Interactive Link Prediction as a Downstream Task for Foundational GUI Understanding Models

Christoph A. Johns[1,2][0000−0002−4175−340X], Michael Barz[2,3][0000−0001−6730−2466], and Daniel Sonntag[2,3][0000−0002−8857−8709]

[1] Aarhus University, Aarhus, Denmark
cajohns@cs.au.dk
[2] German Research Center for Artificial Intelligence (DFKI), Saarbrücken, Germany
{michael.barz,daniel.sonntag}@dfki.de
[3] Applied Artificial Intelligence, Oldenburg University, Oldenburg, Germany

Abstract. AI models that can recognize and understand the semantics of graphical user interfaces (GUIs) enable a variety of use cases ranging from accessibility to automation. Recent efforts in this domain have pursued the development of a set of foundation models: generic GUI understanding models that can be used off-the-shelf to solve a variety of GUI-related tasks, including ones that they were not trained on. In order to develop such foundation models, meaningful downstream tasks and baselines for GUI-related use cases will be required. In this paper, we present *interactive link prediction* as a downstream task for GUI understanding models and provide baselines as well as testing tools to effectively and efficiently evaluate predictive GUI understanding models. In interactive link prediction, the task is to predict whether tapping on an element on one screen of a mobile application (source element) navigates the user to a second screen (target screen). If this task is solved sufficiently, it can demonstrate an understanding of the relationship between elements and components across screens and enable various applications in GUI design automation and assistance. To encourage and support research on interactive link prediction, this paper contributes (1) a preprocessed large-scale dataset of links in mobile applications (18,830 links from 5,362 applications) derived from the popular RICO dataset, (2) performance baselines from five heuristic-based and two learning-based GUI understanding models, (3) a small-scale dataset of links in mobile GUI prototypes including ratings from an online study with 36 end-users for out-of-sample testing, and (4) a Figma plugin that can leverage link prediction models to automate and assist mobile GUI prototyping.

Keywords: Link prediction · GUI prototyping · Mobile GUIs · Online machine learning.

1 Introduction

The goal of graphical user interface (GUI) understanding is to model the semantics of a GUI by learning effective representations of user interface screens and

their components. These semantic representations can then be used to enable various assistive technologies related to GUI accessibility such as more effective screen readers and voice assistants (e.g., [29, 3]) or related to GUI design and prototyping such as GUI search or recommendation engines and design diagnostic tools (e.g., [23, 25]). Since GUIs are inherently multimodal, layered, and complex, they offer a particular challenge to any single embedding or modeling technique [18]. Recent efforts within the field of GUI understanding have thus moved away from the development of narrow single-purpose models designed to solve a specific GUI understanding task (e.g., icon labeling [3] or tappability prediction [23]) in pursuit of a set of foundation models that can be used off-the-shelf to solve a variety of GUI-related downstream tasks without requiring any or while only requiring minimal finetuning (see, for example, [26, 15, 18]).

The development of such general GUI understanding models requires their evaluation against a range of varied and meaningful downstream tasks. Previous efforts in this domain have mainly focused on single-screen tasks (e.g., summarization [27, 26], captioning [17], or labeling [3]). In this paper, we present *interactive link prediction*, a downstream task that requires an understanding of the relationship between two GUI screens and their components and complements existing downstream tasks related to screen similarity [8] and click prediction [12]. To facilitate the development of GUI understanding models capable of interactive link prediction, we provide a preprocessed large-scale dataset of links in Android applications (18,830 links from 5,362 applications) derived from the popular RICO dataset [7] as well as a custom small dataset of links in mobile GUI prototypes including ratings from an online study with 36 end-users that can be used to test the generalization of link prediction models to out-of-sample data[4]. We further provide performance baselines from five heuristic-based and two learning-based GUI understanding models inspired by recent GUI understanding model architectures. Finally, we present a custom plugin to the GUI design software Figma that can be used to leverage and evaluate link prediction models for design assistance in real-world applications[4].

2 Related Work

GUI understanding techniques and datasets are tools and resources used to analyze and understand the structure and functionality of screens and components of GUI-based applications. These techniques commonly leverage supervised machine learning models trained on large-scale mobile GUI datasets and have proven successful across a wide range of tasks related to link prediction like tappability prediction [25, 29, 23] or screen transition prediction [8]. Schoop *et al.* [23], for example, use the large-scale mobile GUI dataset RICO [7] to train a visual recognition model that predicts the tappability of GUI elements

[4] The datasets and source code for the models are available at https://github.com/christophajohns/link-prediction. The source code for the Figma plugin prototype is available at https://github.com/christophajohns/suggested-links.

in Android applications. Feiz *et al.* [8], similarly, utilize the RICO dataset to train a model that predicts whether a pair of screenshots assumed to be part of the same interaction trace represents the same underlying screen (e.g., representing different scroll positions or showing a notification). Finally, Wang *et al.* [26] utilize prompting of a pre-trained large language model to solve a variety of single-screen GUI understanding tasks including screen summarization and screen question answering. These efforts are emblematic of a recent development in the field of GUI understanding: the pursuit of a set of foundational GUI understanding models that can be used off-the-shelf to solve a variety of interaction tasks in GUI-based applications (see, for instance, [15, 26, 18]). The downstream task presented in this paper, *interactive link prediction*, supports the development of such foundational GUI understanding models by offering a new challenge to evaluate models' multi-screen generalization.

