This is a preprint for personal use only. The published paper may be subject to some form of copyright.

Describing UI Screenshots in Natural Language

LUIS A. LEIVA, University of Luxembourg, Luxembourg
ASUTOSH HOTA and ANTTI OULASVIRTA, Aalto University, Finland

Being able to describe any user interface (UI) screenshot in natural language can promote understanding of the main purpose of the UI, yet currently it cannot be accomplished with state-of-the-art captioning systems. We introduce XUI, a novel method inspired by the global precedence effect to create informative descriptions of UIs, starting with an overview and then providing fine-grained descriptions about the most salient elements. XUI builds upon computational models for topic classification, visual saliency prediction, and natural language generation. XUI provides descriptions with up to three different granularity levels that, together, describe what is in the interface and what the user can do with it. We found that XUI descriptions are highly readable, are perceived to accurately describe the UI, and score similarly to human-generated UI descriptions. XUI is available as open source software.

CCS Concepts: • **Computing methodologies** → **Natural language generation**; *Probabilistic reasoning*; • **Human-centered computing** → Interaction design process and methods.

Additional Key Words and Phrases: Captioning; Visual Saliency; Deep Learning; Natural Language Processing

ACM Reference Format:
Luis A. Leiva, Asutosh Hota, and Antti Oulasvirta. 2022. Describing UI Screenshots in Natural Language. *ACM Trans. Intell. Syst. Technol.* 1, 1, Article 1 (January 2022), 28 pages. https://doi.org/10.1145/3564702

1 INTRODUCTION

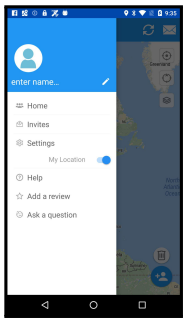
Being able to describe any user interface (UI) screenshot in natural language can promote understanding of the main purpose of the UI, so there is quite a range of applications and user groups that can benefit from such descriptions. First-time users, for example, may struggle to navigate the UI, therefore they could be offered an introductory help message describing the main UI parts or an overview of the overall functions before allowing the user to dive in and explore. Similarly, textual descriptions of UI screenshots can be incorporated as ALT text on websites and app marketplaces, thereby providing semantic information that can help search engines to better rank the app and support accessibility, for example, for users with vision impairments. Technical documentation, such as user manuals or API specifications, may improve understandability by including relevant descriptions of their screenshots, automatically. Finally, one could argue that artificial intelligence (AI) systems also need some human-like understanding of the UI. In this regard, natural language descriptions could be used by interactive AI systems to help users for example by tutoring them or suggesting them how to explore the UI.

Pareddy et al. [61] noticed that screenshots are frequently shared on social media and in academic papers, however screenshot tools strip away semantics useful for understanding the UI contents, leaving only pixels. Optical character recognition (OCR) technology is not enough for describing

Authors' addresses: Luis A. Leiva, University of Luxembourg, Luxembourg, name.surname@uni.lu; Asutosh Hota; Antti Oulasvirta, Aalto University, Finland, name.surname@aalto.fi.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2022 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2157-6904/2022/1-ART1
https://doi.org/10.1145/3564702

This is a preprint for personal use only. The published paper may be subject to some form of copyright.



XUI Caption: *'A menu screen with an icon component at the top-left part.'*
XUI Simple: *'This screen is a menu app with an icon at the top-left area of the screen. You can see a list of selectable options.'*
XUI Detailed: *'The interface must be a menu app. It shows a list of selectable options. You may notice an icon ubicated at the top-left part.'*
Microsoft Cognitive Services: *'A screenshot of a cell phone screen with text.'*

Fig. 1. Examples of the actual descriptions provided by our method, from less to more verbosity about the layout design topic and the most salient element. Current state-of-the-art image captioning systems, such as the Microsoft Cognitive Services API (bottom example), are unable to provide a meaningful UI description. See Figure 11 for additional outputs and the Appendix for outputs by other image captioning systems.

the UI, since it does not capture semantics about the structure of the UI. While text is an important component in UIs, not all UIs comprise worthy text for a usable description. Furthermore, OCR does not work if the UI has no text.

Thanks to recent advances in AI research, and more concretely in deep learning (DL) based captioning, computers can describe objects, people, and scenery in images [17, 18, 66, 77]. However, current automatic image captioning tools are far from being usable in the context of UIs, since these tools are trained on natural images, which are quite different from UIs. For example, UIs are typically highly structured, colorful, make heavy use of icons and text, present navigation elements, buttons, etc. In addition, DL models trained on natural images are strongly biased towards recognising textures rather than shapes [23]. Furthermore, DL captioning approaches often require a large amount of training data and still produce unpredictable results that are difficult to control in terms of phrasing or content [46, 78]. Early work proposed to rely on human crowdworking to provide accurate descriptions [6], but in practice it is both expensive and burdensome [7]. So currently either we use an imperfect DL model and accept that UI captions will be hardly useful, or we have to pay crowdworkers and forget about generating captions in real-time.

In this article, we propose XUI, a novel method to automatically generate informative descriptions of a given UI screenshot without having to rely on human-authored text nor DL based captioning models. For practical reasons (Section 4), we will focus on mobile UIs to showcase the value of our method. Note that XUI does not only generate captions, but also describes what is in the interface and what the user can do with it. In other words, what role it plays for interaction and what it affords. This, we believe, is a more natural and information-efficient approach to describing a UI than simply listing elements on it. To achieve this, XUI builds upon computational models for layout design topic classification and element saliency prediction ("what to say") and a template-based natural language generation engine ("how to say it"). Our approach considers a declarative representation that provides full control over the generated text, ranging from string manipulation rules, nested calls to other templates, to conditional expressions within a template to determine how a text should be generated based on the values defined in template slots.

XUI is inspired by the global precedence effect [58] from human perception: it begins with an overview of the screen with defining higher-level features, and then 'zooms in' to individual elements in a selective fashion. Global precedence is closely related to the Gestalt principles of grouping [76], in that humans naturally perceive objects as organized patterns and objects. As observed in Figure 1, XUI can describe a UI screenshot in different ways, providing from less to more verbose information. In 'caption' mode, XUI provides a succinct description of the UI,

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

informing about the class of interface layout, or *topic*, and the most salient UI element. In 'simple' mode, XUI provides an additional sentence that describes further the topic, thereby providing more context to the user about the overall purpose of the UI. Finally, in 'detailed' mode XUI provides a systematic set of short sentences informing about: topic classification, topic description, salient element description, and, if the element is interactive, an additional sentence informing that the element can be clicked (or, in the context of mobile UIs, *tapped*). More examples of the descriptions delivered by XUI are provided in Figure 11.

