodes, i.e. UI screens and UI elements exist in the data. The attributes of UI screen and element nodes, i.e. the *node attributes*, comprise features from multiple modalities. In this paper, the *multi-modal features* of a UI screen node include its image, description, and genre. Similarly, a UI element node's features include its class name, and component type. Edges are formed between UI screen nodes and UI element nodes whenever a UI element is part of a UI screen, e.g., an image element that is part of a gallery screen. As a UI screen can be linked to a UI element multiple times, e.g., a gallery screen with multiple image elements, the number of such links can be treated as *edge attributes*. Therefore, in this paper, edge attributes, or features of the edges, refer to the number of such links for each of these edges. Figure 2 provides an example of a UI screen and its constituent UI elements in its UI view code.

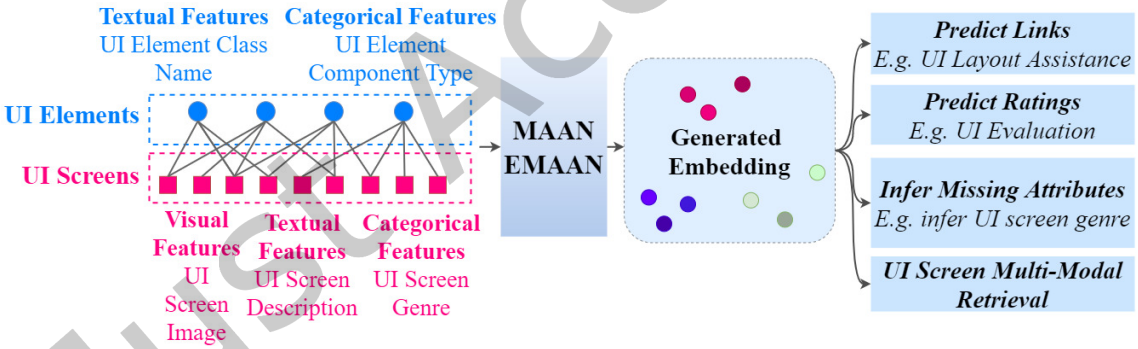

Fig. 1. UI dataset in a network embedding framework.

The middle of Figure 1 shows our proposed multi-modal network embedding framework. In this framework, we design two multi-modal attention-based attributed network embedding models, MAAN and EMAAN, that can be used to learn the embeddings representations of UI screens and UI elements in a self-supervised manner. Specifically, MAAN captures the structural network information present within the linkages between UI screens and elements, as well as multi-modal attributes of the UI screens and elements to generate semantically rich embeddings of UI screens and elements. EMAAN further generalizes MAAN to capture edge attributes, e.g., the number of occurrences of linkages between UI screens and elements, or the depth of the UI element in the UI view code of the UI screen as shown in Figure 2.

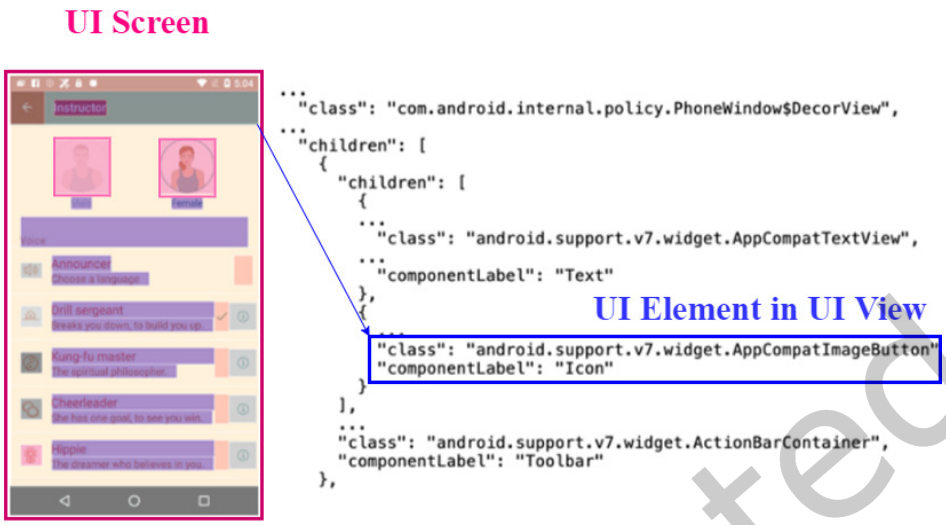

Fig. 2. Example of a UI screen image and its constituent UI elements in the UI view of the screen. The UI screen's genre is *'Health & Fitness'*, and it is associated with the description *'Getting fit has never been so easy – or so much fun! Seven workouts are based on scientific studies to give you the maximum benefits of exercise with only 7-minutes a day...'*. The boxed UI element has the class name *android.support.v7.widget.AppCompatImageButton*, and is of component type *Icon*. The UI screen is associated with multiple elements of the same type, e.g., the two *image* elements highlighted in pink boxes.

While the UI networks in this paper are defined based on the RICO dataset, this network definition can be generalized and potentially applied to other UI datasets with different information, e.g., instead of the number of occurrences of linkages between a pair of UI screen and UI element nodes, the edge attributes may be defined by the frequency of users interacting with UI elements in the UI screen; and multi-modal node attributes can represent information from other modalities, e.g., audio modality (e.g., sound associated with the user touching a UI element), gesture modality (e.g., coordinates and velocity associated with the user interacting with a UI element) [8, 26].

Both the MAAN and EMAAN embeddings can be used in several downstream tasks, e.g., predicting links between UI screens and elements for UI layout assistance, rating predictions, missing attribute inference, i.e. predicting missing UI screen or element node attributes, and UI screen retrieval.

There have been other works that take a network-based approach or used latent representations/embeddings for tasks relating to user interactions [7, 12, 25, 44, 45], but *the use of network embedding models for such mobile UI development related tasks has not been explored so far.*

MAAN and EMAAN aim to incorporate several new requirements: 1) *capture correlations between node attributes from different modalities*; 2) *ensure no one modality dominates the generated embedding*; 3) *capture semantics present in multi-modal node attributes together with edge attributes*; 4) *effectively balance between the increased number of training objectives.* Capturing correlations, or affinities, between information from multiple modalities (e.g., between visual styles and application genres, or visual structure and code classes) plays a critical role in generating informative embeddings. The dimensions of modalities could differ substantially (e.g., number of genres vs. number of words) and simply encoding them together could cause the generated embedding to focus only on one or two modes. The increased number of modalities also means additional training objectives, and a need to balance between these objectives that could diverge. Adopting a network-based approach for learning UI

embeddings is useful as the linkages/edges between UI entities contain valuable information about the context of the UI entity's role in the UI. In addition to the linkages/edges, being able to capture the attributes of these linkages/edges can further inform the encoding process, e.g., the number of occurrences of linkages between UI screens and elements can indicate the relative role and importance of different neighboring UI entities.

Both MAAN and EMAAN comprise a *novel graph variational autoencoder (GVAE)-based model to generate multi-modal embeddings of UI screen and element nodes.* Specifically, the GVAE in MAAN (and EMAAN) *jointly embeds information from multiple modalities*; uses an *attention mechanism to discover correlations between different modalities and weight their contribution*; and *improves existing loss functions used for graph variational autoencoders* to learn more meaningful embeddings. *EMAAN further extends MAAN by integrating linear and non-linear mechanisms within MAAN in a novel manner to capture edge attributes when encoding multi-modal and structural network information.*

The main contributions of our article are as follows:

- To our knowledge, this is the first work to present a multi-modal network embedding framework that combines attention mechanisms with the variational autoencoding framework to generate embeddings for link prediction, attribute inference, regression and retrieval tasks relating to mobile UIs.
- We propose a two-stage encoding process to ensure no one modality dominates the generated embedding.
- We use the attention mechanism to capture relationships between information from different modalities, which is different from past works [17] that use attribute-to-attribute matrices.
- The use of the Maximum Mean Discrepancy (MMD) [52] term to more effectively balance between multiple training objectives has not featured in existing GVAEs.
- We show that MAAN consistently out-performs other state-of-the-art models on two datasets across an extensive set of prediction tasks. User evaluations on retrieval results show that MAAN out-performs the other models by as much as five times.
- We further extend MAAN as EMAAN that takes edge attributes into account during the encoding process and demonstrate that the incorporation of edge attributes into the multi-modal network embedding framework can lead to improved performance on downstream tasks, e.g., prediction of UI screen ratings, inference of missing UI features.
- To further validate our findings, we conduct additional experiments with both MAAN and EMAAN on a larger dataset covering a different set of mobile applications. These additional experiments, which include another set of user evaluations on retrieval results, further demonstrate the advantages of capturing multi-modal information, structural network information, as well as edge attributes, as we observe similar improvements in performance on downstream tasks such as prediction of UI screen ratings and inference of missing UI features.

2 MOTIVATING SCENARIOS

Before we describe the approach proposed for MAAN/EMAAN, we describe these motivating application scenarios to illustrate how one can utilize the UI embeddings generated by MAAN/EMAAN. Figure 3 provides an overview of these motivating scenarios.

2.1 UI Layout Assistance

Consider the mobile UI design scenario where a designer/developer has a partially completed layout with a few UI elements selected from a UI repository, and requires suggestions on potential UI screen designs or other UI elements to complete the mobile UI design. The embeddings of the selected UI elements generated by MAAN/EMAAN can be combined to represent the embedding of the partially completed UI screen, and used to predict the probability of linkages between the partially completed UI screen and other UI elements

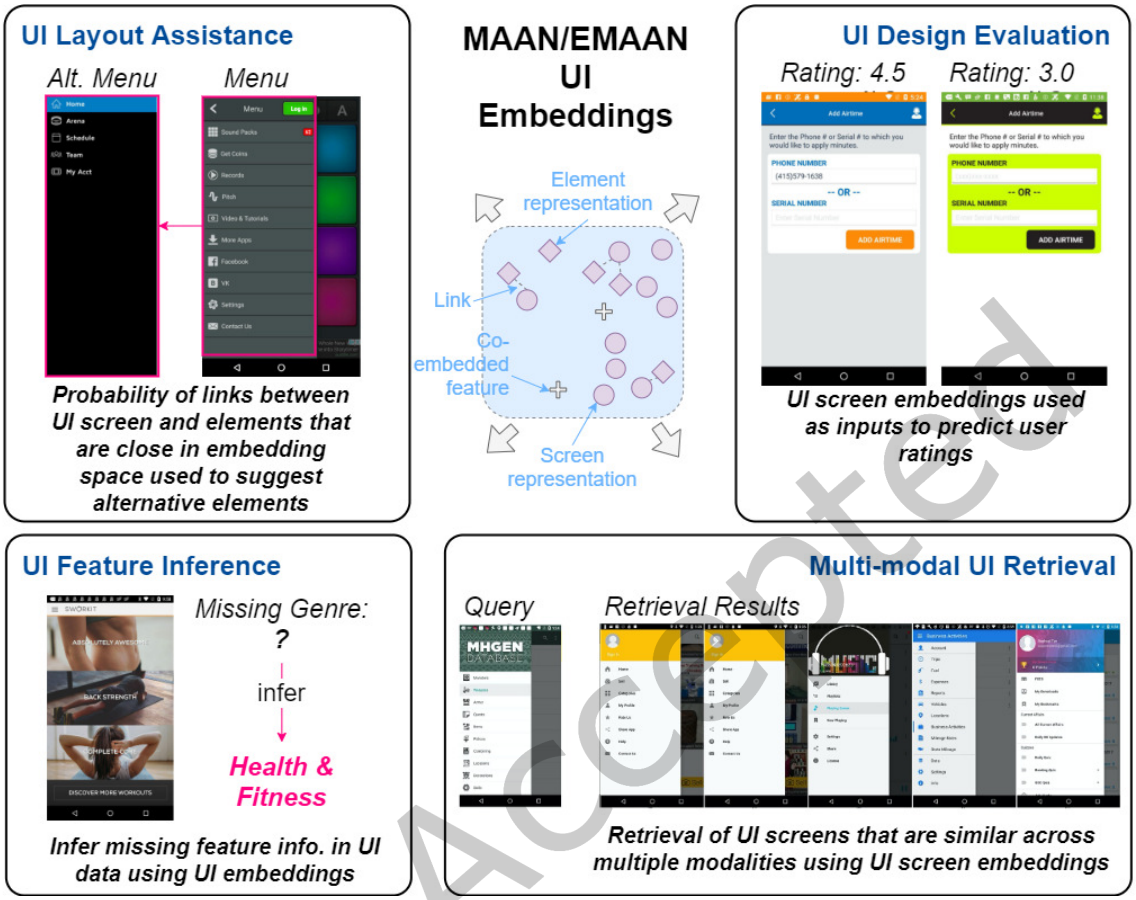

Fig. 3. Overview of motivating scenarios for UI embeddings generated by MAAN/EMAAN.

in the UI repository. The links with the highest probabilities represent links to the semantically closest UI elements in the embedding space that can be presented as suggestions to the designer/developer. Alternatively, the designer/developer may have assembled a complete set of UI elements for a mobile UI layout, but would like to refine it, and consider suggestions on alternative UI elements. The embeddings of the assembled UI elements generated by MAAN/EMAAN can be combined to represent the embedding of the completed UI screen and predict the probability of linkages between the completed UI screen and other UI elements in the UI repository. Links with the highest probabilities represent links to the semantically closest UI elements in the embedding space that can be presented as alternatives to the designer/developer.