3 Methodology

Interactive link prediction refers to a binary two-screen GUI classification task that is designed to facilitate and test a model's understanding of the relationship between screens and components of a GUI-based application. Given a 3-tuple (S, T, e) of a source screen S, a target screen T, and a source element e from the same application as input, the task is to predict whether a tap gesture (i.e., an interaction with a single touch point) performed on the source element e triggers a transition to the target screen T (see Figure 1). The source screen S and the target screen T are defined as element hierarchies; the source element e is a unique identifier for an element on the source screen S. The goal of the interactive link prediction task is to learn a function f that can accurately predict whether a transition occurs between the source and target screens given the source element:

$$f(S, T, e) = \begin{cases} 1 & \text{if tapping on element } e \text{ on screen } S \text{ triggers a} \\ & \text{transition to screen } T \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

A model f that is able to recognize and predict interactive links should be able to identify semantic relationships between elements and screens across source element types (e.g., icons or buttons) and application contexts (e.g., banking or entertainment) from the screens' visual, structural, and textual information alone. To provide benchmarks for such understanding, we implement, train, and test seven link prediction models, five based on heuristics and two based on supervised learning, on a set of links in Android applications derived from the popular RICO dataset [7] and report accuracy, recall, and F1 scores for all models across the preprocessed dataset.

3.1 Datasets

Our baselines utilize the RICO [7] dataset: a large-scale collection of element hierarchies and user interactions of free Android applications from 2017. Using

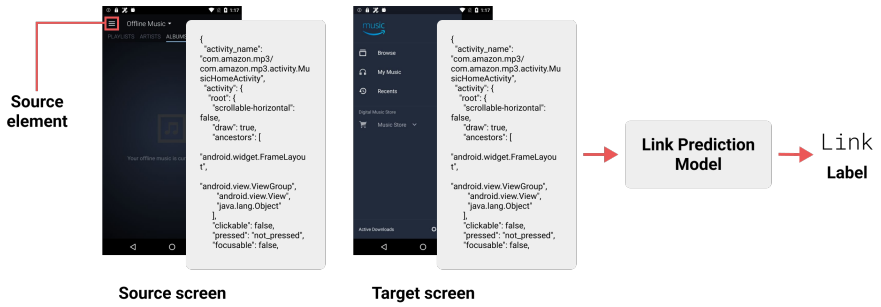

Fig. 1. Given a 3-tuple of a source screen, a target screen and a source element, the task in *interactive link prediction* is to predict whether tapping on the source element triggers a transition to the target screen. The source and target screen are represented as view hierarchies (GUI element trees); the source element is specified by its identifier.

RICO's [7] interaction traces as basis, we derive a sub-dataset called $\mathbf{RICO_{links}}$ that contains data on links in mobile GUIs using heuristic selection. In a first step all traces with at least three screens and at least one tap gesture are filtered, resulting in 5,726 remaining traces. Then, for each tap gesture in those traces, three samples are constructed: (1) a positive sample representing a link from the touched element (i.e., the node lowest in the view hierarchy whose bounds contain the touch point; source element) on a given screen (source screen) to the next screen in the trace (target screen), (2) a negative 'non-source' sample originating from a randomly selected visible[5], static[6] element to the true target screen and (3) a negative 'non-target' sample representing a link from the true source element to a randomly selected screen from the same trace. In total, 18,830 samples are constructed for each of the three sample types, resulting in 56,790 screen pairs across 5,362 applications.

We additionally create a custom small-scale dataset of links in high-fidelity interactive mobile GUI prototypes called $\mathbf{Figma_{links}}$ and conduct an online study with 36 end-users to rate a set of potential links across these prototypes. These data can facilitate the evaluation of a model's generalization to recent visual design styles and from real applications to operating system-independent high-fidelity prototypes. We, first, create a custom set of mock-ups for 4 small applications in the GUI design software Figma spanning 4 screens each and representing common visual styles and application categories, which are chosen following the procedure described in [7]. The view hierarchies of these mock-ups are then extracted using a custom Figma plugin. Next, three trained designers with an academic degree in human-computer interaction who use GUI design

[5] A node within a view hierarchy from the RICO dataset is considered visible if its `visibility` or `visible-to-user` attribute is set to `True` and it has positive width and height as well as at least part of its bounding area is within the screen bounds.
[6] A node with the `clickable` attribute set to `False`.