The development of better UI descriptions is not only a machine learning problem, but poses an exciting problem for human-computer interaction researchers. Some exciting research questions we aimed to answer are the following: Can we generate UI descriptions without training datasets? Can we control such descriptions in terms of phrasing or content? Are such generated descriptions readable? Are they perceived as useful by end users? Eventually, the key question is how to generate automatic descriptions that quickly but accurately provide the 'gist' of the UI. To approach this question, we need to understand what people 'see' in interfaces, how they perceive the information therein, and how they represent it conceptually. The contributions of this article are summarized as follows:

- A data-driven method to create informative descriptions of UI screenshots in natural language.
- Validation of the method via objective and subjective evaluations.
- A set of reusable software libraries and computational models, to facilitate further research and applications.

2 RELATED WORK

Producing a human-interpretable outcome for a given machine learning model outcome is a very active and a relatively new research field. Comprehensive surveys are provided by Miller [54] and Adadi et al. [1]. XUI is mainly informed by three key areas in computer science: image captioning, natural language generation, and reverse-engineering of UIs. In the following we discuss how these have informed the design of XUI.

2.1 Image Captioning

Captioning tools have the potential to empower end-users to know more about any image without having to rely on human-authored text, which is time consuming and burdensome to produce [7, 24, 55]. However, automatically generating a natural language description of an image is a hard problem. It has recently received a lot of attention not only in Computer Vision but also in Natural Language Processing and Machine Learning communities, yet it still remains as a grand challenge [82]. A survey paper by Hossain et al. [28] agglutinates recent research in this regard.

Modern approaches to image captioning and scene understanding rely on a DL based encoding-decoding framework: first a convolutional neural net identifies key concepts in the image, then a recurrent neural net generates the text for the identified concepts. It is extremely hard for these models to generate rich descriptions, because vanishing gradients often occur in the decoding process [25] and there is an exposure bias between training and testing [49, 67]. Finally, as neural systems begin to move toward generating longer outputs in response to longer and more complicated inputs, the generated texts begin to display reference errors, sentence incoherence, and a lack of fidelity to the source material [78]. Among these, the lack of *compositionality* (i.e., producing syntactically correct but semantically irrelevant language structures) is still considered particularly challeging and an open issue [15].

UI captioning with end-to-end supervised learning models is further challenged by the fact that currently there is no dataset that includes UI screenshots and associated ground-truth descriptions,

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

authored by humans. The reference datasets for benchmarking image captioning systems are MS-COCO [47] and Flickr30K [62]. They comprise mostly natural images, which are quite different from UIs (see Section 1). Other popular datasets include restaurant reviews [59], weather reports [46], and author biographies [38], which cannot be leveraged to generate UI descriptions either, because the data belong to a completely different domain. The recent Widget Captioning dataset [45] includes natural language descriptions of individual UI elements, which is also insufficient to generate descriptions for full screenshots.

2.2 Data-to-text Generation

Data-to-text (or table-to-text) generation is a subfield of Natural Language Generation (NLG) aimed at translating structured data into unstructured text [22, 68]. A core task of NLG is to generate textual descriptions of knowledge base records. A common approach is to use hand-engineered templates [52, 53] but there has also been interest in creating templates in an automated manner [4, 19, 34, 43]. Recently, the NLG community has investigated the concept of "neural templates" [30, 79], where a machine learning model learns template-like latent objects (discrete structures) for conditional text generation. Generative models have produced impressive results in NLG, however they are still unpredictable, largely uninterpretable, and difficult to control in terms of their phrasing or content, resulting in linguistic mistakes.

Morash et al. [55] proposed to generate image descriptions by slotting worker answers into a template: a set of predesigned sentences and a table with appropriate "blanks" to be filled in by worker answers. Additionally, worker answers could determine whether certain sentences are included or omitted from the template's text description, and determine the number of rows and columns in the table description. XUI uses a similar template-based approach, but no human intervention is required.

Finally, Kulkarni et al. [37] developed a system to automatically generate natural language descriptions from images that exploits both data-driven statistics from larg-scale text analysis and Computer Vision. The system used object detection, modifiers (adjectives), and spatial relationships (prepositions) to generate descriptions. The main limitations are that (1) it is difficult to enforce grammatically correct sentences and (2) the system ignores discourse structure (coherency among sentences), as each sentence in a description is generated independently of the previous one.

2.3 Reverse Engineering of UIs

Another line of work loosely related to XUI is that of reverse engineering tools that can repurpose a UI or build upon it. For example, systems like Graphstract [31] and HILC [32] can quickly summarize the steps required to complete a tutorial by taking a screenshot and masking non-relevant parts. This kind of "visual tutorials" has been shown effective for creating contextual help [81] and could be considered an alternative form of describing a UI. However, creating graphically illustrated documentation can be time consuming and expensive. Further, natural language descriptions can be easily indexed, searched, copied, and translated.

Moriyon et al. [56] proposed a technique to generate hypertext help messages from a UI design model, and Pangoli et al. [60] demonstrated how to generate task-oriented help from user interface specification. However, these generators assume the knowledge of the UI design model, which requires expert designers. Sikuli [80] addressed this shortcoming by allowing end-users to search UI documentation using screenshots as input. Rather than describing UIs, Sikuli is more focused on visual scripting, which is outside the scope of this article.

Waken [5] leverages computer vision techniques to extract usage information from tutorial videos. It can identify UI elements such as icons, menus, and tooltips. However, Waken can only recognize UI elements the user has interacted with; i.e., if an icon is never clicked, or a tooltip is

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

never shown, then it will not be recognized. Further, UI elements must provide some visual hint informing about their affordability; for example, the recognition of clickable elements requires those elements to become highlighted when the cursor enters, and to provide an additional highlight when the element is clicked.

Prefab Layers and Annotations [14] help developers write interpretation logic from UI screenshots, and can be used to label UI elements with metadata needed to enable runtime enhancements. Interestingly, Prefab can reconstruct a UI hierarchy from raw pixels. Unfortunately, it relies on a well-defined widget library, with predictable visual styles that can be recognized using pattern matching [13]. This is critical because not all interfaces are created entirely from standard widgets [12]. This is especially true for websites and mobile UIs.

Finally, in line with this research topic, previous work has been directed at making UIs more accessible. X-Ray [61] stores the view hierarchy tree of a UI as Exif tags when taking a screenshot, allowing to preserve some semantics that can be recovered later by e.g. screen readers. Screen Recognition [84] is an on-device object detection model that extracts UI elements from raw pixels in a screenshot and creates accessibility metadata, such as navigation hints for screen readers. While these works are not really focused on describing UIs using natural language, they nevertheless highlight the importance of being able to annotate the UI semantics, even if using an alternative communication medium such as Exif tags and metadata.