2.2 UI Design Evaluation

This is an application scenario where a designer/developer has developed a new mobile UI and would like an initial evaluation of the potential success of the mobile UI. The multi-modal and structural network information of the new mobile UI could be used as inputs to MAAN/EMAAN to generate its embedding. The generated

embedding can then be fed as inputs into a regression model trained to predict its UI rating as feedback to the designer/developer.

2.3 UI Feature Inference

An entity (e.g., company or research lab) has been collecting UI information over time for a mobile UI repository. However, due to errors during the collection process, certain feature information relating to a number of UIs are missing. To impute the missing UI information in this mobile UI repository, MAAN/EMAAN can be used to infer missing features of UI screens and elements based on affinities between UI entity nodes and features. A pre-trained MAAN/EMAAN (e.g., trained on RICO) or MAAN/EMAAN trained in a self-supervised manner on this mobile UI repository could be used to infer the missing features.

2.4 Multi-modal UI Retrieval

A designer/developer is conducting research into UIs for a potential mobile application and has come across a few promising UIs in a mobile UI repository, and would like to retrieve from the repository more UIs that are similar. The designer/developer is not only interested in UIs that are visually similar, but would like to retrieve UIs that are similar across multiple modalities, e.g., genre, description, UI view classes used, and that are also similar in their structures. By capturing information from multiple modalities and structural network information in the same embedding space, MAAN/EMAAN generates embeddings that can be used for such multi-modal UI retrieval.

3 RELATED WORK

Key related works in the areas of UI retrieval, evaluation and design, and network embedding models are outlined in this section.

3.1 UI retrieval, evaluation and design

[2, 18, 25, 36, 48] are examples of recent work that also generate representations or embeddings of UIs. Such works however only use information from one or two modalities, and also do not utilize the structural network information present in the linkages between UI screens and elements. For example, Swire [18] is a system that retrieves UI screen images based on UI sketches. Both UI sketches and UI screenshots are represented as image embeddings for visual similarity comparison. Swire however only captures information from a single modality, and does not capture the structural network information present in the linkages between UI screens and elements. [48] converts UIs into sequences to support retrieval applications. This approach only captures structural network information of UI objects but not multi-modal information. Screen2Vec [25], a recent work that was published after MAAN [1], adopts an approach that is similar to MAAN. Screen2Vec generates representations of UI screens and components that capture the multi-modal information of UIs and UI layouts. However, it does not capture the structural network information present within the linkages between UI screens and elements.

The comprehensive semantic representations of UIs with multi-modal and structural network information learnt by a pre-trained MAAN or EMAAN can also be used to complement or augment a range of other UI-related systems, such as UI object detection and annotation systems [5, 16, 46, 51]. MAAN/EMAAN can also complement systems that assist UI designers [4, 24, 32]. For example, DesignScape [32] provides suggestions for designers to brainstorm and refine their graphic design. However, DesignScape was designed for graphic design layouts, and would not be suitable for UI designs which involve multi-modal and structural network information. GUIComp [24] developed a design assistant that uses measures of visual complexity [35] to help assess visual complexity of UI screens. This work focuses on the ranking of UI screens based on heuristics instead of their embedding

representations. MAAN/EMAAN can complement systems such as GUIComp in UI design evaluations with a data-driven approach that captures multi-modal and structural network information.

3.2 Network embedding models

As UI screens and elements are inter-connected in a large network, we thus review network embedding models.

3.2.1 Homogeneous Networks. DeepWalk [34] and Node2Vec [13] extend the idea of word embeddings to networks by treating paths as sentences and nodes as words. They however do not utilize the attributes of nodes when learning network embeddings. Graph neural networks (GNN) learn node embeddings by propagating the attributes of nodes repeatedly or over multiple neural network layers via a message passing framework [11]. Graph Convolutional Networks (GCN) [23] aggregate features of neighboring nodes and normalize the aggregated representations by the node degrees. GraphSAGE [14] further considers mean, LSTM or pooling aggregation methods. While GCN uses the full neighborhood, GraphSAGE samples a fixed number of neighbors. In the Graph Attention Network (GAT) [41], different nodes in the neighborhood are assigned different importances during aggregation. Messages passed between each layer in most GNNs go through non-linear layers such as rectified linear activation units (ReLU). [47] finds that similar performance can be achieved even without such layers. The Graph Variational Autoencoder (GVAE) [22] applies the variational autoencoder (VAE) [21] framework to learn the node embeddings of homogeneous networks. The Co-Embedding Attributed Network (CAN) [30] uses two VAE channels to jointly encode and decode the adjacency matrix and a single feature matrix to jointly learn the representations of both nodes and attributes. Semi-supervised Co-embedding Attributed Network (SCAN) [31] extends CAN to co-embed both attributes and nodes of partially labelled networks.

3.2.2 Heterogeneous Networks. Such models have also been applied to heterogeneous networks (networks with multiple node or edge-types). Relational Graph Convolutional Networks [37] and Graph Convolutional Matrix Completion [39] use multiple GCNs to encode embeddings of multiple adjacency matrices, one for each edge type, before aggregating them. Neural Graph Collaborative Filtering [43] and LightGCN [15] encode embeddings for different number of hops before aggregating them. [19] captures indirect proximity between the same node types in bipartite networks. Multinomial VAE [27] is a VAE-based approach that generates embeddings by using a multinomial distribution instead of the Bernoulli distribution used in GVAE. Heterogeneous Graph Attention Network [42] and General Attributed Multiplex Heterogeneous Network [3] use multiple GNN-based layers to encode networks formed from different metapaths [9] before using an attention mechanism to aggregate the embeddings.

While the related works outlined in this section only use information from a single modality, the proposed MAAN and EMAAN models in this paper use a VAE-based approach to encode information from multiple modalities. A two step encoding process is used so that no one modality dominates the generated embedding. Further, we introduce an attention mechanism to capture the correlations between different modalities and allow the underlying importance of each modality to be self-discovered. Instead of just applying the generated embeddings to common link prediction and attribute inference tasks, we use the generated embeddings to predict continuous labels (application ratings) and for multi-modal retrieval of the UI screens.

4 MULTI-MODAL ATTENTION-BASED ATTRIBUTED NETWORK EMBEDDING

In this section, we give a detailed description of our two proposed multi-modal attention-based attributed network embedding models, MAAN and EMAAN. We will first cover MAAN as it forms the base model for EMAAN.

To cope with both multi-modal features and the heterogeneity of structural network information in UI designs, we propose a new embedding model known as **Multi-modal Attention-based Attributed Network Embedding (MAAN).** It represents UI designs by a bipartite graph $G = (V, E, X)$ constructed from a UI design

Table 1. Summary of Key Notations.

Symbol	Description
U	Set of UI screen nodes
W	Set of UI element nodes
V	$U \cup W$
P	Set of UI screen node attributes
Q	Set of UI element node attributes
F_p	Feature dimension of UI screen node attribute p, $p \in P$
F_q	Feature dimension of UI element node attribute q, $q \in Q$
$X_{U,p}/X_{W,q}$	Features of attribute p/q of all nodes in U/W (That is, $X_{U,p}$ is a $\|U\| \times F_p$ matrix, and $X_{W,q}$ is a $\|W\| \times F_q$ matrix)
E	Set of edges between UI screen nodes and UI elements ($\subseteq U \times W$)
F_e	Feature dimension of edge $e \in E$
X_e	Feature of attribute of edge $e \in E$ (That is, X_e is a $\|E\| \times F_e$ matrix)
$X'_{V,1}, .., X'_{V,r} \in \mathbb{R}^{\|V\| \times d'}$	Hidden multi-modal representations of nodes after GAT modules where $r \in \{p, q\}$
$X''_V \in \mathbb{R}^{\|V\| \times d'}$	Hidden node representations after multi-modal attention fusion
$Z \in \mathbb{R}^{\|V\| \times d}$	Final latent representation/embedding of nodes
$H_r \in \mathbb{R}^{F_r \times d'}$	Hidden representation of feature from the r$^{\text{th}}$ modality where $r \in \{p, q\}$
$Z_r \in \mathbb{R}^{F_r \times d}$	Final latent representation/embedding of feature from the r$^{\text{th}}$ modality where $r \in \{p, q\}$

dataset (e.g., RICO), where V comprises two disjoint sets of nodes U and W. U represents UI screen nodes and W represents UI element nodes. Edges only exist between different node-types, i.e. $E \subseteq U \times W$. e_{ij} represents the edge between $v_{U,i}$ and $v_{W,j}$. X comprises multi-modal features, $|P|$ of these attributes are for UI screens and $|Q|$ of them are for UI elements, denoted by $\{X_{U,p}|p \in P\}$, and $\{X_{W,q}|q \in Q\}$ respectively. MAAN will jointly embed the nodes and their multi-modal features as low-dimensional vectors by learning the mapping functions $f : V \mapsto Z \in \mathbb{R}^{|V| \times d}$ for the nodes, and $g : X_{V,r} \mapsto Z_r \in \mathbb{R}^{F_r \times d}$ for features in each modality where $r \in \{p, q\}$ and $d \ll |V|$.

We first construct a $|V| \times |V|$ heterogeneous adjacency matrix A from the network edges E:

$$A = \begin{bmatrix} 0 & M \\ M^T & 0 \end{bmatrix}$$

where $M \in [0, 1]^{|U| \times |W|}$ is the UI screen to UI element adjacency matrix.

We also construct the multi-modal feature matrices $\{X_{U,p}|p \in P\}$, $\{X_{W,q}|q \in Q\}$. Each feature matrix $X_{V,r} \in \mathbb{R}^{|V| \times F_r}$ for modalities $r \in \{p, q\}$ of nodes $V \in \{U, W\}$ contains attribute information for all $|V|$ nodes. Where a feature does not apply to the node, e.g., feature p for a node v in W, the feature matrix $X_{U,p}$ will have the row corresponding to node v filled with zeros. MAAN uses one VAE channel to encode/decode the adjacency matrix; and one VAE channel to encode/decode every feature matrix. Hence, it requires $|P| + |Q| + 1$ VAE channels. VAE is chosen because of its expressiveness, generative ability and flexibility to be combined with other deep learning models. Figure 4 depicts the different channels in the MAAN architecture, which consists of a multi-modal attention-based GAE and several feature VAEs.