and prototyping tools at least once a week (all male, age ranging from 25 to 28 years) are recruited and individually tasked to add links to these previously created static application mock-ups using Figma while thinking out loud. Finally, we use the intersection of their three individual link sets (Fleiss' $\kappa = 0.978$) to form the basis for our rating task. In total, 30 designer-created links are selected across the 4 mobile application prototypes and 16 GUI screens. These links are then extended with a random sample of 70 additional potential links across the 4 example applications. This resulting set of 100 potential links is then rated by end-users in an asynchronous online study. 36 adult smartphone users—19 female, 17 male—aged 21 to 64 (mean: 35.3; std: 15.3) are recruited via convenience sampling and instructed to complete a rating task that asks participants to indicate on a five-point Likert scale how strongly they expect that tapping on a given link's source element will open that link's target screen. This data adds further nuance to prediction evaluation and may be used to validate positive predictions that lie outside the set of designer-created links. In total, 100 links are rated by the 36 end-users—including the 30 designer-created links—, across all 4 mobile GUI prototypes and 16 screens (Fleiss' $\kappa = 0.305$).

3.2 Heuristic Models

To provide meaningful baselines for model evaluation, we implement five heuristic models inspired by related work on information foraging theory and information scent (see, for instance, [4]) based on rules about the text in and around the source element and about the text on the target screen. If a model includes a free hyperparameter, its value is tuned using mean F1 score across threefold cross-validation via twenty iterations of Bayesian optimization on a training set derived from the $\text{RICO}_{\text{links}}$ dataset. Our baseline model *PageContainsLabel* is built upon the assumption that a target screen is accurately described by the label of the source elements that link to it [4]. The model suggests adding a link, if the label[7] of the source element contains at least one word from the target screen. All other heuristic models are based on a similar principle. The *LargestTextElementsContainLabel* model suggests adding a link, if the label of the source element contains at least one word of the n largest text elements on the target screen as measured by the elements' area on screen ($n = 20$). It assumes that prominent text elements accurately describe the content of their screen. The *LabelTextSimilarity* model is based on the assumption that semantic text embeddings (i.e., vector representations that aim to encode the meaning of a word or phrase) better capture the semantic similarity of two texts than simple bag-of-words approaches, which treat texts as arbitrarily ordered collections of character sequences and typically rely on co-occurrence as a measure of relatedness. This heuristic model uses the natural language model all-MiniLM-L6-v2 [21], a small-footprint successor to the Sentence-BERT model that was successfully used in the Screen2Vec GUI embedding technique [16]. The model

[7] A label in the $\text{RICO}_{\text{links}}$ dataset is given by any non-empty value of a node's `text` attribute.

suggests adding a link if the cosine similarity between the text embeddings of the source element's label and of text on the target screen exceeds a tuned threshold t (here, $t = 0.122$). The *TextSimilarity* model addresses the issue that models relying on the source element's label are unable to predict links for common GUI elements that do not have a label (e.g., a menu item consisting of an icon and a text element). It suggests adding a link if the cosine similarity between the text embeddings of the source element and of the target screen exceeds a tuned threshold t (here, $t = 0.263$). The text of an element here is defined as its own label or the labels of all its descendants if it does not have a label of its own. The *TextSimilarityNeighbors* model is designed to consider non-text elements (i.e., GUI elements without own or descendant labels; e.g., images or icons) as well as elements whose label is not descriptive of their target's content (e.g., buttons labeled "Click here" or "Read more"). In addition to the source element's own text, its embedding includes the text of the source element's n closest neighboring leaf text elements (here, $n = 2$) as measured by Euclidean distance between their bounding boxes. The model suggests adding a link if the cosine similarity between the embeddings of the text in and around the source element and of the target screen's text exceeds a tuned threshold t (here, $t = 0.292$). This context-based approach is mirrored in both learning-based GUI understanding techniques such as Screen2Vec [16] and in non-learning based techniques like information scent models (e.g., [4]).

3.3 Supervised Learning Models

In addition to the five heuristic models, we implement two supervised learning-based models using binary online learning classifiers. The selection of binary classifier included in each of the two models is detailed in Section 3.4 below. The *TextOnly* model is similar in architecture and implementation to the *TextOnly* model included in the Screen2Vec GUI embedding technique described in [16]. Using the same preprocessing of the *TextSimilarityNeighbors* model, extracting the text in and around the source element and on the target screen and encoding them using the all-MiniLM-L6-v2 language model, the *TextOnly* model labels the concatenated embeddings using a binary random forest classifier trained with random under-sampling. In addition to the number of neighboring leaf text elements to consider n (here, $n = 6$), this model includes two hyperparameters of the random forest classifier: the number of decision trees t (here, $t = 10$) and the size of the random subsets of features to consider f (here, $f = 85$). The *LayoutOnly* model utilizes the layout autoencoder from the original RICO publication [7] to predict links based on the concatenated layout embeddings of the source and target screens as well as a vector denoting the relative position and size of the source element. As was the case with the *TextOnly* model above, a random forest classifier using random under-sampling is used as the classification model with a tuned number of decision trees t (here, $t = 24$) and size of the random subsets of features to consider f (here, $f = 52$). The models are implemented using Python 3.10 and the scikit-learn [20], imbalanced-learn [14], and PyTorch packages [19]. They are trained, tuned and tested using a MacBook Pro with a

2.3 GHz 8-Core Intel Core i9 CPU on a random split of the $\text{RICO}_{\text{links}}$ dataset into a training (80%) and test (20%) set, ensuring similar distribution of source element types (i.e., containing or not containing text), application categories and number of data points per application. The models' hyperparameters are tuned using mean F1 score across threefold cross-validation via twenty iterations of Bayesian optimization on the train set.