3 THE XUI METHOD

Essentially, XUI focuses on both content selection ("what to say") and surface realization ("how to say it"). Figure 2 provides an overview of the processes and computations involved. Content selection is determined by computational models for topic classification and visual saliency prediction, whereas surface realization is achieved by means of a template-based NLG engine. We assume that both visual and structural information about the UI is given, though XUI can generate descriptions using visual information only, as hinted in the previous section and discussed further in Section 8. The structural information is any suitable representation of the *visible* UI elements (type, position, and size), since the most salient pixels are mapped to actual UI elements. In this article, this structural information will be a *semantic wireframe* image (see Section 4 for more details) In the following we describe the three key components of the processing pipeline.

3.1 Topic Classification

The main key component of XUI is a classifier that predicts the type of screen design; e.g. login, news, chat (see Figure 6 for a complete list). The classifier is a deep convolutional network (ConvNet) that processes visual information from the UI screenshot, denoted as ConvNet_v in Figure 2, which is complemented with the output from another ConvNet that processes the associated structural information, denoted as ConvNet_s in Figure 2. Then, a softmax layer selects the most probable UI topic and the corresponding template is populated with information about the topic, to be explained later.

On the one hand, the architecture of the ConvNet that processes the visual information (i.e. the UI screenshot) is similar to the popular VGG16 architecture [73]. We chose this architecture as a starting reference because it promotes small receptive fields and a homogeneous architecture. One of the most important design decisions in CNN architectures is the size of the receptive fields [50]. We use adaptive filters of size 3×3, which make it possible for different hidden neurons to become highly specialized in specific regions of the input screenshot.

On the other hand, the architecture of the ConvNet that processes the structural information (i.e. the semantic wireframe) is a deep convolutional autoencoder similar to Leiva et al. [39], where the bottleneck layer of the encoder network creates a latent vector, also known as *semantic embedding*,

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

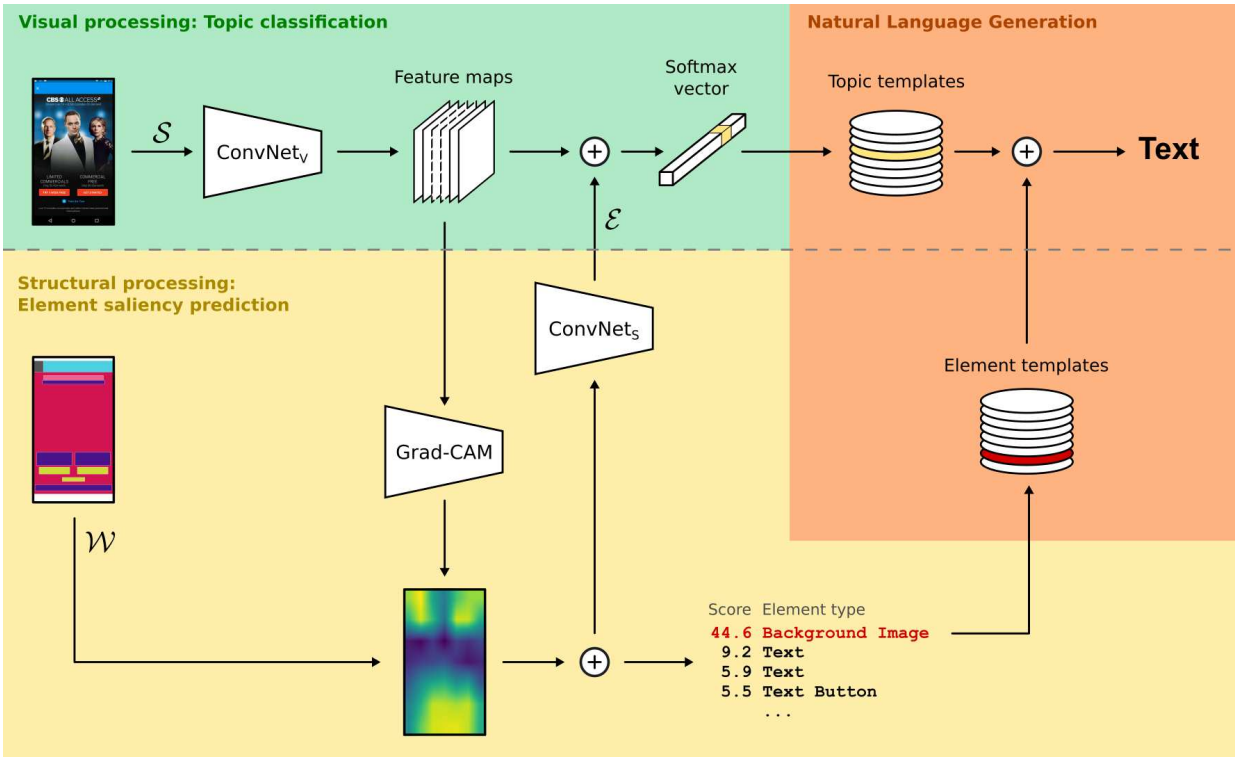


Fig. 2. XUI processing pipeline. Our solution dynamically extends template-based natural language generation with general information about the UI (visual processing part, top) and particular information about the most salient element (structural processing part, bottom). XUI takes as input a UI screenshot and its semantic wireframe to decide what is the main purpose of the UI and what are the most salient elements. This information is then processed by a template-based NLG engine.

that is concatenated with the feature maps computed by the visual ConvNet. Such a bottleneck layer has a dimensionality of 32×16×32 and all convolutional layers use a small receptive field of size 3×3. We experimented with other models and input representations, and the combination of these two ConvNets delivered the most promising results; see Section 5.

Following the example from Figure 1, at this point XUI can generate descriptions like *'A tutorial app.'*, *'This must be a tutorial screen.'*, or *'This is a tutorial app. This screen has information about the functionality of the app.'*. In the latter case, the additional sentence is derived from a knowledge base. Notice that structural information is not embedded yet into the textual description, since at this point it is only used to improve classification performance. In the next section we describe how the UI structure is leveraged further in the final textual description.