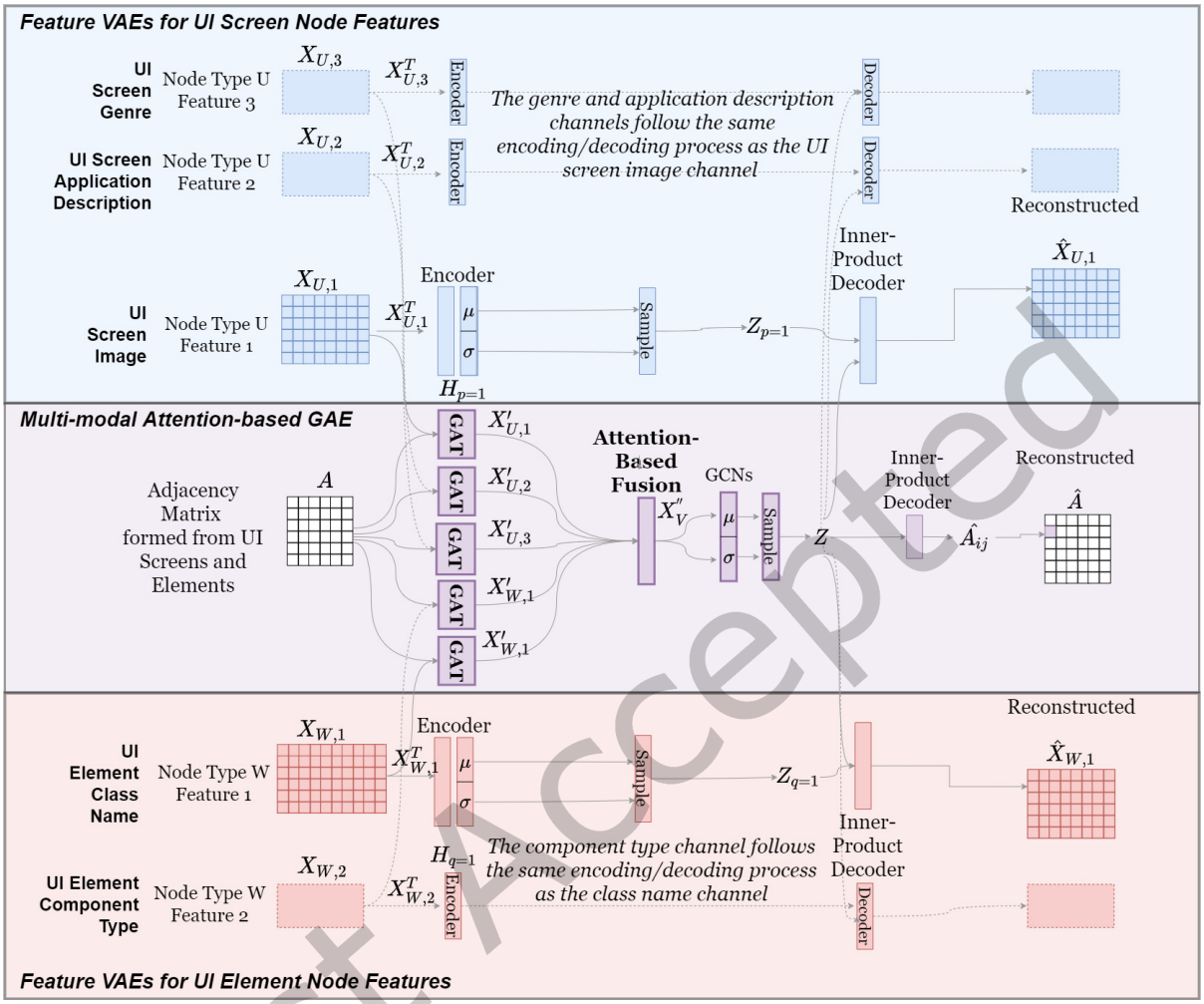

Fig. 4. MAAN Architecture for RICO with $|P| = 3$ VAE channels for UI screen nodes and $|Q| = 2$ VAE channels for UI element nodes.

4.1 Multi-modal Attention-based GAE

This module generates the node embeddings. Multiple GATs encode node features from different modalities, which are then fused with an attention mechanism. The fused node features are encoded with GCNs into Gaussian embeddings, and an inner product decoder used to reconstruct the adjacency matrix by sampling from the learned Gaussian embeddings.

The module first takes as inputs the adjacency matrix A and the node features $\{X_{U,p}|p \in P\}$, $\{X_{W,q}|q \in Q\}$. It uses $|P| + |Q|$ GAT modules to separately transform a node's features of different modalities to hidden representations. Encoding each modality with separate GAT modules before combining them avoids any modality dominating the final embedding, and allows the model to accommodate any number of modalities. For each layer in a GAT module, we first take as input the node feature of node i from U or W and apply a linear transformation:

$$s_{i,r}^{(l)} = W^{(l)} x_{i,r}^{(l)} \tag{1}$$

where $W^{(l)}$ is the learnable weight matrix of the l^{th} layer, and $x_{i,r}^{(l)}$ is either the original feature or the hidden representation of the node i from an earlier layer of some attribute p or q, i.e., $r = p$ or $r = q$ depending on node $i \in U$ or $i \in W$.

We then compute the pair-wise un-normalized attention score between node i and each of its neighbors, say node j. We concatenate the hidden representations of nodes i and j before taking a dot product of it with a learnable weight vector $a^{(l)}$. A LeakyReLU activation is then applied:

$$e_{ij}^{(l)} = LeakyReLU(a^{(l)T}(s_{i,r}^{(l)} || s_{j,r'}^{(l)})) \tag{2}$$

where $r' = q$ if $r = p$ or $r' = p$ if $r = q$, depending on node $i \in U$ or $i \in W$.

Softmax is then used to compute the attention scores to weight the hidden representations received by node i from its neighboring nodes $N(i)$:

$$\alpha_{ij}^{(l)} = \frac{exp(e_{ij}^{(l)})}{\sum_{k \in N(i)} exp(e_{ik}^{(l)})} \tag{3}$$

The final step in each GAT layer aggregates the hidden representations received by node i from its neighbors $N(i)$, weighted by the attention scores.

$$x_{i,r}^{(l+1)} = \sigma(\sum_{j \in N(i)} \alpha_{ij}^{(l)} s_{j,r}^{(l)}) \tag{4}$$

The steps outlined above are repeated for each layer in the GAT module with node features from each modality, resulting in each node having $|P| + |Q|$ separate sets of intermediate hidden representations of d' dimensions $\{X'_{U,p} = X_{U,p}^{(L)} | p \in P\}$, $\{X'_{W,q} = X_{W,q}^{(L)} | q \in Q\}$ after L layers , where $d' > d$. Setting the dimensions of intermediate node representations to be higher than the final latent representation of nodes allows for a more gradual compression of the embedding space.

Next, we use an attention mechanism to fuse representations from different modalities into a $|V| \times d'$ representation matrix X''_V. The use of the attention mechanism here allows the model to self-discover correlations between different modalities, and weight contributions accordingly.

We first apply a non-linear transformation to each of these representations for each node to obtain a scalar $k_{i,r}$ for each modality, which represents the importance of each modality.

$$k_{i,r} = W_{att}^{(1)} tanh(W_{att}^{(0)} x'_{i,r} + b) \tag{5}$$

where $x'_{i,r}$ is the intermediate hidden representation of the node for the r^{th} modality where $r \in \{p, q\}$, $W_{att}^{(0)}$ and $W_{att}^{(1)}$ are learnable weight matrices and b is the bias vector. The parameters are shared across all modalities.

We then normalize $k_{i,r}$ with a softmax function to obtain the weights:

$$\beta_{i,r} = \frac{exp(k_{i,r})}{\sum_{r=1}^{R} exp(k_{i,r})} \tag{6}$$

where $r \in \{p, q\}$ and $R \in \{P, Q\}$.

Finally, we use these weights to fuse the multiple representations for each node:

$$x_i'' = \sum_{p \in P}^{P} \beta_{i,p} x_{i,p}' + \sum_{q \in Q}^{Q} \beta_{i,q} x_{i,q}' \tag{7}$$

At this point, we denote the resultant aggregated hidden representations for all nodes as $X_V'' \in \mathbb{R}^{|V| \times d'}$. Two layers of GCNs are then used to encode the aggregated hidden representations X_V'' into the Gaussian embeddings. The use of stochastic Gaussian embeddings allows for more expressive embeddings to be generated.

$$X_V''^{(1)} = ReLU(\tilde{A} X_V'' W_V^{(0)}) \tag{8}$$

$$[\mu_V, \sigma_V^2] = \tilde{A} X_V''^{(1)} W_V^{(1)} \tag{9}$$

where $\tilde{A} = D^{-0.5} A D^{-0.5}$ is the symmetrically normalized adjacency matrix with D as the adjacency matrix's degree matrix, μ_V and σ_V^2 are the means and variances of the learned Gaussian embeddings of the nodes, and $W_V^{(0)}$, $W_V^{(1)}$ the learnable weight matrices.

The final embedding of each node can then be computed as -

$$Z_i = \mu_{V,i} + \sigma_{V,i}^2 \odot \epsilon \tag{10}$$

where ϵ is a random variable sampled from a standard normal distribution.

To reconstruct the adjacency matrix, we first apply an inner product between the embeddings of node i and node j to obtain:

$$[\mu_{i,j}, \sigma_{i,j}^2] = sigmoid(Z_i^T Z_j) \tag{11}$$

As the edges are binary, the edge between nodes i and j can be reconstructed with:

$$p_\theta(A_{ij} | Z_i, Z_j) = Ber(\mu_{i,j}) \tag{12}$$

where $Ber(\mu_{i,j})$ is the Bernoulli distribution parameterized by $\mu_{i,j}$.

4.2 Feature VAE

In MAAN, we use feature VAEs to encode node features of each modality. The feature VAE adopts stochastic Gaussian embeddings to allow for more expressive embeddings. Two linear layers with a tanh activation are first used to infer the Gaussian embeddings:

$$H_r = tanh(X_{V,r}^T W_r^{(0)} + b^{(0)}) \tag{13}$$

$$[\mu_r, \sigma_r^2] = H_r W_r^{(1)} + b^{(1)} \tag{14}$$

where μ_r and σ_r^2 are the means and variances of the learned Gaussian embeddings of the features of the r^{th} modality where $r \in \{p, q\}$, and $W_r^{(0)}$ and $W_r^{(1)}$ are learnable weight matrices. The embedding of each feature can then be computed as $Z_{r,a_r} = \mu_{r,a_r} + \sigma_{r,a_r}^2 \odot \epsilon$ where a_r is the index of the feature in the r^{th} modality, e.g., index of *Social* genre for *UI screen genre modality*, and ϵ is a random variable sampled from a standard normal distribution.

We then reconstruct the feature matrix by first applying an inner product between the embedding of the nodes and features:

$$[\mu_{r,i,a_r}, \sigma_{r,i,a_r}^2] = sigmoid(Z_i^T Z_{r,a_r}) \tag{15}$$

The node to feature edges for feature matrices with continuous values can be reconstructed with:

$$p_\theta(X_{r,i,a_r} | Z_i, Z_{r,a_r}) = \mathcal{N}(\mu_{r,i,a_r}, \sigma_{r,i,a_r}^2 I) \tag{16}$$

where $\mathcal{N}(\mu_{r,i,a_r}, \sigma^2_{r,i,a_r} I)$ is the Gaussian distribution parameterized by μ_{r,i,a_r} and σ^2_{r,i,a_r}.

The node to feature edges for feature matrices with binary values are reconstructed with:

$$p_\theta(X_{r,i,a_r}|Z_i, Z_{r,a_r}) = Ber(\mu_{r,i,a_r}) \tag{17}$$

where $Ber(\mu_{r,i,a_r})$ is the Bernoulli distribution parameterized by μ_{r,i,a_r}.

4.3 Objective Functions

Under the VAE framework, we want $q_\phi(z)$ for the encoders in the MAAN model to match the prior distribution $p_\theta(z)$ for the decoders in the MAAN model and the evidence lower bound objective (ELBO) is:

$$\mathcal{L}_{ELBO}(x) = \mathbb{E}_{q_{\phi(z|x)}}[logp_\theta(x|z)] - KL(q_\phi(z|x)||p(z)) \tag{18}$$

The first term measures the reconstruction loss while the second term is the Kullback-Leibler (KL) divergence. During training, the reparameterization trick is used to rewrite the expectation with respect to $q_\phi(z|x)$ such that the Monte Carlo estimate of the expectation is differentiable with respect to ϕ [21]. This VAE set-up is known to suffer from over-fitting when the KL divergence term is not strong enough or to generate uninformative embeddings when the KL divergence term is too restrictive. We use the MMD term [52] instead of the KL divergence term for the feature VAEs. We do not use α/λ hyper-parameters to balance between the KL divergence and MMD terms as proposed in [52] as tuning these hyper-parameters did not improve performance substantially. The use of the MMD term improves training stability and enhances the model's ability to learn meaningful embeddings from multiple modalities. The resulting objective function which we use in our model for the multi-modal attention-based GAE is:

$$\mathcal{L}_{ELBO}(x) = \mathcal{L}_{reconstruction} + \mathcal{L}_{KL} = \mathbb{E}_{q_{\phi(z|x)}}[logp_\theta(x|z)] - KL(q_\phi(z|x)||p(z)) \tag{19}$$

The resulting objective function which we use in our model for the feature VAEs is:

$$\mathcal{L}_{ELBO}(x) = \mathcal{L}_{reconstruction} + \mathcal{L}_{MMD} = \mathbb{E}_{q_{\phi(z|x)}}[logp_\theta(x|z)] + MMD(q_\phi(z)||p(z)) \tag{20}$$

We use the binary cross-entropy loss for $\mathcal{L}_{reconstruction}$ in the multi-modal attention-based GAE. For the feature VAEs, we use the binary cross-entropy loss when reconstructing a feature matrix of binary values, and mean square error loss when reconstructing a feature matrix of continuous values.