3.4 Model Selection

We explore a selection of common binary classifiers from supervised learning literature for the task of link prediction as defined above: random forest, logistic regression, and naive Bayes. These models are chosen due to the beneficial properties of both probabilistic—being able to produce meaningful confidence estimates—and online learning classifiers—being able to continue learning during deployment in interactive systems. To further address the issue of class imbalance in the training set, balanced variants of the classification models and imbalanced learning techniques (i.e., random under-sampling, random over-sampling, SMOTE [2] and ADASYN [11]) are included. We estimate the performance of all models based on their mean F1 score from a threefold cross-validation. We found a random forest classifier with a random under-sampler to be the best performing model for both the *TextOnly* and *LayoutOnly* models, and thus selected it for the final implementation of these two models.

4 Experiments

Using the test set constructed from the $\text{RICO}_{\text{links}}$ dataset, the benchmark models are compared to an additional baseline of a random classifier predicting a positive label with a probability reflecting the frequency of true links in the $\text{RICO}_{\text{links}}$ dataset (i.e., 0.333). While all but the *TextOnly* model outperform this baseline random classifier (see Table 1), our benchmark models' overall level of performance is rather low, indicating the challenge of the prediction task. Among our models, the *LabelTextSimilarity* model is able to most accurately predict the links in the given test set.

Further analysis of the performance of the models by source element type (i.e., text or non-text), all models achieve better prediction performance using only text elements. Comparing the best model's (i.e., the *LabelTextSimilarity* model's) performance across the ten most common application categories in the test set, its performance generalizes well across categories with F1 scores ranging from 0.496 (Entertainment) to 0.603 (Weather). To further illustrate some of the limitations of the *LabelTextSimilarity* model, exemplary data points from the true and false positives with highest heuristic decision score and false negatives with lowest score are displayed in Figure 2. These cases demonstrate some of the shortcomings of the $\text{RICO}_{\text{links}}$ dataset that is used to train and tune the link prediction models. Both misclassifications can be traced back to mislabeled or mismatched data from the original RICO (see false negative) or derived

Table 1. Prediction performance of heuristic and learning-based models for links in Android applications using the $\text{RICO}_{\text{links}}$ test set.

Model	Precision	Recall	F1
LabelTextSimilarity	0.476	**0.629**	**0.542**
PageContainsLabel	0.543	0.450	0.492
LargestTextElementsContainLabel	**0.571**	0.407	0.475
LayoutOnly	0.381	0.584	0.461
TextSimilarity	0.436	0.397	0.416
TextSimilarityNeighbors	0.445	0.358	0.397
TextOnly	0.223	0.301	0.256
(Random (stratified))	0.339	0.337	0.338

$\text{RICO}_{\text{links}}$ datasets (see false positive). For further discussion of the shortcomings of the RICO dataset, we refer to recent reviews in [13], [6], and [28].

When applying the benchmark models to the out-of-sample data in the $\text{Figma}_{\text{links}}$ dataset, the models are found to predict sets of links that strongly differ from those created by human designers. For each of the seven benchmark models, we thus compare the ratings for the predicted links and predicted non-links across the dataset using descriptive statistics. Additionally, we construct a baseline of human performance by comparing the ratings for the designer-created links and the additional random set of links included in the sample.

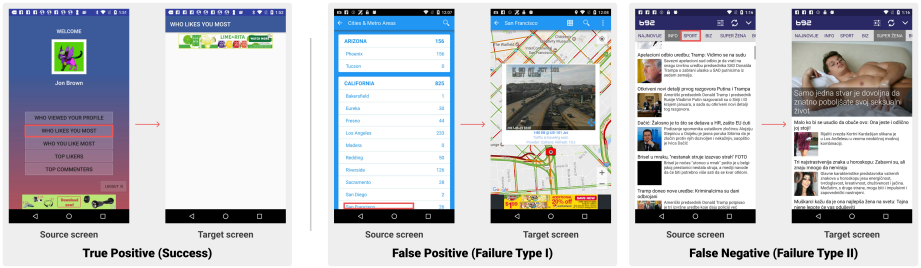

Fig. 2. Selected true positive (left), false positive (middle) and false negative (right) samples with highest/low decision score as determined by the best-performing model in this evaluation. The source element is highlighted with a red frame. While the true positive sample shows the general capability of the model to identify meaningful relations, the false negative (result of a full-screen advertisement interrupting the screen transition) and false positive (mismatched view hierarchy or misreported touch point) samples illustrate limitations of the $\text{RICO}_{\text{links}}$ dataset.