3.2 Element Saliency Prediction

Another key component of XUI is a saliency model[1] that estimates which UI elements are more visually important. Such importance is computed as the number of "fixations" (convolutional attention maps) that a UI element would receive, informed by the Gradient-weighted Class Activation Mapping (Grad-CAM) [70]. Essentially, Grad-CAM uses the feature maps from the last convolutional layer of our visual ConvNet (Figure 2) and produce a coarse localization map L (i.e. the saliency map) highlighting important regions in the image that led to the UI topic classification. Formally, Grad-CAM is computed as a linear combination of the feature map activations of any

[1]Saliency refers to the ability of an object to attract visual attention in a scene.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

convolutional layer A^k:

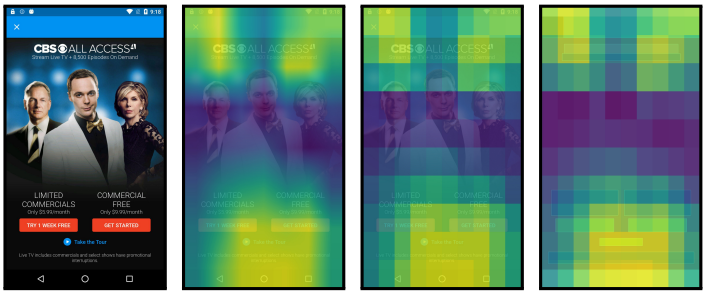
$$L = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right) \tag{1}$$

with

$$\alpha_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k} \tag{2}$$

where α_k^c is the k-th neuron weight for class c, Z is the number of pixels in the feature map, with i and j denoting the index of width and height dimensions, and y^c is the predicted UI topic. The ReLU function is applied in Equation 1 to remove negative gradient values, since we are only interested in the features that have a positive influence on the UI topic of interest.

The resulting saliency map is *discretized* to speed up computations using a fixed bin size of 10, proportional to the screenshot width and height; see Figure 3. Discretization is achieved in two steps: first downsampling the map (up to the aforementioned bin size) using bilinear interpolation and then upsampling it using nearest-neighbour interpolation. Then each UI element is assigned a saliency score, proportional to the number of intersections between each discrete region and the element's bounding box. The top-ranked element is selected and the corresponding template is populated with information about the element, such as type, size, and position. This process is illustrated in the bottom part of Figure 2.



(a) (b) (c) (d)

Score	Element type
44.60	Background Image
9.28	Text
5.95	Text
5.52	Text Button
4.95	Text
4.89	Text Button
3.25	Image
3.25	Text
2.81	Icon
1.22	Text Button

Fig. 3. The UI screenshot (a) is processed with our visual ConvNet, and the Grad-CAM of its last convolutional layer produces a saliency map (b), which is discretized (c) and aligned with the semantic wireframe (d) to rank each element according to the attended regions of the saliency map.

We should note that other computational models could be used to process the screenshots and later compute the Grad-CAM saliency maps, since our pipeline is quite modular, however reusing the feature maps from our visual ConvNet not only is more accurate (see Section 5) but also allows for a more coherent architecture overall.

3.3 Natural Language Generation

The last key component of XUI is an NLG engine that composes the final UI description, by combining the visual and structural information processed so far. Given that our focus is on generating informative textual descriptions, the engine operates on top of a knowledge base. Following standard notation [4, 46, 78], let $\boldsymbol{x} = \{r_1 \dots r_J\}$ be a UI knowledge base, or collection of UI records. A record is made up of a UI topic (`r.t`) or domain, a UI element (`r.e`) or entity, and one or more values (`r.v`). In our case, a UI knowledge base might have a record with `r.t = "login interface"`, `r.e = "button"`, and `r.v = ["sign up", "sign in"]`. Currently XUI comprises

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

$|t| = 20$ UI topics and $|e| = 23$ UI elements. The aim is to generate an adequate text description $\hat{y}_{1:T} = \hat{y}_1 \dots \hat{y}_T$ of each $r_j \in \boldsymbol{x}$. However, since there is no public dataset with $(r_j, y_{1:T})$ pairs worth of training data, it is not possible to learn a plausible data distribution via maximum likelihood estimation. Therefore, we use instead a domain-independent template-based engine acting on local decisions.

Our approach to generating UI descriptions exploits the so-called global precedence effect [58] (see Kimchi [33] for a review), and begins with an overview of the screen with defining higher-level features, and then 'zooms in' to individual elements in a selective fashion. As shown in Figure 1, following the aforementioned global precedence effect, our NLG engine first emits a sentence describing the UI topic and its main purpose, and then emits one or more sentences exemplifying some interaction that can be performed with the most salient UI element.

Our NLG engine is similar in soul to previous work [4, 53], where templates are defined by a knowledge expert to capture the range of syntactic structures that are needed. However, unlike previous work, our template system is probabilistic and straightforward to maintain. Currently XUI has three main templates (caption, simple, detailed) and five sub-templates (opener, location, action, confidence, component). Table 1 and Table 2 show some examples of such sub-templates, respectively. The three main templates use a combination of these sub-templates. For example, the template of XUI caption is "a {topic} {what} {with} a {size} {element} {location}", where each slot {·} is filled in at runtime. This is the simplest template, as it has no opener text, no confidence-based verbs, nor an element description. Note that "confidence" is a construct derived from the probability distribution of the topic classifier, as explained later, so different phrasings can be provided to the user accordingly; see Figure 4.

Each template is composed of two main parts: template slots and template rules. Template slots are parameters or variables that are filled with values at runtime. Template rules are declarative statements that define how inputs to the template should be realized as text. A breakdown example of the resulting realization is shown in Figure 4. The text realization steps include resolving concordances (e.g. a/an determinants) and other linguistic conventions such as ensuring the first letter at the beginning of every sentence is capitalized and that all sentences end with a dot.

Type	**Template**
Caption	a {topic} {what} {with} a [{size}] {element} {*location*}
Simple	[{*opener*}] a {topic} {what} {with} a {size} {element} {*location*}. ↷ {has} {class}
Detailed	{*opener*} a {topic} {screen}. {has} {class}. ↷ {exists} a {size} {element} {*location*} {*action*}. ↷ [{*component*}]

Table 1. Examples of XUI main templates. Entities in braces {·} denote slots to be filled in at runtime. Entities in square brackets [·] denote optional realizations. Sub-templates are denoted in *italic* font. The ↷ symbol denotes sentence concatenation.

'The interface must be a tutorial screen. It has introductory information about the app. You may notice a large image placed at the center part.'

Fig. 4. Breakdown of the 'detailed' description shown in Figure 1, informing about confidence, overall topic and topic description, sizing, positioning, and salient element.