4.4 EMAAN: Extensions to Incorporate Edge Attributes

Capturing edge attributes within the message-passing framework enables richer network embeddings to be generated. Edge attributes, e.g., different number of occurrences of linkages between UI screens and elements, can indicate different roles and importances of different neighbouring UI entities. We now extend MAAN to handle edge attributes. We denote edge attributes as X_e, where $X_e \in \mathbb{R}^{|E| \times F_e}$ is an edge feature matrix with $x^e_{u,w} \in \mathbb{R}^{F_e}$ representing the feature vector of an edge (u, w).

To capture the effect of edge attributes, we extend MAAN by modifying the computation of the attention score $e^{(l)}_{ij}$ in equation (2) when computing the pair-wise attention scores between nodes. This approach has also been adopted in previous GNN works that model edge attributes [10, 11, 38, 49]. Depending on the nature of the edge attributes, the effect of edge attributes can be linear or non-linear. Hence, we consider both additive and multiplicative methods of capturing edge attributes:

- Additive - In **EMAAN-A**, we capture the effect of edge attributes in a linear fashion with concatenation by extending equation (2) to be:

$$e^{(l)}_{ij} = LeakyReLU(a^{(l)T}(s^{(l)}_{i,r}||s^{(l)}_{j,r}||W^{e,(l)}x^e_{i,j})) \tag{21}$$

- Multiplicative - In **EMAAN-M**, we capture the effect of edge attributes in a non-linear fashion with multiplication by extending equation (2) to be:

$$e_{ij}^{(l)} = W^{e,(l)} x_{i,j}^{e} \odot (a^{(l)T}(s_{i,r}^{(l)} || s_{j,r}^{(l)})) \tag{22}$$

In both EMAAN-A and EMAAN-M variants, $W^{e,(l)}$ is the learnable weight matrix of the l^{th} layer for the edge attributes, and $x_{i,j}^{e}$ is the attribute of the edge between node i and node j.

5 DATASETS AND EXPERIMENT SETUP

In this section, we describe the datasets utilized for experiments in this article, and the experiment setups used to evaluate MAAN and EMAAN against other baselines to determine the importance of representation learning using multi-modal and structural network information.

5.1 Datasets

The datasets used in the experiments are extracted from **RICO**, a repository covering the mobile UIs of 9,384 Android applications. We scraped updated metadata of these applications from Google Play Store and filtered out applications and UI screens still available for download in Feb 2020. This leaves us 6,583 applications, released from Jan 10 to Apr 17. The repository includes UI screenshots and their UI elements.

In [1], to ensure that the experiment findings for MAAN were valid for UI datasets with different characteristics, we extracted two datasets from the RICO repository, namely: (i) **RICO-N**, comprising the most recently released 1000 applications (Oct 15 to Apr 17); and (ii) **RICO-O**, comprising the earliest released 1000 applications (Jan 10 to Aug 11). The two datasets differ in a number of areas, as shown in Table 2. RICO-O is about twice the size of RICO-N by number of UI screens, UI elements and edges. The UI elements of both datasets have different dimensions for multi-hot class name vectors. The distributions of the genres in the two datasets are also different. The most popular genre in RICO-N is Social, whereas News and Magazines genre is most popular in RICO-O. RICO-O and RICO-N share 33 common genres between their UI screens, and RICO-N has three additional genres. We next extract UI elements occurring within each UI screen from the view hierarchies. A high number of occurrences of a UI element in a UI screen suggests that the UI element is important to the UI screen. On the other hand, we observe that a low number of occurrences are often boilerplate elements used to initialize the UI structure. Hence, we include an edge between a pair of UI screen and UI element only if the UI element occurs more than five times in the UI screen.

To evaluate EMAAN so as to determine the informativeness of edge attributes and to strengthen the validity of result findings, we extract three more datasets from the RICO repository where we do not filter edges based on such an empirical approach, and which include edge attributes. These additional datasets are: (iii) **W-RICO-N** and (iv) **W-RICO-O**, which are the RICO-N and RICO-O datasets with edge attributes that represent the number of occurrences of a UI element in a UI screen; and (v) **W-RICO-M**, a larger dataset of 2000 applications released earlier than RICO-N but later than RICO-O (Nov 12 to Sep 14) with the same type of edge attributes. W-RICO-M is a larger dataset with a larger number of nodes and edges comprising UIs from a different period, and hence enables us to further confirm the validity of our experiments. The details of these datasets are shown in Table 2.

For all datasets, visual information, i.e. the UI screen images, is encoded with a separate autoencoder. Following [6] and based on empirical experiments, we choose an embedding dimension of 64 for encoding the representations of UI screen images. Textual information is encoded with pre-trained Glove model, chosen based on empirical experiments, which generates embeddings of dimension 50. We pre-process the class names of the UI elements by breaking them up by their periods, special characters and camel casing. For example, *com.android.internal.policy.PhoneWindow$DecorView* is tokenized as *android, internal, policy, phone, window, decor, view*. Where relevant, we generate the class name features by: 1) using a pre-trained Char-NGram (CHAR), chosen based on

empirical experiments, which generates embeddings of dimension 100; and/or 2) using bag-of-words (BOW) representation, where the embedding dimension represents the number of unique tokens in the class names. UI screen ratings are based on the ratings of the applications that they are associated with. Other embedding dimensions and pre-trained encoders can also be utilized but these were chosen to manage the complexity and computation requirements while still achieving good performance. In general, the dataset statistics shown in Table 2 indicate that:

- There are less UI screens and elements in newer mobile applications (e.g., when we compare between the newer RICO-N dataset with the older RICO-O dataset), which could be indicative of UIs becoming less complex and more standardized over time.
- There are also less unique tokens for class names for the newer datasets (as indicated by the dimensions of the multi-hot vectors for UI element class names), which could similarly indicate greater standardization in newer UIs.
- The distribution of links between UI screens and elements is skewed with fat tails, based on the skewness (computed using the adjusted Fisher-Pearson skewness coefficient), which is greater than zero for all datasets.

Edge-lists are then divided into training, validation and testing datasets in the ratios 85%/5%/10% for the link prediction task. Under the co-embedding framework, we use the same generated embedding to evaluate both link prediction and attribute inference tasks. Hence, nodes that are part of the edges in the validation and testing edge-lists are used as validation/testing datasets for the attribute inference task, and their attribute values set to zero in the training dataset. For the rating prediction task, UI screen ratings are divided into training and testing datasets in the ratio 80%/20%. We do not utilize validation data for the rating prediction task as we simply use a simple series of dense layers for regression with the generated embeddings and do not tune any hyper-parameters for consistent comparisons between MAAN/EMAAN and baselines.

5.2 Experiment Setup

We compare the performance of MAAN and EMAAN with other state-of-the-art models on tasks that could facilitate the mobile UI design/development process. Each task is evaluated with separate evaluation metrics:

- **Link Prediction (UI Layout Assistance)** - The UI layout assistance task can be framed as a link prediction task, where we predict UI screens to be associated to each UI element via unobserved links. Each model returns the top ranked (or similar) UI screens for each UI element using the embeddings of the UI screens. To train and evaluate a prediction model with both positive and negative links, we randomly generate 10 pairs of nodes not in E for each positive link $(u, w) \in E$ as negative links. For each model, we use the inner product of node embeddings, i.e., $Z^T Z$, to predict the probability of a link forming between a pair of UI screen and UI element nodes. The model then ranks the candidate links in the test data by the inner product score, i.e., $sigmoid(Z^T Z)$. Area under the receiver operating characteristic curve (AUC) and average precision (AP) are used as the evaluation metrics.
- **Prediction of UI Screen Ratings (UI Design Evaluation)** - To evaluate UI designs, we predict UI screen ratings for the RICO dataset by using generated node embeddings as inputs to a regression model implemented with a series of five dense layers. This task is useful for predicting the success of new applications. We use root mean square error (RMSE) as the evaluation metric.
- **Attribute Inference (UI Feature Inference)** - We infer missing binary and continuous valued attributes of UI screens and elements based on affinities between nodes and attributes. For binary valued attributes, we treat this as predicting links between nodes and attributes, e.g., if a UI screen node has the categorical genre attribute 'Social' which is missing, we predict a link between the UI screen node and the categorical genre attribute 'Social', where 1 indicates the presence of the link, and 0 indicates the absence of the link.

Table 2. Dataset Statistics.

	RICO-N	RICO-O	W-RICO-N	W-RICO-O	W-RICO-M
UI Screens/Elements	5879/1563	9108/2920	5879/1563	9108/2920	15973/3880
Number of Edges	10762	19418	77922	136610	223722
Number of UI elements for each UI screen (Number of UI screens for each UI element)					
Average	1.9 (17.6)	1.9 (16.7)	6.6 (24.0)	7.5 (23.4)	7.0 (28.8)
Maximum	6.0 (1004.0)	7.0 (1711.0)	22.0 (5787.0)	21.0 (8954.0)	27.0 (15750.0)
Minimum	1.0 (1.0)	1.0 (1.0)	1.0 (1.0)	1.0 (1.0)	1.0 (1.0)
Skewness	0.9 (7.9)	0.9 (11.1)	0.7 (18.0)	0.6 (23.2)	0.6 (27.7)
Attribute Types and Dimensions					
Visual - Latent vectors of UI screen images *(Cont.)*	64	64	64	64	64
Textual - Glove vectors of app. descriptions for UI screen *(Cont.)*	50	50	50	50	50
Categorical - Multi-hot vectors of app. genre for UI screen *(Bin.)*	36	33	36	33	35
Textual - Multi-hot/CharNGram vectors of class name of UI element *(Bin./Cont.)*	1548/100	2161/100	1548/100	2161/100	2949/100
Categorical - Multi-hot vectors of UI element component type *(Bin.)*	25	25	25	25	25

Hence, we also sample 10 negative instances for each positive instance as per the link prediction task. We compare MAAN/EMAAN with CAN as GAT is not designed for attribute inference. The inner-product of generated node embeddings and attribute/feature embeddings is used to infer missing attributes. For binary attribute inference which involves UI screen genre, UI element class name BOW, and UI element component type, we measure the performance by AUC and AP. For UI element class name inference, we also split the class name into name tokens and compute the multi-label weighted F1 score to assess the performance of the model in inferring the class name-level attribute (and not just token-level attributes). The multi-label weighted F1 score returns the average score of all tokens in the class name weighted by support (number of true instances for each token). For continuous attribute inference which involves UI screen image, UI screen description, and UI element class name CHAR, RMSE is used.

- **Multi-Modal UI Retrieval** - This facilitates the retrieval of UI screens for an input UI screen query. The relevant results should be visually similar UI screens considering their multiple modalities. An overview of the retrieval evaluation process is shown in Figure 5. In this task, we use human judgement to determine the relevant UI screen results. We used the Amazon Mechanical Turk (AMT) platform to recruit participants for evaluating the relevance of retrieval results. To generate the user evaluation surveys, we use 20 randomly sampled UI screen images as queries to retrieve the top five results (based on nearest neighbors) from MAAN/EMAAN and the baseline models. For each query UI screen, we construct survey tasks, one for each of the models. Each survey task shows the same query UI screen and the result UI screens, one result UI screen from MAAN/EMAAN and each of the baseline models. Samples of the screen images presented in the surveys are shown in Figure 6. For quality assurance, we also add either the query UI screen or a

non-existent UI screen to each survey task. Each survey is then assigned to three AMT workers. Workers are asked to compare the query with randomly ordered results from MAAN/EMAAN and the baseline models and select UI screens that are most similar. If a worker fails to select the ground truth duplicate query UI screen image or selects the non-existent UI screen, the worker's responses for the entire survey will be discarded. The survey process is repeated until we obtain three good quality responses for each survey. Two metrics are used to measure retrieval performance. We use Average Precision@5 (precision@5) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the retrieval accuracy. For **precision@5**, a result is deemed relevant if two or more workers select it as relevant. As workers can select multiple UI screens for each query UI screen image, a visual check is also conducted to assess the reasonableness of their responses, and we compute the **NDCG** metric with the actual number of workers selecting the result as relevant.