The designer-created links received a mean rating of 4.281 (std: 1.211) from the end-users, indicating a high-quality of the chosen links. The random sample of additional links not chosen by the designers, by comparison, received a mean rating of 1.938 (std: 1.415), resulting in a considerable difference in

mean rating of Δ Mean = 2.342 between the two sets. If a foundational GUI understand models achieved similar discriminative performance to the human designers, the difference in mean rating between their predicted links and predicted non-links should be expected to fall into a similar range of this delta. All benchmark models included in this study fail to achieve a comparable result. The *TextSimilarityNeighbors* model is able to most closely match the designers' difference in mean rating, but still performs considerably worse with Δ Mean = 1.355. Both supervised learning-based models *LayoutOnly* (Δ Mean = -0.263) and *TextOnly* (Δ Mean = -0.357) as well as the heuristic *LabelTextSimilarity* (Δ Mean = -0.508) even produce negative differences where the predicted non-links achieve a higher mean rating than the predicted links.

We further analyze the relationship between the models' decision score (heuristic models) or confidence estimate (supervised learning-based models) and the end-user ratings by comparing the ratings for the ten predicted links with highest ("Top 10") and the ten predicted non-links with lowest decision score ("Bottom 10"). Only the *TextSimilarity* and *TextSimilarityNeighbors* models achieve a comparable difference in mean rating to the designers' sets, further suggesting a limited predictive performance of the benchmark models included in this evaluation. An additional Wilcoxon signed-rank test shows a significant difference in rating between the top ten and bottom ten predictions for all but the *LabelTextSimilarity* and *TextOnly* models (see Table 2). Overall, our baselines illustrate the challenge in interactive link prediction to generalize across element types, application contexts, and available modalities.

Table 2. Comparison between the mean ratings of the ten potential links with highest and lowest decision score (heuristic models) or confidence estimate (supervised learning-based models) per model, including the results of a Wilcoxon signed-rank test. The difference in mean rating between the designer-created and random additional links in the dataset was Δ Mean = 2.342. Note: *: $p < 0.05$; **: $p < 0.01$; ***: $p < 0.001$

Model	Δ Mean Rating	Top 10 (std)	Bottom 10 (std)
TextSimilarity	**2.556*** **	**4.333 (1.143)**	**1.778 (1.371)**
TextSimilarityNeighbors	**2.556*** **	**4.333 (1.143)**	**1.778 (1.371)**
LayoutOnly	0.567***	2.428 (1.663)	1.861 (1.351)
PageContainsLabel	0.336**	3.097 (1.792)	2.761 (1.715)
LargestTextElementsContainLabel	0.336**	3.097 (1.792)	2.761 (1.715)
LabelTextSimilarity	0.111	2.264 (1.604)	2.153 (1.628)
TextOnly	-0.403	2.411 (1.663)	2.814 (1.664)

5 Application: 'Suggested Links' Figma Plugin

To illustrate and facilitate potential uses of GUI understanding models capable of interactive link prediction, we present a prototype for an assistive GUI

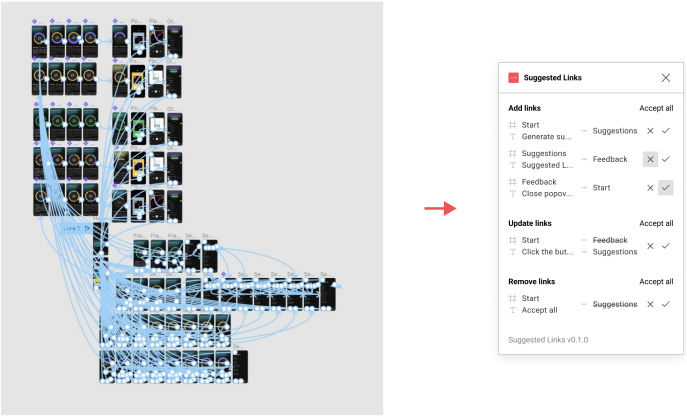

Fig. 3. Screenshot of a prototype in the GUI design application Figma (left). To address the issue of system state visibility, an assistive plugin suggesting links to add, update or remove for a given Figma prototype is implemented (right).

prototyping tool called 'Suggested Links' that can leverage GUI understanding models to support designers of interactive GUI prototypes. A common problem in the development and maintenance of interactive GUI prototypes is that actively developing static GUI designs often leads to cluttered, outdated, and inappropriate links in their corresponding prototypes [9, 10].An example of such a GUI prototype is shown in Figure 3 (left). We envision a system that utilizes capable link prediction models to automatically infer meaningful connections for a given prototype, suggesting them to a designer, and reducing the required effort to develop or maintain that GUI design. Within a research-oriented setting, the same tool could be used to rapidly and interactively diagnose the ability of link prediction models to accurately predict appropriate links in custom prototypes.