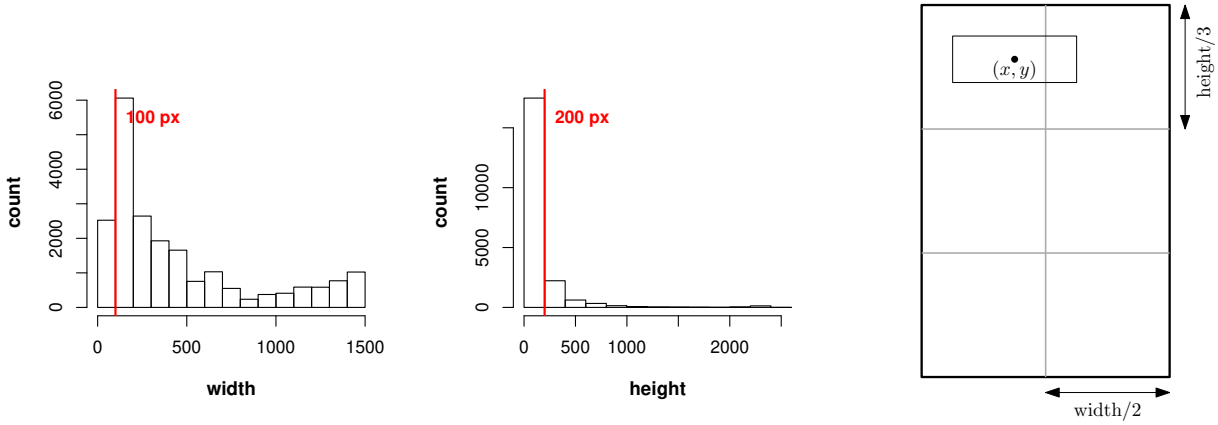
This is a preprint for personal use only. The published paper may be subject to some form of copyright.

Type	Template	Realization example
Opener	{it} {what} {verb}	*'this interface is'*
Location	{verb} at the {position} {action}	*'placed at the center of the screen'*
Action	{conj} you can {do}	*'which you can go and click'*
Confidence	{verb}	*'could be'*
Component	a {type} {description}	*'a web view displays website contents'*

Table 2. Examples of XUI sub-templates. Entities in braces {·} denote slots to be filled in at runtime. The three main templates (caption, simple, detailed) use a combination of these sub-templates. The 'confidence' sub-template is actually a reusable sub-sub-template. Note: Realization examples here are not final (e.g. no capitalization rules are incorporated).

An example of the flexibility of XUI is illustrated as follows. The topic classification probability informs about the classification confidence of the model, which is leveraged to create different verbs accordingly. For example, when the probability distribution is flat, the description begins with *'This might be ...'* or *'This could be ...'*. Conversely, when the probability distribution is sharp, the description begins with *'This is ...'* or *'This must be ...'*. To decide how confident is our topic classifier, we compute the kurtosis of the softmax vector for any prediction. Kurtosis is a measure of whether a distribution is heavy-tailed (high confidence) or light-tailed (low confidence).

Similarly, XUI derives information about sizing and positioning by inspecting the structural properties of the UI elements. In the former case, a previous analysis of the width and height distributions helped us determine how 'small' and 'large' elements are considered; see Figure 5(a). In the later case, the UI screen is divided in a 6-cell grid and the centroid of the element's bounding box will determine if the object is positioned at the top, bottom, center, left, or right part of the screen; see Figure 5(b). Note that these rules are quite flexible and developers could change the values chosen here. We should point out that XUI templates, while currently tailored to mobile UIs, they are agnostic to the mobile Operating System. This means that XUI can work with Android or iOS screenshots, for example. We will release our software upon publication of this article so that others can inspect our implementation in detail.



(a) Element size distributions (b) Element position

Fig. 5. The distribution of element sizes (a) determines which elements should be considered 'small' (width > 100 px) or 'large' (width > 500 px and height > 200 px). The position of any element is described according to its bounding box coordinate with regards to a 6-cell grid (b).

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

4 UI DATASET

XUI relies on appropriate structural information about the UI to produce compelling descriptions, so it is important to ensure that this information is of high-quality. Such a UI structure does not need to be the same as the actual, logical UI structure, as long as it indicates the visible UI elements in a machine-readable format.

In this article, we use the Enrico dataset [39], a curated version of Rico [10]. Enrico comprises 1460 mobile user interfaces categorized according to a design taxonomy of 20 UI layout categories; e.g. news, login, settings, tutorial, etc. See Figure 6 for some examples. Each user interface comprises a screenshot with associated metadata such as annotations of element types, visual and structural data, and interactive design properties.

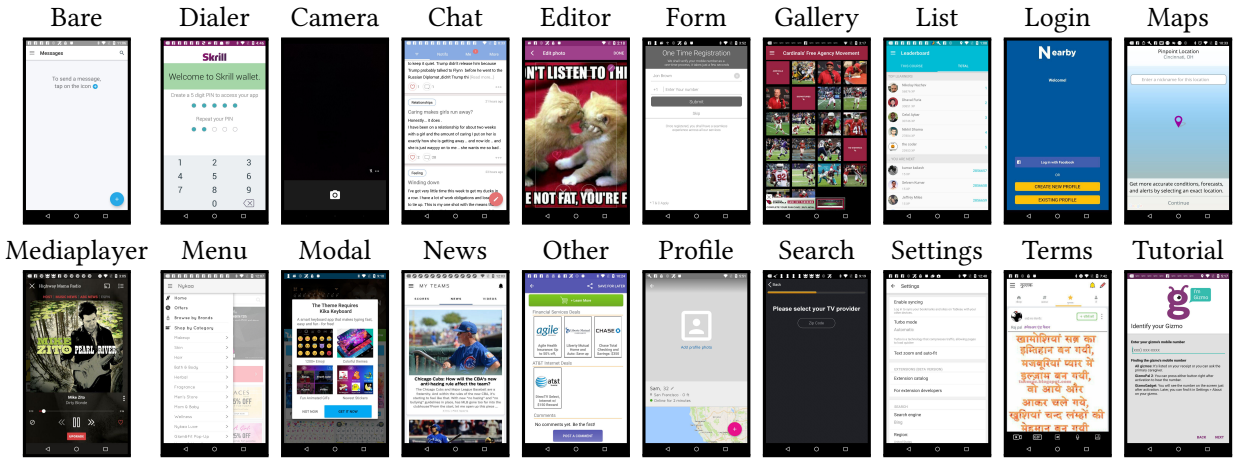


Fig. 6. Examples of Enrico categories. The "Other" category is considered a "rejection" or out-of-distribution class; useful to categorize unkown UIs or UIs that have not been yet defined.

Enrico provides semantic annotations for more than 30K UI elements, classified into 25 categories, such as icon, button, text, navigation, etc. These annotations include the structural and functional roles that UI elements play in the screen design [48]. Finally, Enrico also provides semantic wireframes, where each UI element is rendered as a bounding box with an associated color code; see Figure 7.