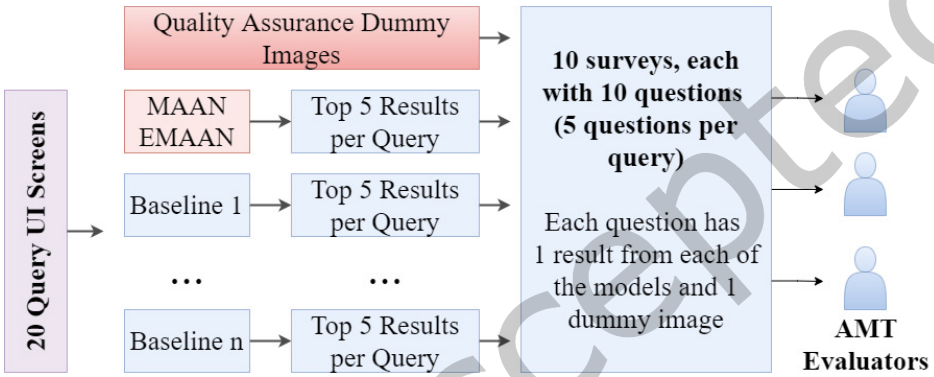

Fig. 5. Retrieval evaluation process.

Baselines and Settings. Co-embedding Attributed Network model (CAN) [30] and Graph Attention Network (GAT) [41] are chosen as baselines as they represent VAE and GNN-based state-of-the-art models respectively. We add a linear feature auto-encoder channel to GAT for it to co-embed representations of both attributes and nodes. For MAAN and EMAAN, we use four single-headed layers for each GAT module, and two layers for the GCN module. Feature VAEs use dense layers as encoders. We use the radial basis function as the kernel for MMD loss. Based on experiments with the validation dataset, the dimensions of the hidden node and feature representations d' is fixed at 64, and the dimension of the final node and feature embeddings d is fixed at 32. Training is run for 1000 epochs to generate the embeddings. The regression model for prediction of UI screen ratings is trained for 5000 epochs. For all models, an Adam optimizer [20] with a learning rate of 0.02 with a cosine annealing scheduler is used and dropout is set to 0.1. All models are implemented with Pytorch [33].

6 EXPERIMENT RESULTS

In this section, we present the results of experiments conducted with **i) MAAN and baseline models with the RICO-N and RICO-O datasets**; and **ii) EMAAN, MAAN and baseline models with the W-RICO-N, W-RICO-O and W-RICO-M datasets, which involve edge weights**.

6.1 Experiment Results with RICO-N/RICO-O

6.1.1 ***Link Prediction Results.*** Table 3 shows the results of the experiments relating to link prediction for the RICO-N and RICO-O datasets. MAAN effectively utilizes multi-modal attributes to generate embeddings

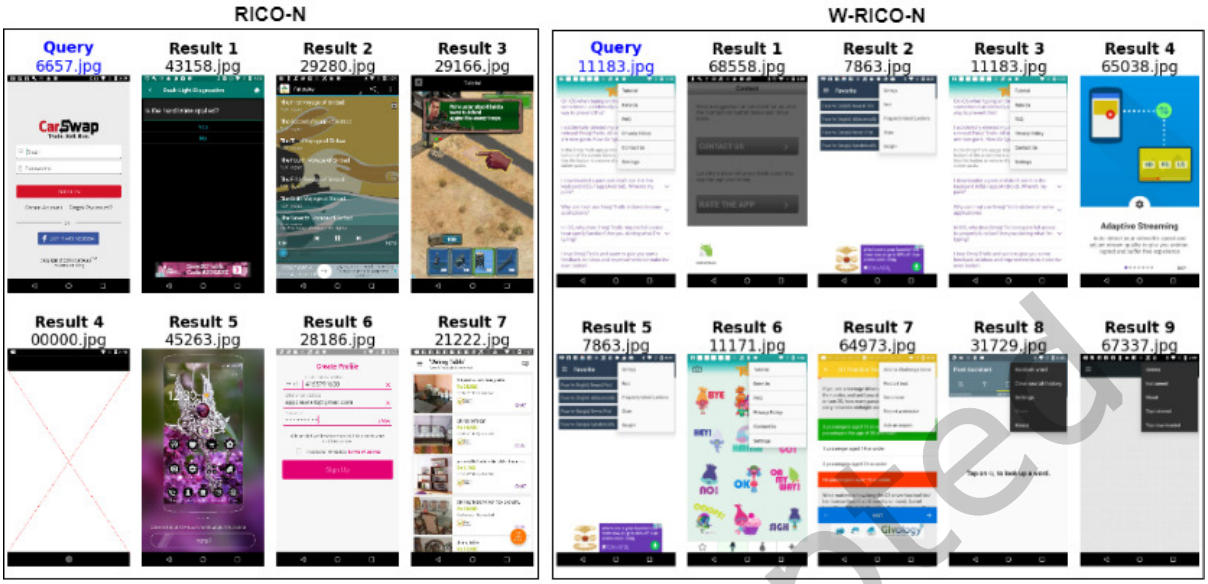

Fig. 6. Samples of retrieval results shown to AMT workers. For RICO-N, we compared MAAN with five baselines. For W-RICO-M, we compared MAAN, EMAAN-A and EMAAN-M with five baselines. Results are randomly ordered. Result 4 (left-hand-side) is a dummy non-existent UI screen. Result 3 (right-hand-side) is identical to query.

that consistently outperform baseline models on the link prediction task. CAN and GAT have greater difficulty utilizing continuous valued attributes to generate embeddings that perform well on the link prediction task, compared to when binary valued attributes are used. Differences in GAT performance when using UI element class names as attributes across the two datasets are likely due to differences in such attributes across the two datasets, e.g., differences in dimensions. VAE-based MAAN and CAN are less affected by such differences.

6.1.2 ***UI Screen Rating Prediction Results.*** Table 3 also sets out the results of regression experiments for the RICO-N and RICO-O datasets. For reference, the screen ratings range from 1 to 5, and the RMSE using mean of training dataset is 0.583 and 0.554 for RICO-N and RICO-O respectively. Here, one of the MAAN models returns the best results for both datasets. The differences in performance across all models for the UI screen rating prediction task is more narrow compared with other tasks. A potential reason for this could be that there are very little differences in the strength of the relationships between the different attributes and UI screen ratings.

6.1.3 ***Attribute Inference Results.*** Table 4 sets out the results of the experiments relating to attribute inference for the RICO-N and RICO-O datasets. The embeddings generated by MAAN consistently outperform embeddings generated by the baseline models across all modalities on this task. The differences in performance between MAAN and CAN using continuous valued attributes and binary valued attributes are even more stark here compared with the link prediction task. In some cases (e.g. screen description), the RMSE for the CAN model is around five times worse than the MAAN model.

6.1.4 ***Retrieval Evaluation Results.*** In this task, we use RICO-N dataset only and compare with five baseline models. The first four baseline models are the CAN and GAT models that utilize UI screen image and element component type attributes as they are likely to return better retrieval results. The fifth baseline model is Screen2Vec [25]. Screen2Vec is a related recent work that was published after MAAN [1]. Screen2Vec is similar to MAAN

Table 3. RICO-N and RICO-O - Link Prediction and Rating Regression Results. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE). We see that MAAN generates embeddings that consistently outperform baseline models on the link prediction and rating regression tasks. For all tables: Comp. refers to component, Desc. refers to description.

	Link Prediction				**Regression**	
	RICO-N		**RICO-O**		**RICO-N**	**RICO-O**
Model	**AUC**	**AP**	**AUC**	**AP**	**RMSE**	**RMSE**
GAT(UI Screen Image)	0.686	0.122	0.761	0.156	0.562	<u>0.517</u>
GAT(UI Screen Desc.)	0.680	0.122	0.729	0.141	<u>0.552</u>	0.522
GAT(UI Screen Genre)	0.939	0.843	0.928	0.658	0.682	0.536
GAT(UI Element Class Name CHAR)	0.379	0.068	0.861	0.716	0.555	0.520
GAT(UI Element Class Name BOW)	0.335	0.065	0.958	0.798	0.579	0.541
GAT(UI Element Comp. Type)	0.934	0.730	0.973	0.829	0.604	0.536
CAN(UI Screen Image)	0.972	0.881	0.960	0.828	0.617	0.534
CAN(UI Screen Desc.)	0.545	0.206	0.903	0.807	0.654	0.522
CAN(UI Screen Genre)	0.967	0.912	0.902	0.799	0.636	0.555
CAN(UI Element Class Name CHAR)	0.502	0.093	0.486	0.086	0.682	0.531
CAN(UI Element Class Name BOW)	0.965	0.856	0.960	0.842	0.593	0.542
CAN(UI Element Comp. Type)	0.963	0.829	0.938	0.866	0.638	0.533
MAAN(All Features Class Name CHAR)	**0.981**	**0.913**	**0.994**	**0.957**	**0.550**	**0.511**
MAAN(All Features Class Name BOW)	<u>0.977</u>	<u>0.905</u>	<u>0.981</u>	<u>0.917</u>	0.585	0.531

Table 4. RICO-N and RICO-O - Attribute Inference. Best performing models per metric marked in boldface. Weighted multi-label F1 score for UI Element Class Name in brackets. Higher (lower) values indicate better performance for AUC/AP/F1 score (RMSE). Embeddings generated by MAAN outperform embeddings generated by the baseline models across all modalities on the attribute inference task, with the differences being even clearer than the link prediction task.

	AUC/AP for bin. valued attributes			**RMSE for cont. valued attributes**		
Model	Screen Genre	Element Class Name *BOW*	Element Comp. Type	Screen Image	Screen Desc.	Element Class Name *CHAR*
			RICO-N			
CAN	0.501/0.091	0.817/0.571 (0.170)	0.733/0.198	1.451	8.958	5.685
MAAN	**0.776/0.223**	**0.922/0.663 (0.170)**	**0.928/0.541**	**0.994**	**1.675**	**1.107**
			RICO-O			
CAN	0.665/0.148	0.929/0.720 (0.196)	0.500/0.090	2.454	12.032	5.806
MAAN	**0.800/0.231**	**0.959/0.784 (0.196)**	**0.958/0.636**	**1.097**	**2.285**	**1.427**

in capturing multi-modal textual and visual information, but does not capture structural network information (see Section 3 for more details on Screen2Vec). Table 5 sets out the retrieval results. MAAN outperforms all baseline models by a large margin. Among the baselines, Screen2Vec's performance is closest to MAAN, which indicates the importance of capturing multi-modal information. The difference in performance between MAAN and Screen2Vec could be due to Screen2Vec not capturing network structural information. CAN using the binary

Table 5. Retrieval Evaluation - RICO-N. Best and second best performing models per metric marked in boldface and underlined. 20.0% of all results selected as relevant by 2 or more AMT workers. Higher values indicate better performance. MAAN outperforms all baseline models by a large margin, with Screen2Vec's performance being closest to MAAN, which indicates the importance of capturing multi-modal information.

Model	**Precision@5**	**NDCG**
GAT(UI Screen Image)	18.0%	46.0%
GAT(UI Element Comp. Type)	18.0%	46.0%
CAN(UI Screen Image)	17.0%	41.2%
CAN(UI Element Comp. Type)	71.0%	82.9%
Screen2Vec	<u>79.0%</u>	<u>84.2%</u>
MAAN	**89.0%**	**92.6%**

Table 6. Model Ablation for MAAN with RICO-N - Link Prediction/Regression. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE).