To demonstrate the feasibility of creating such a real-time link suggestion system and to furthermore illustrate the promise of GUI understanding models that can learn from user feedback, we develop a plugin to the well-known interface design tool Figma, based on our presented models. The plugin computes the set of all potential links in a given prototype and uses these potential connections as input for a pre-trained link prediction model. The resulting predictions are then compared to the current set of links in the design and grouped into connections that can be added, updated, or removed by the user (see Figure 3, right). If a user decides to accept one of these suggestions, this supervisory signal can be used to update the underlying the link prediction model. We used online learning for our benchmark models in this work to enable such seamless model updates. However, several techniques including Reinforcement Learning from Human Feedback (RLHF) [5] can be imagined that fulfill a similar purpose and enable users to train a shared global model or create a personal one.

The presented prototype is a first encounter in the direction of an AI-supported GUI design process that saves human time and cost when developing new mobile interfaces. However, our predictive benchmark models do not perform well enough to enable suitable link suggestions and, hence, the utility of our prototype is currently very limited. Still, we could demonstrate the feasibility of integrating link suggestions with interactive feedback in a state-of-the-art GUI design tool. To leverage the full potential such a tool, strong GUI understanding models will be required that are capable of accurate link prediction and which can adapt models to individual or project-specific needs. One option to achieve this is to use interactive machine learning techniques as described above.

6 Discussion

We reflect on the pros and cons of our task definition and our provided tools for the development and evaluation of foundational GUI understanding models.

Challenges in Interactive Link Prediction. Interactive link prediction is a challenging task within GUI understanding as it not only requires an understanding of individual screens and their components but of the relationship between components across screens, including non-text elements like images or icons. This is difficult for many of the state-of-the-art GUI understanding models as they commonly focus on either single screens (e.g., [23]) or simple components (e.g., [26]) and thus are likely to struggle with this task by nature of their architecture alone. Link prediction, similarly to other challenging tasks like screen summarization [27, 26] and tappability prediction [23, 25], is furthermore subject to individual differences in perception, requiring effective GUI understanding models to handle ambiguity and contradictions in the training and testing data. The out-of-sample GUI prototype data and its associated end-user ratings provided in this paper may aid research in this issue.

Baseline Model Limitations. The baseline models provided here are limited due to their reliance on unimodal text and layout-based GUI embedding techniques, which may not be as effective in handling non-text source elements or capturing the relationships between multiple GUI elements. They are furthermore trained on a dataset where true links are defined to originate from the lowest element in the screen's element hierarchy that contains a screen transition's touch point. While following convention (cf. [12]), this is in conflict with common GUI design patterns and alternative approaches that have been shown to be effective in tappability prediction (e.g., using the element *highest* in the view hierarchy [25]). As a result, the baseline models' performance appears to generalize poorly to data from high-fidelity prototypes. Future research could explore this issue of generalization and investigate alternative definitions for true links in the RICO dataset. Further, we could investigate the use of computer vision-based techniques (cf. [23, 8]) for improved performance in dealing with non-text elements, as well as of multi-GUI pre-training tasks (cf. [12]) to develop more effective

semantic representations for both real applications and high-fidelity prototypes. Outside the domain of GUI understanding, interactive link prediction models may benefit from taking inspiration from factual link prediction in knowledge graphs (see, for example, [22]). While this direction is not further explored in this work, this body of literature provides relevant insights into alternative problem formulations for multi-screen and multi-component relationship prediction and may aid in the development of new approaches to GUI understanding.

GUI Design Automation and Assistance. Models that can understand and effectively predict semantic relationships across components and screens as required by interactive link prediction hold great promise for design automation and assistance. As our prototypical Figma plugin illustrates, such models could empower designers to spend more time on their high-fidelity prototypes and automate costly maintenance tasks. Beyond this illustration, several use cases can be imagined that enhance existing design automation tools (e.g., [24, 1]) where rough GUI sketches are automatically translated into high-fidelity interactive prototypes, enabling more rapid iterations in early stages of the design process. When link prediction is accurate and efficient, it can lead to time and cost savings in the development process, ultimately resulting in better user experiences.

Towards Foundational GUI Understanding Models. Interactive link prediction points to future research directions in developing more accurate and tailored models that can recognize and understand the relationships between components and screens across parts of an application. As our Figma plugin suggests, adapting models to specific design requirements and personal preferences of designers may be an effective way to create more targeted and efficient design tools. Employing interactive machine learning techniques or other personalization techniques such as RLHF [5] and leveraging additional input modalities may guide the development of foundational GUI understanding models in this vain and further improve the efficiency and effectiveness of the design process.