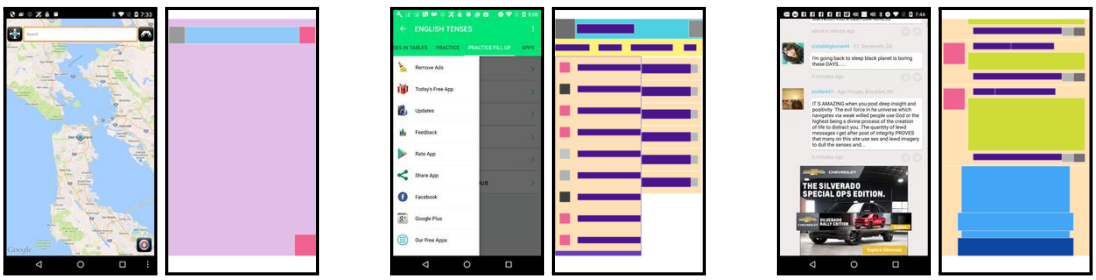


Fig. 7. Examples of Enrico UIs, shown here together with their associated semantic wireframes. Note that the latter are not classic UI wireframes, but structural hints about the UI contents, where each color denotes a different UI component.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

5 PERFORMANCE EVALUATION

In this section we evaluate the machine learning modules of XUI individually (UI topic classification and element saliency prediction), to inform about their practical performance. On the one hand, the identified UI topic will guide the final description at the *overview* level. On the other hand, the identified salient element will guide the final UI description at the *detail* level. In the next section we evaluate XUI as a whole, including its NLG capabilities, via a formal user study followed by a readability analysis in a later section.

5.1 Topic Classification

We test different data representations and up to seven computational models to handle those: three basic models and their four combinations. Here we provide an overview of these models. The *Supplementary Materials* on our companion website describe each model in detail. These experiments helped us to decide the final model for visual processing, discussed in Section 3.

5.1.1 Models. The first basic model is a ConvNet that processes UI screenshots, for which we use a simplified version of the VGG16 architecture [73], with smaller receptive fields, batch normalization and 0.2 dropout after each convolutional block, and only one fully connected layer before the output softmax layer. The second basic model is a ConvNet that processes UI wireframes, which are much simpler than screenshots, for which we use the same architecture of the visual ConvNet but without batch normalization and fewer convolutional blocks. The third basic model is a deep convolutional autoencoder that processes UI wireframes, with a 32×16×32 bottleneck layer followed by a softmax layer. The decoder network is not needed for topic classification, but is very useful to verify the quality of the resulting UI embeddings. As previously hinted in Section 3.1, a UI embedding is the latent vector created by the bottleneck layer of the encoder network. Finally, the combined models use two (or all) basic models as input (see Table 3), by concatenating their outputs and connecting two fully connected layers followed by the output softmax layer.

5.1.2 Data and Optimizer. The screenshot and wireframe images are encoded as RGB vectors of 256×128 px resolution, to speed up training. We train all models with the popular Adam optimizer using a learning rate $\eta = 0.001$ and decay rates $\beta_1 = 0.9, \beta_2 = 0.99$. The loss function is categorical cross-entropy, since the task is multi-class classification.[2]

5.1.3 Procedure. We randomly split the 1460 UIs into three data partitions: 80% of the data is used for training (with 15% of the training data used as model validation) and the remaining 20% is used for testing. We use stratified sampling to ensure that our partitions are well balanced. Further, we compute class weights to avoid that some topics predominate the others, given that some of them are more frequent [39].

We train each model for 200 epochs at most, using early stopping (10 epochs patience) to retain the best model weights, and monitor its performance on the validation set. We do not apply any data augmentation technique (e.g. cropping or flipping) since UIs should be assessed according to the original layout design in any case.

5.1.4 Results. Table 3 summarizes the results of these experiments. Top rows report individual input representations and bottom rows report combined representations. As can be seen in the top-k accuracy columns, our model is able to identify the right topic most of the time. As a reference, a random classifier would be $1/20 = 5\%$ accurate. To illustrate model performance further, we also

[2]The autoencoder uses the mean squared error as loss function, since its job is to reconstruct an input image, but once trained we use the output of the bottleneck layer (the UI embedding) as a feature vector for classification.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

report the usual retrieval-based metrics (Precision, Recall, and F-measure) and the Area Under the Curve (AUC) score, which measures the discriminative power of any classifier [63].

	Top-k Accuracy						
Input	$k = 1$	$k = 3$	$k = 5$	**Prec.**	**Rec.**	F_1	**AUC**
$\mathcal{S}$: Screenshot	**75.8**	**88.4**	**92.9**	**76.6**	**75.8**	**75.4**	**95.1**
$\mathcal{W}$: Wireframe	39.4	66.1	78.7	50.1	39.4	40.2	85.3
$\mathcal{E}$: Embedding	50.9	73.6	84.3	60.2	50.9	51.1	89.5
$\mathcal{S} + \mathcal{E}$	**85.5**	89.5	**94.1**	**86.3**	58.5	**85.6**	**97.3**
$\mathcal{W} + \mathcal{E}$	81.7	**90.7**	93.3	82.9	81.7	81.7	96.5
$\mathcal{S} + \mathcal{W}$	55.8	75.3	83.7	60.3	55.8	56.4	91.3
$\mathcal{S} + \mathcal{W} + \mathcal{E}$	83.2	90.3	94.0	83.9	**83.2**	83.3	96.5

Table 3. Performance results of different UI design representations for multi-class classification (20 classes). All metrics are reported in percentage. The best result (column-wise) is denoted in bold typeface, both for individual and combined models.

We can see that the individual input representation that achieves the best classification performance is Screenshot (75% Top-1 accuracy, 95% AUC), followed by Embedding (50% Top-1 accuracy, 89% AUC), and Wireframe (39% Top-1 accuracy, 85% AUC). These results are quite competitive, considering that a random classifier would achieve 5% accuracy and 50% AUC, however there is room for improvement when it comes to using semantic wireframes as input data. We argue that the poor performance of the wireframes is probably due to the chosen color scheme to represent each UI element, which was inherited from the Rico dataset. For example, 'Image' and 'Date Picker' have a very similar (red-based) color, so both UI elements may become indistinguishable to any classifier that considers color as discriminative feature. Interestingly, all model combinations improve performance but it is the combination of Screenshot + Embedding that achieves the best classification results, with 85% Top-1 accuracy and 97% AUC. We conclude that our UI embeddings improve classification performance. We should note, however, that the Screenshot + Wireframe + Embedding combination achieved better recall overall, so depending on the task at hand it may be more interesting to consider this model over the simpler Screenshot + Embedding combination where recall is more important than precision. In our current implementation, XUI uses the Screenshot + Embedding model for topic classification.

5.2 Element Saliency Prediction

We evaluate the performance of our saliency model for predicting the key parts of a UI. We test also a pre-trained DL model and a classic computer vision model as baseline conditions. Here we provide an overview of these models. The *Supplementary Materials* on our companion website describe each model in detail. These experiments helped us to decide the final model for structural processing we implemented in Section 3.