	Link Prediction		**Regression**
Modification	AUC	AP	RMSE
Replace GAT layers with GCN	0.966	<u>0.896</u>	<u>0.594</u>
Use multi-heads (4 heads) for each GAT layer	<u>0.970</u>	0.888	0.636
Replace attention fusion with addition	0.689	0.129	2.570e5
MAAN	**0.981**	**0.913**	**0.550**

Table 7. Model Ablation for MAAN with RICO-N - Attribute Inference. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE).

	AUC/AP for bin. valued attributes		**RMSE for cont. valued attributes**		
Modification	UI Screen Genre	UI Element Comp. Type	UI Screen Image	UI Screen Desc.	UI Element Class Name *CHAR*
Replace GAT layers with GCN	0.609/0.138	0.761/0.266	**0.819**	2.186	3.835
Use multi-heads (4 heads) for each GAT layer	<u>0.771/0.201</u>	**0.955/0.680**	1.122	<u>1.589</u>	**1.037**
Replace attention fusion with addition	0.519/0.098	0.686/0.140	2.487	**1.059**	1.611
MAAN	**0.776/0.223**	<u>0.928/0.541</u>	<u>0.994</u>	1.675	<u>1.107</u>

valued UI element component type attribute also performs relatively well, in line with what we had observed for other tasks.

6.1.5 ***Ablation Studies on MAAN with RICO-N.*** We first conduct ablation studies on the main MAAN model in this section, before extending it to capture edge attributes in EMAAN in the subsequent sections. Tables 6 and 7 set out the results of the ablation studies for different model configurations of MAAN using the RICO-N dataset. The choices made in the proposed MAAN lead to better link prediction and regression performance.

Table 8. Modality Ablation for MAAN with RICO-N - Link Prediction/Regression. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE).

	Link Prediction		**Regression**
Modification	AUC	AP	RMSE
MAAN-Image+Classname	<u>0.977</u>	<u>0.907</u>	0.564
MAAN-Image+Classname+Desc.	0.716	0.617	<u>0.555</u>
MAAN-Image+Classname+Desc.+Genre	0.907	0.828	0.600
MAAN All	**0.981**	**0.913**	**0.550**

Table 9. Modality Ablation for MAAN with RICO-N - Attribute Inference. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE).

Modification	UI Screen Image (RMSE)	UI Element Class Name *CHAR* (RMSE)	UI Screen Desc. (RMSE)	UI Screen Genre (AUC/AP)
MAAN-Image+Classname	1.353	<u>0.712</u>	-	-
MAAN-Image+Classname+Desc.	<u>0.729</u>	**0.536**	**0.699**	-
MAAN-Image+Classname+Desc.+Genre	**0.659**	1.152	<u>1.312</u>	<u>0.746/0.193</u>
MAAN All	0.994	1.107	1.675	**0.776/0.223**

For attribute inference, multi-head can improve performance for some attributes. [40] found that multi-head attention allows information from different representation sub-spaces at different positions to be attended to. Here, multi-head allows the model to jointly attend to information from different nodes. Tables 8 and 9 set out the results of ablation studies for different number of modalities. MAAN-Image+Classname refers to MAAN with UI screen images and element class-names (CHAR) attributes. We further add the UI screen description for MAAN-Image+Classname+Desc., and add the UI screen genre for MAAN-Image+Classname+Desc.+Genre. For link prediction and regression, using more modalities improves performance. For attribute inference, using all features (i.e. MAAN All) does not necessarily lead to better performance. Different combinations of features, i.e. feature selection, can further improve the performance of MAAN.

6.2 Experiment Results Involving Edge Weights

In this section, we repeat the experiments that were conducted above for MAAN and baselines, but with the two EMAAN models and on the additional W-RICO-N, W-RICO-O and W-RICO-M datasets.

6.2.1 ***Link Prediction Results.*** Table 10 shows the results of the experiments relating to link prediction for the W-RICO-N, W-RICO-O and W-RICO-M datasets. The key differences between experiments on the RICO-N/RICO-O and W-RICO-N/W-RICO-O datasets are the higher number of edges, as shown in Table 2, and the utilization of edge attributes, which are the the number of occurrences of a UI element in a UI screen, in the EMAAN models. While MAAN is still able to outperform the baselines, the results show that EMAAN models materially improve in performance over MAAN. In particular, we see that EMAAN-M shows the best performance, which indicates a non-linear relationship between the probability of linkages between UI screens and elements and the edge attributes.

6.2.2 ***UI Screen Rating Prediction Results.*** Table 11 sets out the results of regression experiments for the W-RICO-N, W-RICO-O and W-RICO-M datasets. For reference, RMSE using mean of training dataset is 0.583,

Table 10. W-RICO-N/W-RICO-O/W-RICO-M - Link Prediction Results. Best and second best performing models per metric marked in boldface and underlined. Higher values indicate better performance. While MAAN continues to outperform the baselines, EMAAN, by further capturing edge attributes, materially improves in performance over MAAN.

	W-RICO-N		**W-RICO-O**		**W-RICO-M**	
Model	**AUC**	**AP**	**AUC**	**AP**	**AUC**	**AP**
GAT(UI Screen Image)	0.829	0.215	0.283	0.060	0.854	0.244
GAT(UI Screen Desc.)	0.819	0.210	0.787	0.178	0.842	0.232
GAT(UI Screen Genre)	0.795	0.203	0.216	0.056	0.381	0.067
GAT(UI Element Class Name CHAR)	0.617	0.105	0.500	0.083	0.408	0.071
GAT(UI Element Comp. Type)	0.905	0.372	0.914	0.346	0.915	0.508
CAN(UI Screen Image)	0.909	0.841	0.915	0.836	0.903	0.844
CAN(UI Screen Desc.)	0.499	0.096	0.502	0.092	0.512	0.109
CAN(UI Screen Genre)	0.908	0.827	0.922	0.844	0.905	0.846
CAN(UI Element Class Name CHAR)	0.499	0.094	0.503	0.090	0.500	0.092
CAN(UI Element Comp. Type)	0.907	0.840	0.917	0.842	0.905	0.840
MAAN(All Feat.)	0.914	0.846	0.923	0.843	0.919	0.871
EMAAN-A(All Feat.)	<u>0.934</u>	<u>0.855</u>	<u>0.929</u>	<u>0.854</u>	<u>0.933</u>	<u>0.877</u>
EMAAN-M(All Feat.)	**0.957**	**0.872**	**0.934**	**0.860**	**0.949**	**0.889**

0.554 and 0.536 for W-RICO-N, W-RICO-O and W-RICO-M respectively. MAAN and EMAAN outperform all baselines. The differences between the EMAAN models and MAAN is however smaller than other tasks. This could indicate that multi-modal information is more important for the screen rating prediction task than the structural network information and edge attributes.

6.2.3 ***Attribute Inference Results.*** Table 12 sets out the results of the experiments relating to attribute inference for the W-RICO-N, W-RICO-O and W-RICO-M datasets. MAAN and EMAAN similarly outperform all baselines across all datasets, particularly for the continuous-valued screen image, screen description and element class name attributes. The utilization of edge attributes by EMAAN leads to an improvement in performance for a number of modalities, demonstrating the usefulness of capturing edge attributes. For the W-RICO-M dataset, we notice that EMAAN-M generally shows better performance than EMAAN-A for most modalities. This could be due to a stronger non-linear relationship between the edge attributes and node attributes in the W-RICO-M dataset.

6.2.4 ***Retrieval Evaluation Results.*** We conduct an additional user evaluation on the W-RICO-M dataset. Table 13 sets out the retrieval results for the W-RICO-M dataset. MAAN and EMAAN outperform all baseline models, with EMAAN models generally performing better than MAAN. Among the baselines, Screen2Vec's performance is again closest to MAAN and EMAAN, which indicates the importance of capturing multi-modal information.

7 QUALITATIVE ANALYSIS

In this section, we conduct two sets of qualitative analyses on the MAAN and EMAAN models, by visualizing the learnt embeddings, as well as the attention weights for multi-modal fusion that were learnt by the key models.

7.1 Visualization of Embedding Space

We use t-Distributed Stochastic Neighbor Embedding [29] with perplexity of 50.0, early exaggeration of 20.0 and learning rate of 200.0 to project the generated embeddings to two dimensions to visualize the embedding spaces.

Table 11. W-RICO-N/W-RICO-O/W-RICO-M - Regression Results. Best and second best performing models per metric marked in boldface and underlined. Lower values indicate better performance. MAAN and EMAAN again outperform all baselines.

Model	**W-RICO-N**	**W-RICO-O**	**W-RICO-M**
GAT(UI Screen Image)	0.575	0.544	0.586
GAT(UI Screen Desc.)	0.572	0.547	0.588
GAT(UI Screen Genre)	0.581	0.525	0.589
GAT(UI Element Class Name CHAR)	0.576	0.547	0.587
GAT(UI Element Comp. Type)	0.598	0.536	0.585
CAN(UI Screen Image)	0.613	0.550	0.580
CAN(UI Screen Desc.)	0.666	0.562	0.586
CAN(UI Screen Genre)	0.609	0.544	0.584
CAN(UI Element Class Name CHAR)	0.642	0.555	0.594
CAN(UI Element Comp. Type)	0.607	0.533	0.584
MAAN(All Feat.)	0.550	0.511	0.531
EMAAN-A(All Feat.)	**0.549**	**0.509**	0.527
EMAAN-M(All Feat.)	0.556	0.511	**0.524**

Table 12. W-RICO-N/W-RICO-O/W-RICO-M - Attribute Inference. Best and second best performing models per metric marked in boldface and underlined. Higher (lower) values indicate better performance for AUC/AP (RMSE). MAAN and EMAAN outperform all baselines across all datasets. The utilization of edge attributes by EMAAN leads to an improvement in performance for a number of modalities, demonstrating the usefulness of capturing edge attributes.

	AUC/AP for bin. valued attributes		**RMSE for cont. valued attributes**		
Model	Screen Genre	Element Comp. Type	Screen Image	Screen Desc.	Element Class Name CHAR
			W-RICO-N		
CAN	0.681/0.178	0.920/0.434	1.217	6.107	5.902
MAAN	**0.780/0.207**	0.965/0.660	0.681	1.211	0.738
EMAAN-A	0.776/0.223	0.928/0.541	0.658	**0.966**	**0.737**
EMAAN-M	0.778/0.202	**0.966/0.666**	**0.490**	1.010	1.052
			W-RICO-O		
CAN	0.692/0.141	0.917/0.842	1.388	5.897	5.665
MAAN	0.805/0.250	0.958/0.609	0.581	**1.090**	0.972
EMAAN-A	**0.809/0.250**	**0.959/0.609**	**0.539**	1.145	**0.925**
EMAAN-M	0.805/0.246	0.957/0.585	0.669	1.441	1.011
			W-RICO-M		
CAN	0.686/0.149	0.825/0.367	2.077	6.173	5.993
MAAN	0.788/0.217	0.961/0.640	1.127	2.260	**1.188**
EMAAN-A	**0.792/0.217**	**0.973/0.727**	1.129	1.880	1.314
EMAAN-M	**0.792/0.217**	0.972/0.725	**0.672**	**1.568**	1.216

Figures 7, 8, and 9 visualize the embedding spaces generated for the RICO-N, W-RICO-N and W-RICO-M datasets respectively. The presence of more clusters seems correlated with better task performance. The embedding

Table 13. W-RICO-M - Retrieval Evaluation. Best and second best performing models per metric marked in boldface and underlined. 37.0% of all results selected as relevant by 2 or more AMT workers for W-RICO-M. Higher values indicate better performance. MAAN and EMAAN outperform all baseline models, with EMAAN models generally performing better than MAAN. Screen2Vec's performance is again closest to MAAN and EMAAN, which indicates the importance of capturing multi-modal information.

Model	**Precision@5**	**NDCG**
GAT(UI Screen Image)	24.0%	63.6%
GAT(UI Element Comp. Type)	24.0%	63.6%
CAN(UI Screen Image)	24.0%	77.7%
CAN(UI Element Comp. Type)	16.0%	70.1%
Screen2Vec	<u>52.0</u>%	84.7%
MAAN	**57.0**%	<u>90.9</u>%
EMAAN-A	**57.0**%	**91.7**%
EMAAN-M	**57.0**%	**91.7**%

generated by the MAAN model has more obvious node clusters than those from CAN. For CAN, the embeddings for modalities where performance is poorer do not have clear node clusters forming.