7 Conclusion

In this work, we proposed a new downstream task for foundational GUI understanding models called *interactive link prediction* and implemented and evaluated seven benchmark models using interactive links from real mobile applications and in high-fidelity GUI prototypes. Our analysis revealed that link prediction is a challenging task that requires understanding not only of various GUI components such as texts, images, and icons but also of their relationship on a single screen and across multiple screens of an application. We discussed the limitations of our presented datasets and benchmark models and further considered interactive machine learning as a methodology to overcome them. Furthermore, we showcased how interactive link suggestions can be integrated in the GUI design process by implementing a plugin for the popular GUI design application Figma. We envision that foundational GUI understanding models that are capable of effective and adaptive link prediction can help to significantly improve

the future GUI design process. We hope that the tools we provide here can encourage and support researchers who want to take up this challenge and develop the next generation of adaptive and flexible GUI understanding models.

Acknowledgements This work was partially funded by the German Federal Ministry of Education and Research (BMBF) under grant number 01IW23002 (No-IDLE), by the Lower Saxony Ministry of Science and Culture, and the Endowed Chair of Applied Artificial Intelligence of the University of Oldenburg.

References

1. Baulé, D., Hauck, J.C.R., Júnior, E.C.V.: Automatic code generation from sketches of mobile applications in end-user development using Deep Learning p. 18 (2021), https://arxiv.org/abs/2103.05704
2. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research **16**, 321–357 (Jun 2002). https://doi.org/10.1613/jair.953, https://www.jair.org/index.php/jair/article/view/10302
3. Chen, J., Swearngin, A., Wu, J., Barik, T., Nichols, J., Zhang, X.: Towards Complete Icon Labeling in Mobile Applications. In: CHI Conference on Human Factors in Computing Systems. pp. 1–14. CHI '22, Association for Computing Machinery, New York, NY, USA (Apr 2022). https://doi.org/10.1145/3491102.3502073, https://doi.org/10.1145/3491102.3502073
4. Chi, E.H., Pirolli, P., Chen, K., Pitkow, J.: Using information scent to model user information needs and actions and the Web. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 490–497. CHI '01, Association for Computing Machinery, New York, NY, USA (Mar 2001). https://doi.org/10.1145/365024.365325, https://doi.org/10.1145/365024.365325
5. Christiano, P.F., Leike, J., Brown, T.B., Martic, M., Legg, S., Amodei, D.: Deep reinforcement learning from human preferences. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 4302–4310. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (Dec 2017)
6. Deka, B., Doosti, B., Huang, F., Franzen, C., Hibschman, J., Afergan, D., Li, Y., Kumar, R., Dong, T., Nichols, J.: An Early Rico Retrospective: Three Years of Uses for a Mobile App Dataset. In: Li, Y., Hilliges, O. (eds.) Artificial Intelligence for Human Computer Interaction: A Modern Approach, pp. 229–256. Human–Computer Interaction Series, Springer International Publishing, Cham (2021). https://doi.org/10.1007/978-3-030-82681-9_8, https://doi.org/10.1007/978-3-030-82681-9_8
7. Deka, B., Huang, Z., Franzen, C., Hibschman, J., Afergan, D., Li, Y., Nichols, J., Kumar, R.: Rico: A Mobile App Dataset for Building Data-Driven Design Applications. In: Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology. pp. 845–854. UIST '17, Association for Computing Machinery, New York, NY, USA (Oct 2017). https://doi.org/10.1145/3126594.3126651, https://doi.org/10.1145/3126594.3126651
8. Feiz, S., Wu, J., Zhang, X., Swearngin, A., Barik, T., Nichols, J.: Understanding Screen Relationships from Screenshots of Smartphone Applications. In: 27th International Conference on Intelligent User Interfaces. pp. 447–458. IUI '22, Association for Computing Machinery, New