5.2.1 Models. We compare the Grad-CAM distributions derived from our visual ConvNet against the Grad-CAM distributions derived from ResNet50 [27] and GBVS [26]. On the one hand, ResNet is the most popular pre-trained deep network used in visual saliency prediction, though it was trained on the ImageNet database [11], which comprises mostly natural/general-purpose images, so it is unclear if it would generalize to UIs. On the other hand, GBVS is a classic computer vision model that requires no training data and achieves reasonable performance [40].

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

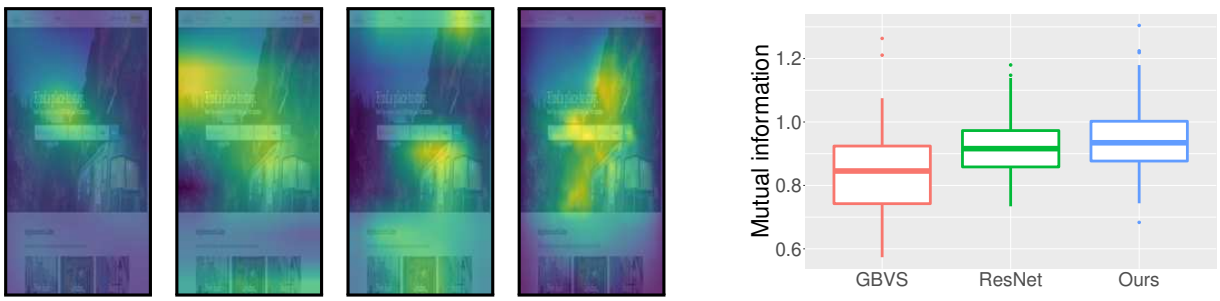
5.2.2 Data. We evaluate the three models over the webpage saliency dataset [72] since it is the closest public resource of UIs that includes visual hierarchies (source code) and accompanying eye fixation data. The dataset has 149 webpage screenshots from various domains rendered with the Chrome browser in full-screen mode at a resolution of 1360×768 px. The dataset includes eye fixation data from 11 users (7 female, 4 male) recorded with a monocular eyetracker at a sampling rate of 1000 Hz.

5.2.3 Procedure. First we resized the screenshots to 256×128 px, which is the same screen size used in our previous experiments. The fixation data were scaled accordingly. Then we generated fixation heatmaps by convolving a 2D Gaussian filter of $\sigma = 25$ px on each fixation point center $\mu_i = (x_i, y_i)$. This size approximates the size of the foveal region in the human vision system [72].

We compute the mutual information score between the ground-truth saliency maps (i.e., the fixation heatmaps) and the predicted saliency maps (i.e. the Grad-CAM activation maps) of each model. The mutual information quantifies the amount of information obtained about one random variable through observing the other random variable [35]. The minimum value occurs when both distributions are independent, i.e. when they do not share information. Therefore the higher the score, the better.

5.2.4 Results. Figure 8(e) summarizes the results of these experiments. As can be observed, the Grad-CAM saliency maps derived from our visual ConvNet correlate better with ground-truth fixation data. To precisely quantify the differences between models, we aggregate the data per condition and ran a Kruskal-Wallis rank sum test as a non-parametric omnibus test. (The Shapiro-Wilk test of normality was non-significant but the Bartlett test of sphericity was significant, therefore violating one of the ANOVA assumptions and suggesting that a non-parametric test is more appropriate.) The test was statistically significant: $\chi^2(2, N = 433) = 67.02, p < .001, \phi = 0.393$; so we ran pairwise comparisons using the non-parametric Wilcoxon rank sum test (Bonferroni-Holm corrected) and found that our model outperformed both GBVS ($p < .001$) and ResNet ($p < .05$). The effect size ϕ is large [8], suggesting a practical importance of these differences. We refer the interested reader to the publications of the American Psychological Association (APA) [2, 83] to gain a deeper understanding in these statistical methods and their suitability for this kind of analysis.

In sum, our visual ConvNet model has learned to attend to salient regions of the screenshots after topic classification training. Therefore, we are confident that it can provide an accurate estimation of the most salient UI elements. Figure 8 provides an example of each model's predictions.



(a) Reference (b) Ours (c) ResNet (d) GBVS (e) Mutual information scores

Fig. 8. Visual saliency predictions from various machine learning models against ground-truth fixations (a) and boxplots comparing the mutual information score (e) between predicted and ground-truth saliency maps.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

6 USER EVALUATION

We conducted a crowdsourcing study to assess the quality of the descriptions delivered by our XUI method, which would determine eventually how end-users perceive XUI in practice. In addition to the three types of XUI descriptions (caption, simple, detailed), we also considered the state-of-the-art captioning service provided by the Microsoft Cognitive Services API.[3] In what follows, we will refer to this method as 'MS caption' which, to our knowledge, it is presumably based on the work of Tran et al. [77] and Hu et al. [29]; although Microsoft has never disclosed the actual captioning system they offer in their Cognitive Services API. We also collected human-generated descriptions, as explained in the next section.

For this evaluation, we used the same test partition of the Enrico dataset from our previous topic classification experiments (Section 5.1) as input stimuli. As discussed in Section 2.1, currently there is no computational model to provide automatic captions of user interfaces. However, a production-level captioning service such as the Microsoft Cognitive Services API is already available, so it is valuable to use it as a baseline condition. In addition, using such an API fosters replicability efforts, since many open source image captioning implementations are very difficult to use in practice. Critically, most of them require complex environment setups and non-trivial data preprocessing pipelines to work with custom images.

6.1 Participants

We recruited 146 participants (64 female, 80 male, 2 other) aged 18–76 (M=30 years) via Prolific,[4] a crowsourcing platform with a large pool of workers, mostly from US and UK. All participants had a worker approval rate of 95% and could complete the study only once. Participants were proficient English speakers, had normal or corrected-to-normal vision, and had not been diagnosed as having any reading disorder. The study took 15 minutes on average to complete and participants were paid £2.5 (3.28 US dollars), which corresponds to an hourly wage of £8.61/h ($11.31/h).

In order to collect human-generated data for later comparisons, we first ran the study with 48 participants (28 female, 20 male) aged 18–72 and recorded 706 human descriptions. Each UI screenshot was described by two different participants on average, following the procedure described later. Then we ran the study with the remaining 98 participants (36 female, 60 male, 2 other) aged 18–76. Participants from the initial group were not allowed to participate in this second round.

In sum, the 48 participants from the initial group assessed four automatic UI captioning methods (the three types of XUI description and the MS captions) whereas the other 98 participants assessed, in addition, the human descriptions generated by the initial group of participants. These human descriptions were shown at random to the participants in the later group.