If we compare MAAN with the EMAAN models on the W-RICO-N and W-RICO-M datasets, we notice that the EMAAN models generally produce more clusters that are also more clearly separated. On the W-RICO-N and W-RICO-M datasets, we similarly observe that the CAN embeddings for modalities where performance is poorer do not have clear node clusters forming.

7.2 Attention Weights

The attention weights for multi-modal fusion learnt by the MAAN, EMAAN-A and EMAAN-M models for each of the nodes, i.e. $\beta_{i,r}$ described in Section 4, are plotted as heatmaps in Figures 10, 11 and 12 for MAAN, EMAAN-A and EMAAN-M models (for the W-RICO-N dataset) respectively. The attention weights show the *relative importance* of each of the modalities. The attention weights in MAAN are focused on the UI screen image and description modalities (corresponding to the darker colors for screen image and description modalities in Figure 10), whereas the attention weights in EMAAN-A and EMAAN-M models indicate that these models utilize features across more modalities, e.g., for EMAAN-A we observe darker colors for the UI screen image, description and genre, as well as the UI element component type modalities in Figure 11. In general, all three models appear to have learnt that the UI screen image modality is important. The attention weights in Figure 11 and 12 show that in addition to the UI screen image and description modalities, the EMAAN-A and EMAAN-M models also focus on the UI screen genre modality. The differences between the attention weights for the EMAAN-A and EMAAN-M model could be due to the differences in how edge attributes are captured, i.e., linear for EMAAN-A but non-linear for EMAAN-M. To facilitate comparisons between models and datasets, we also plot the average attention weights (averaged across all nodes) for the MAAN, EMAAN-A and EMAAN-M models and compare them across the W-RICO-N, W-RICO-O and W-RICO-M datasets in Table 14. Similarly, a darker shade of blue indicates higher average attention weights for the modality, relative to the other modalities for the model and dataset. While there are variations in the average attention weights across the three datasets, we see that the modality with the highest attention weights is generally in line with what we observed for the W-RICO-N dataset.

8 DISCUSSION

From the experimental results on link prediction, rating prediction and attribute inference across all datasets, we see that the proposed MAAN and EMAAN models are able to effectively outperform the baseline CAN and GAT

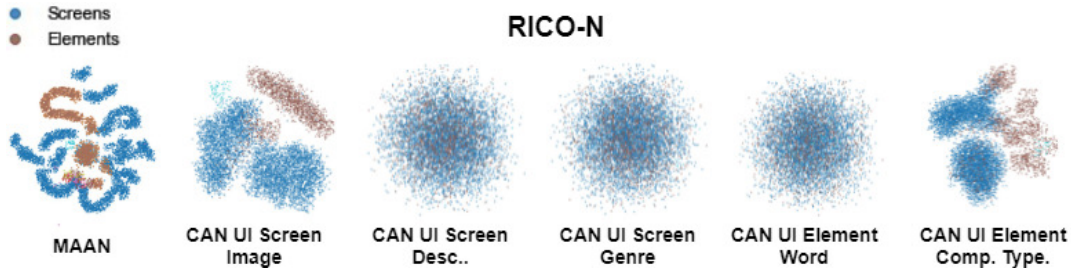

Fig. 7. Visualization of embedding spaces for RICO-N. We focus on the embeddings of UI screens (blue datapoints) and elements (brown datapoints) when comparing MAAN and CAN. As MAAN captures multi-modal attributes, it additionally generates embeddings for the multi-modal features (e.g., cyan, red datapoints). These embeddings were not utilized in the paper, but we have included them in the figure for completeness.

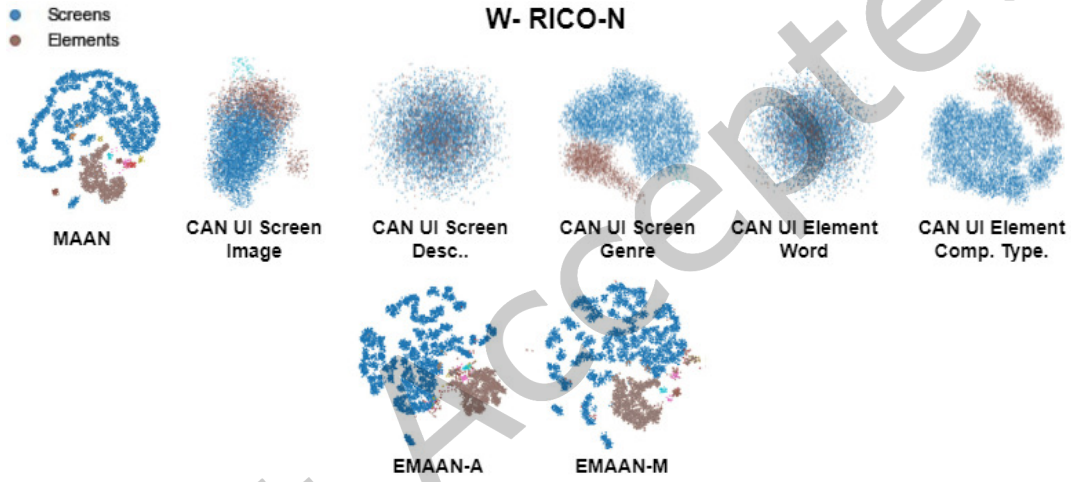

Fig. 8. Visualization of embedding spaces for W-RICO-N. We focus on the embeddings of UI screens (blue datapoints) and elements (brown datapoints) when comparing MAAN, EMAAN and CAN. As MAAN and EMAAN capture multi-modal attributes, it additionally generates embeddings for the multi-modal features (e.g., cyan, red datapoints). These embeddings were not utilized in the paper, but we have included them in the figure for completeness.

models that only utilize information from a single modality. We also see that being able to capture edge attributes leads to better performance with the various EMAAN models. In relation to the retrieval task, the two sets of user evaluations conducted on the RICO-N and W-RICO-M datasets consistently show that the MAAN/EMAAN model is able to more effectively capture structural network information and information from multiple modalities, and hence generate embeddings that retrieve more relevant results than the baseline CAN, GAT and Screen2Vec models.

From the multiple experiments, we see that being able to utilize information from more modalities generally leads to better performance. However, the ablation studies also show that factor selection is still important as performance could decline for certain combinations of modalities.

While MAAN/EMAAN generally outperforms both the baseline CAN and GAT models, we do notice that the difference between MAAN/EMAAN and the baseline CAN and GAT models is more pronounced when attributes

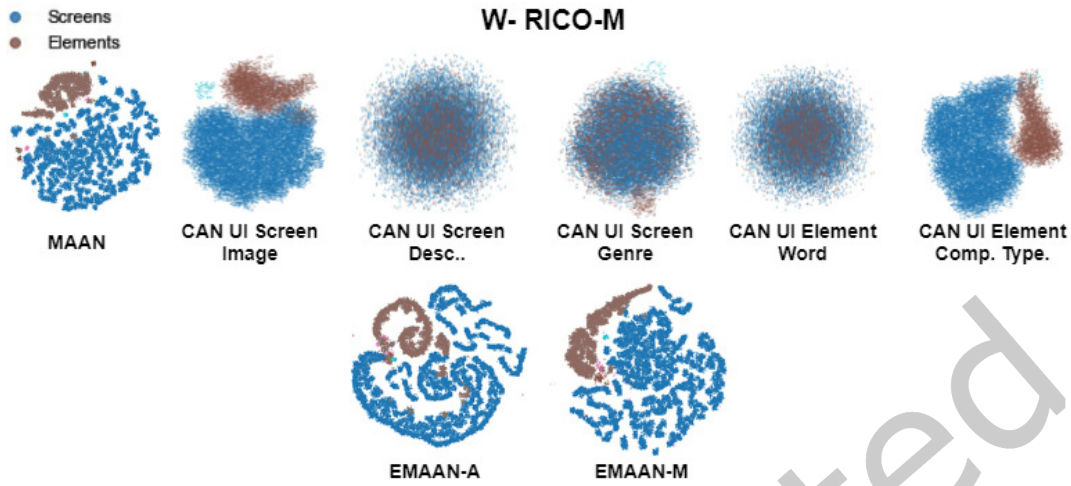

Fig. 9. Visualization of embedding spaces for W-RICO-M. We focus on the embeddings of UI screens (blue datapoints) and elements (brown datapoints) when comparing MAAN, EMAAN and CAN. As MAAN and EMAAN capture multi-modal attributes, it additionally generates embeddings for the multi-modal features (e.g., cyan, red datapoints). These embeddings were not utilized in the paper, but we have included them in the figure for completeness.

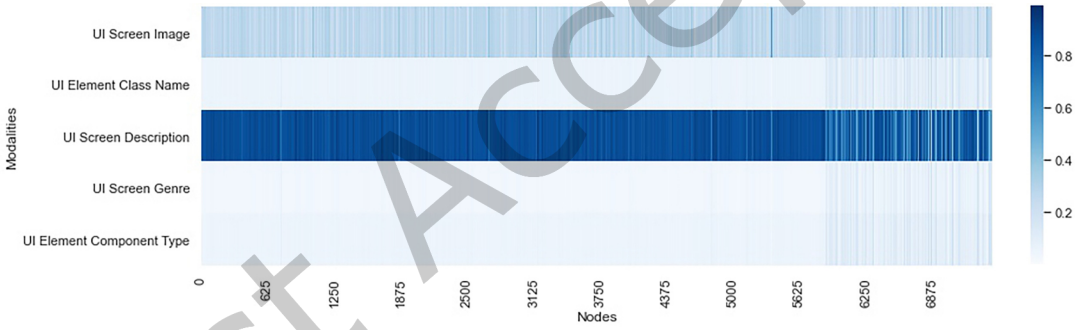

Fig. 10. Visualization of attention weights for MAAN (W-RICO-N).

are continuous-valued. We postulate that the use of the MMD loss term in MAAN/EMAAN helps it to better utilize continuous valued attributes when generating embeddings.

The use of attention mechanisms generally leads to better performance as seen in the ablation studies. In addition, we also show that the learned attention weights used for multi-modal fusion are useful in understanding the relative contribution of information from each modality to the generated embeddings.

The improvement in performance when we include the additive and multiplicative mechanisms to capture edge attributes in EMAAN further demonstrates the usefulness of capturing edge attributes. We generally see better performance with EMAAN-M - which could be linked to the non-linear relationship between the edge attributes and the tasks.

The key idea demonstrated in this paper is that capturing semantics that are available in the network structures and multimodal information of UI datasets can help improve predictive performance on a range of UI-related

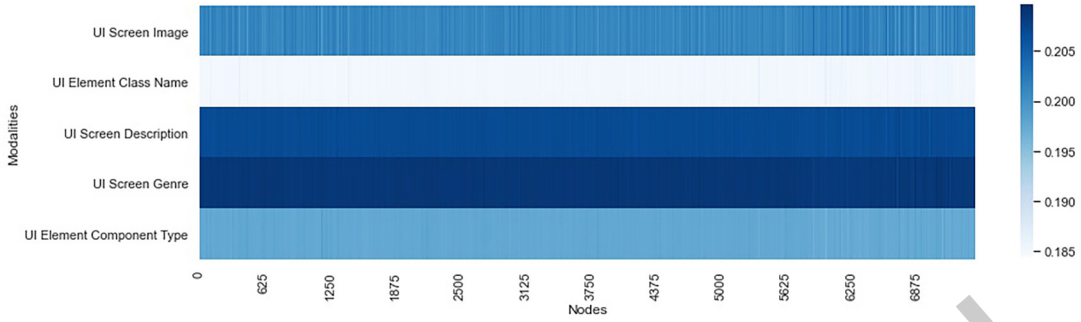

Fig. 11. Visualization of attention weights for EMAAN-A (W-RICO-N).