York, NY, USA (Mar 2022). https://doi.org/10.1145/3490099.3511109, https://doi.org/10.1145/3490099.3511109
9. Figma Community Forum: Header nav and prototype spaghetti - Ask the community (Mar 2021), https://forum.figma.com/t/header-nav-and-prototype-spaghetti/1534/4
10. Figma Community Forum: How to change several interactions at once? - Ask the community (Nov 2022), https://forum.figma.com/t/how-to-change-several-interactions-at-once/33856
11. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence). pp. 1322–1328 (Jun 2008). https://doi.org/10.1109/IJCNN.2008.4633969, iSSN: 2161-4407
12. He, Z., Sunkara, S., Zang, X., Xu, Y., Liu, L., Wichers, N., Schubiner, G., Lee, R., Chen, J.: ActionBert: Leveraging User Actions for Semantic Understanding of User Interfaces. Proceedings of the AAAI Conference on Artificial Intelligence **35**(7), 5931–5938 (May 2021), https://ojs.aaai.org/index.php/AAAI/article/view/16741, number: 7
13. Leiva, L.A., Hota, A., Oulasvirta, A.: Enrico: A Dataset for Topic Modeling of Mobile UI Designs. In: 22nd International Conference on Human-Computer Interaction with Mobile Devices and Services. pp. 1–4. MobileHCI '20, Association for Computing Machinery, New York, NY, USA (Oct 2020). https://doi.org/10.1145/3406324.3410710, https://doi.org/10.1145/3406324.3410710
14. Lemaître, G., Nogueira, F., Aridas, C.K.: Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. Journal of Machine Learning Research **18**(17), 1–5 (2017), http://jmlr.org/papers/v18/16-365.html
15. Li, G., Li, Y.: Spotlight: Mobile UI Understanding using Vision-Language Models with a Focus (Feb 2023). https://doi.org/10.48550/arXiv.2209.14927, http://arxiv.org/abs/2209.14927, arXiv:2209.14927 [cs]
16. Li, T.J.J., Popowski, L., Mitchell, T., Myers, B.A.: Screen2Vec: Semantic Embedding of GUI Screens and GUI Components. In: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems. pp. 1–15. CHI '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3411764.3445049, https://doi.org/10.1145/3411764.3445049
17. Li, Y., Li, G., He, L., Zheng, J., Li, H., Guan, Z.: Widget Captioning: Generating Natural Language Description for Mobile User Interface Elements. arXiv:2010.04295 [cs] (Oct 2020), http://arxiv.org/abs/2010.04295, arXiv:2010.04295
18. Li, Y., Li, G., Zhou, X., Dehghani, M., Gritsenko, A.: VUT: Versatile UI Transformer for Multi-Modal Multi-Task User Interface Modeling (Dec 2021). https://doi.org/10.48550/arXiv.2112.05692, http://arxiv.org/abs/2112.05692, arXiv:2112.05692 [cs]
19. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Wallach, H., Larochelle, H., Beygelzimer, A., Alché-Buc, F.d., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32. Curran Associates, Inc. (2019),

https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
21. Reimers, N.: Hugging Face - sentence-transformers/all-MiniLM-L6-v2 (Aug 2021), https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
22. Rossi, A., Barbosa, D., Firmani, D., Matinata, A., Merialdo, P.: Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. ACM Transactions on Knowledge Discovery from Data **15**(2), 1–49 (Apr 2021). https://doi.org/10.1145/3424672, https://dl.acm.org/doi/10.1145/3424672
23. Schoop, E., Zhou, X., Li, G., Chen, Z., Hartmann, B., Li, Y.: Predicting and Explaining Mobile UI Tappability with Vision Modeling and Saliency Analysis. In: CHI Conference on Human Factors in Computing Systems. pp. 1–21. ACM, New Orleans LA USA (Apr 2022). https://doi.org/10.1145/3491102.3517497, https://dl.acm.org/doi/10.1145/3491102.3517497
24. Suleri, S., Sermuga Pandian, V.P., Shishkovets, S., Jarke, M.: Eve: A Sketch-based Software Prototyping Workbench. In: Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems. pp. 1–6. CHI EA '19, Association for Computing Machinery, New York, NY, USA (May 2019). https://doi.org/10.1145/3290607.3312994, https://doi.org/10.1145/3290607.3312994
25. Swearngin, A., Li, Y.: Modeling Mobile Interface Tappability Using Crowdsourcing and Deep Learning. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. pp. 1–11. CHI '19, Association for Computing Machinery, New York, NY, USA (May 2019). https://doi.org/10.1145/3290605.3300305, https://doi.org/10.1145/3290605.3300305
26. Wang, B., Li, G., Li, Y.: Enabling Conversational Interaction with Mobile UI using Large Language Models. In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. pp. 1–17. ACM, Hamburg Germany (Apr 2023). https://doi.org/10.1145/3544548.3580895, https://dl.acm.org/doi/10.1145/3544548.3580895
27. Wang, B., Li, G., Zhou, X., Chen, Z., Grossman, T., Li, Y.: Screen2Words: Automatic Mobile UI Summarization with Multimodal Learning. In: The 34th Annual ACM Symposium on User Interface Software and Technology. pp. 498–510. ACM, Virtual Event USA (Oct 2021). https://doi.org/10.1145/3472749.3474765, https://dl.acm.org/doi/10.1145/3472749.3474765
28. Wu, J., Wang, S., Shen, S., Peng, Y.H., Nichols, J., Bigham, J.P.: WebUI: A Dataset for Enhancing Visual UI Understanding with Web Semantics. In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. pp. 1–14. ACM, Hamburg Germany (Apr 2023). https://doi.org/10.1145/3544548.3581158, https://dl.acm.org/doi/10.1145/3544548.3581158
29. Zhang, X., de Greef, L., Swearngin, A., White, S., Murray, K., Yu, L., Shan, Q., Nichols, J., Wu, J., Fleizach, C., Everitt, A., Bigham, J.P.: Screen Recognition: Creating Accessibility Metadata for Mobile Applications from Pixels. In: Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems. pp. 1–15. ACM, Yokohama Japan (May 2021). https://doi.org/10.1145/3411764.3445186, https://dl.acm.org/doi/10.1145/3411764.3445186