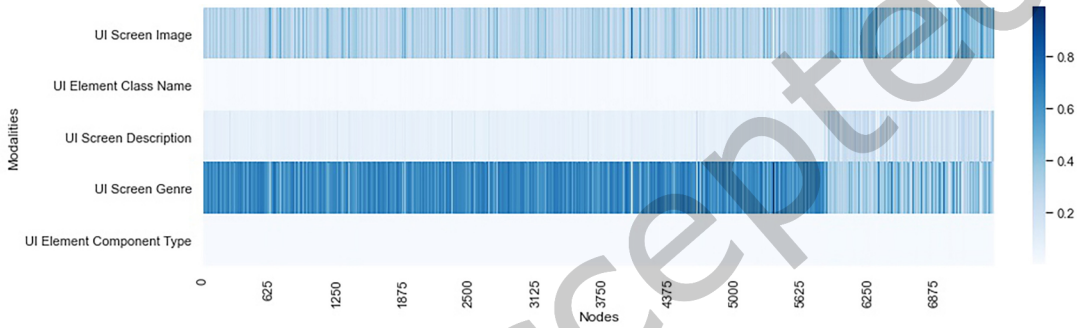

Fig. 12. Visualization of attention weights for EMAAN-M (W-RICO-N).

tasks. Section 2 described key scenarios where such predictions can be applied in tasks that can facilitate and assist UI design processes. Predicting links between UI screen and elements can assist UI developers and designers in narrowing down designs for viable UI layouts; prediction of UI ratings can help provide an early indication of UI popularity; UI attribute/feature inference can improve the quality of UI datasets; and being able to retrieve semantically similar screens can facilitate UI designers and developers in searching for UI designs to use as reference designs. More importantly, being able to capture and utilize such rich information within UI datasets, which are likely to increase in volume and variety over time, could enable a more data-driven approach to be taken in improving the usability of UI designs.

9 CONCLUSION AND FUTURE WORK

Based on the results of our experiments, we see that the MAAN and EMAAN models, due to their use of 1) information from multiple modalities 2) structural network information and edge attributes, 3) attention mechanisms, and 4) the MMD loss term performs more consistently than the baseline models whose performance varies based on the nature of the information. We also see that the use of node-wise attention in GAT layers together with modality-wise attention fusion leads to better performance. The learned attention weights also help us understand the importance of information from each modality. The experiment results also demonstrate that MAAN/EMAAN is able to generate UI screen and element embeddings that can be used in a range of downstream tasks, e.g. predicting links between UI screens and elements for UI layout assistance, UI rating predictions for UI design evaluation, missing UI attribute/feature inference and UI screen retrieval, and could also potentially

Table 14. Average attention weights (across all nodes) for MAAN, EMAAN-A, EMAAN-M for the three datasets. A darker shade of blue indicates higher average attention weights for the modality, relative to the other modalities for the model and dataset.

	MAAN	EMAAN-A	EMAAN-M
		W-RICO-N	
UI Screen Image	0.08	0.20	0.17
UI Element Class Name	0.03	0.19	0.00
UI Screen Description	0.85	0.21	0.01
UI Screen Genre	0.01	0.21	0.81
UI Element Component Type	0.02	0.20	0.01
		W-RICO-O	
UI Screen Image	0.02	0.24	0.18
UI Element Class Name	0.03	0.20	0.11
UI Screen Description	0.55	0.08	0.09
UI Screen Genre	0.11	0.40	0.52
UI Element Component Type	0.28	0.07	0.10
		W-RICO-M	
UI Screen Image	0.01	0.28	0.13
UI Element Class Name	0.30	0.00	0.12
UI Screen Description	0.60	0.27	0.00
UI Screen Genre	0.02	0.36	0.73
UI Element Component Type	0.07	0.09	0.02

complement other UI development-related systems. Given MAAN/EMAAN's performance on the five distinct datasets that were used for experiments, the model is likely to be equally applicable to iOS applications even if characteristics of the information differ. Directions for future work include:

- Multiple node and edge-types: The current MAAN/EMAAN models utilize structural network information for a bipartite network and a single edge-type. Capturing other node and edge types, i.e. heterogeneous networks, could improve performance.
- Positional information: We did not capture the order of the UI nodes - e.g., order of UI elements in the view hierarchy. Such positional information could improve task performance.
- End-to-end model: Information from some modalities - e.g., images, descriptions - are encoded separately. Integrating the encoders within MAAN/EMAAN could lead to better performance.
- Exploring the use of the framework outlined in this paper for UIs in other domains, e.g., web UIs or tangible UIs, and other tasks relevant for such UIs.
- Generating representations of UI objects for UI networks formed from other types of linkages, e.g., linkages formed by how users interact with different UI objects.
- Application of explainability methods for deep learning and graph neural networks [50] to discover features that are important for usability in UI designs.

ACKNOWLEDGMENTS

This research is supported by the National Research Foundation, Singapore under its Strategic Capabilities Research Centres Funding Initiative. Gary Ang is supported by a Monetary Authority of Singapore Postgraduate Scholarship. Any opinions, findings and conclusions or recommendations expressed in this material are those

of the author(s) and do not reflect the views of National Research Foundation, Singapore, nor the Monetary Authority of Singapore.

REFERENCES

[1] Gary Ang and Ee-Peng Lim. 2021. Learning Network-Based Multi-Modal Mobile User Interface Embeddings. In *26th International Conference on Intelligent User Interfaces (IUI), 2021.*

[2] Sara Bunian et al. 2021. VINS: Visual Search for Mobile User Interface Design. In *Conference on Human Factors in Computing Systems (CHI), 2021.*

[3] Yukuo Cen et al. 2019. Representation Learning for Attributed Multiplex Heterogeneous Network. In *Proceedings of the 25th International Conference on Knowledge Discovery & Data Mining (KDD), 2019.*

[4] Chunyang Chen et al. 2019. Gallery D.C.: Design Search and Knowledge Discovery through Auto-Created GUI Component Gallery. *Proc. ACM Hum.-Comput. Interact.* 3, CSCW, Article 180 (Nov. 2019), 22 pages.

[5] Jieshan Chen et al. 2020. Object detection for graphical user interface: old fashioned or deep learning or a combination?. In *28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, 2020.*

[6] Biplab Deka et al. 2017. Rico: A Mobile App Dataset for Building Data-Driven Design Applications. In *Proceedings of the 30th Annual Symposium on User Interface Software and Technology (UIST), 2017.*

[7] Michal Derezinski et al. 2018. Discovering Surprising Documents with Context-Aware Word Representations. In *23rd International Conference on Intelligent User Interfaces (IUI), 2018.*

[8] Nuno Do Nascimento Correia and Atau Tanaka. 2021. From GUI to AVUI: situating audiovisual user interfaces within human-computer interaction and related fields. *EAI Endorsed Transactions on Creative Technologies* 8, 27 (2021).

[9] Yuxiao Dong et al. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *Proceedings of the 23rd ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2017.*

[10] Peter W. Battaglia et al. 2018. Relational inductive biases, deep learning, and graph networks. *CoRR* (2018).

[11] Justin Gilmer et al. 2017. Neural Message Passing for Quantum Chemistry. In *Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.*

[12] Asnat Greenstein-Messica et al. 2017. Session-Based Recommendations Using Item Embedding. In *22nd International Conference on Intelligent User Interfaces (IUI), 2017.*

[13] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2016.*

[14] William L. Hamilton et al. 2017. Inductive Representation Learning on Large Graphs. In *Annual Conference on Neural Information Processing Systems (NIPS), 2017.*

[15] Xiangnan He et al. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *Proceedings of the 43rd International ACM Conference on Research and Development in Information Retrieval (SIGIR), 2020.*

[16] Zecheng He et al. 2021. ActionBert: Leveraging User Actions for Semantic Understanding of User Interfaces. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, 2021.*

[17] Feiran Huang et al. 2019. Network embedding by fusing multimodal contents and links. *Knowledge-Based Systems* 171 (2019), 44–55.

[18] Forrest Huang et al. 2019. Swire: Sketch-based User Interface Retrieval. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI), 2019.*

[19] Wentao Huang et al. 2020. BiANE: Bipartite Attributed Network Embedding. In *Proceedings of the 43rd International ACM Conference on Research and Development in Information Retrieval (SIGIR), 2020.*

[20] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations (ICLR), 2015.*

[21] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations (ICLR), 2014.*

[22] Thomas N Kipf and Max Welling. 2016. Variational Graph Auto-Encoders. In *NIPS Workshop on Bayesian Deep Learning.*

[23] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations (ICLR), 2017.*

[24] Chunggi Lee et al. 2020. GUIComp: A GUI Design Assistant with Real-Time, Multi-Faceted Feedback. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI), 2020.*

[25] Toby Jia-Jun Li et al. 2021. Screen2Vec: Semantic Embedding of GUI Screens and GUI Components. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI), 2021.*

[26] Yang Li. 2012. Gesture-based interaction: a new dimension for mobile user interfaces. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI), 2012.*

[27] Dawen Liang et al. 2018. Variational Autoencoders for Collaborative Filtering. In *Proceedings of the 2018 World Wide Web Conference (WWW)*.
[28] Thomas F. Liu et al. 2018. Learning Design Semantics for Mobile Apps. In *The 31st Annual ACM Symposium on User Interface Software and Technology (UIST), 2018*.
[29] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. *Journal of Machine Learning Research* 9, Nov (2008), 2579–2605.
[30] Zaiqiao Meng et al. 2019. Co-Embedding Attributed Networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining (WSDM), 2019*.
[31] Zaiqiao Meng et al. 2019. Semi-supervisedly Co-embedding Attributed Networks. In *Annual Conference on Neural Information Processing Systems (NIPS), 2019*.
[32] Peter O'Donovan et al. 2015. DesignScape: Design with Interactive Layout Suggestions. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI), 2015*.
[33] Adam Paszke et al. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Annual Conference on Neural Information Processing Systems (NIPS), 2019*.
[34] Bryan Perozzi et al. 2014. DeepWalk: online learning of social representations. In *Proceedings of the 20th ACM International Conference on Knowledge Discovery and Data Mining (KDD), 2014*.
[35] Andreas Riegler and Clemens Holzmann. 2018. Measuring Visual User Interface Complexity of Mobile Applications With Metrics. *Interacting with Computers* 30, 3 (2018), 207–223.
[36] Aneeshan Sain et al. 2021. StyleMeUp: Towards Style-Agnostic Sketch-Based Image Retrieval. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021*.
[37] Michael Sejr Schlichtkrull et al. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference (ESWC), 2018*.
[38] Martin Simonovsky and Nikos Komodakis. 2017. Dynamic Edge-Conditioned Filters in Convolutional Neural Networks on Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017*.
[39] Rianne van den Berg et al. 2017. Graph Convolutional Matrix Completion. *CoRR* (2017).
[40] Ashish Vaswani et al. 2017. Attention is All you Need. In *Annual Conference on Neural Information Processing Systems (NIPS), 2017*.
[41] Petar Veličković et al. 2018. Graph Attention Networks. *International Conference on Learning Representations (ICLR), 2018*.
[42] Xiao Wang et al. 2019. Heterogeneous Graph Attention Network. In *Proceedings of the 2019 World Wide Web Conference (WWW)*.
[43] Xiang Wang et al. 2019. Neural Graph Collaborative Filtering. In *Proceedings of the 42nd International ACM Conference on Research and Development in Information Retrieval (SIGIR), 2019*.
[44] Evgenia Wasserman Pritsker et al. 2017. Assessing the Contribution of Twitter's Textual Information to Graph-Based Recommendation. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI), 2017*.
[45] Kodzo Wegba et al. 2018. Interactive Storytelling for Movie Recommendation through Latent Semantic Analysis. In *Proceedings of the 23rd International Conference on Intelligent User Interfaces (IUI), 2018*.
[46] Thomas D. White et al. 2019. Improving random GUI testing with image-based widget detection. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis, 2019*.
[47] Felix Wu et al. 2019. Simplifying Graph Convolutional Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML), 2019*.
[48] Yingtao Xie et al. 2019. User Interface Code Retrieval: A Novel Visual-Representation-Aware Approach. *IEEE Access* 7 (2019), 162756–162767.
[49] Da Xu et al. 2020. Inductive representation learning on temporal graphs. In *International Conference on Learning Representations (ICLR), 2020*.
[50] Hao Yuan et al. 2020. Explainability in Graph Neural Networks: A Taxonomic Survey. *CoRR* (2020).
[51] Xiaoyi Zhang et al. 2021. Screen Recognition: Creating Accessibility Metadata for Mobile Applications from Pixels. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI), 2021*.
[52] Shengjia Zhao et al. 2019. InfoVAE: Balancing Learning and Inference in Variational Autoencoders. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence, 2019*.