6.2 Materials

Our study was made available via a web-based application. Participants had to assess UI descriptions for the screenshots in the test partition of the Enrico dataset. Therefore, the UIs were not seen by XUI before and thus reflect a practical, real-world scenario. Participants were shown one UI description at a time, randomly sampled from the available data. Figure 9 shows a stimulus condition example.

Since participants completed the study remotely, we introduced control descriptions as an additional quality control mechanism, to be described later. The total number of ratings received by each condition is depicted in Table 4.

[3]https://azure.microsoft.com/en-us/services/cognitive-services/
[4]https://www.prolific.co/

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

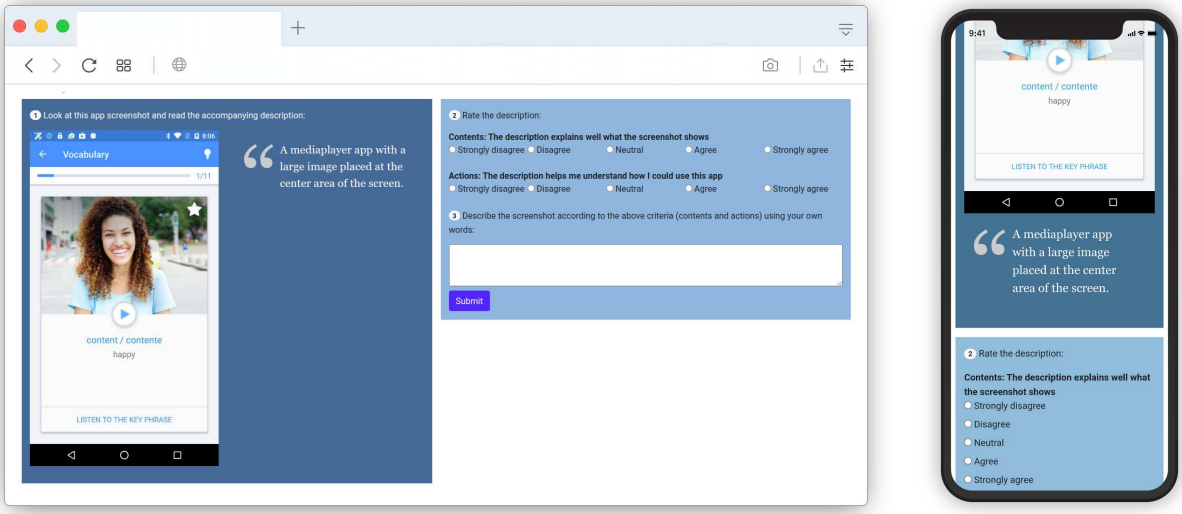


Fig. 9. Stimulus condition example, showing how a 'XUI caption' did look like on a desktop browser (left) and on a mobile phone (right). The user is prompted to look at the UI screenshot and read the accompanying description, then rate it and describe it using their own words.

Condition	Initial group (N=48)	Later group (N=98)	Total (N=146)
XUI Caption	152	255	407
XUI Simple	150	269	419
XUI Detailed	150	246	396
MS Caption	153	251	404
Control	101	169	270
Human	—	255	255

Table 4. Number of ratings collected for each of the evaluated UI captioning methods.

6.3 Procedure

Participants were asked to rate each UI description using a 5-point Likert scale, with 1 denoting 'strongly disagree', 3 denoting 'neutral', and 5 'strongly agree'. Each participant had to assess 3 descriptions from each method plus 2 control descriptions. We ensured that no duplicated cases (i.e. a combination of screenshot and captioning method) were shown to the same participant.

6.4 Design

The study is a within-subjects repeated-measures design; i.e. the same participant was exposed to all conditions (5 types of UI description plus 1 control condition) and had to rank each of them more than once (3 ratings per description plus 2 control ratings). Statistical significance was considered at the $\alpha = .05$ level. The dependent variables of this experiment are:

(1) *Contents:* It describes well what the screenshot shows.

(2) *Actions:* It helps me understand how I could use this app.

We chose these variables because, together, they assess the fundamental properties of any UI description; i.e., a good description should inform about what is shown on the interface and should do it in a way that it promotes understandability about the UI usage.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

To ensure that participants would take the study seriously, we introduced dummy sentences (lorem ipsum text)[5] as control conditions, which should score low in all measures. Twenty participants rated higher than 2 some of the control descriptions, so we did not consider these participants for our subsequent analysis.

6.5 Results

Figure 10 shows the score distributions segregated by each of the experiment conditions. As expected, the Human descriptions scored higher than any other method, both in terms of *Contents* ($M = 3.52$) and *Actions* ($M = 3.05$). All XUI descriptions scored higher than those provided by the Microsoft Cognitive Services API. Out of the three XUI description types, 'simple' scored slightly higher than 'detailed' in terms of *Contents* ($M_{\text{simple}} = 3.22$ vs $M_{\text{detailed}} = 3.15$) and vice versa in terms of *Actions* ($M_{\text{simple}} = 2.67$ vs $M_{\text{detailed}} = 2.69$). As observed, we can conclude that XUI and Human descriptions performed similarly.

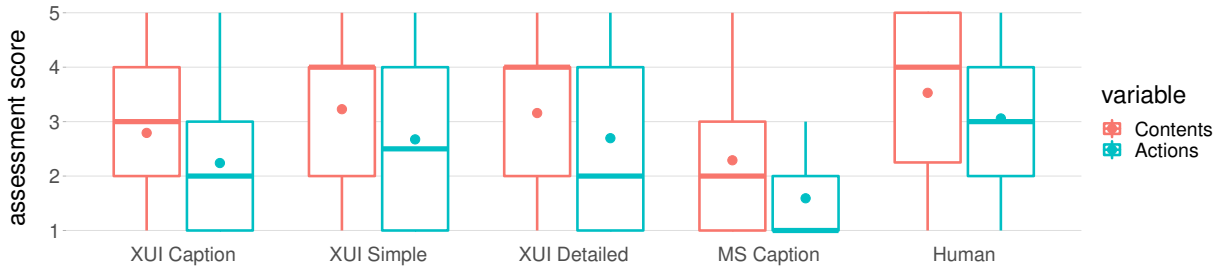


Fig. 10. Boxplot of user study results. Thick lines denote median values. Dots denote mean values. 'MS caption' refers to the descriptions provided by the Microsoft Cognitive Services API.

We investigated whether there is any difference between each of the five conditions, for which we use a linear mixed-effects (LME) model where each dependent variable (either *Contents* or *Actions*) is explained by each condition and participants are considered random effects. An LME model is appropriate here because the dependent variables are discrete. In addition, LME models are quite robust to violations of several distributional assumptions [69].

We fit the LME models and compute the estimated marginal means for specified factors. We then run pairwise comparisons (also known as *contrasts* in LME parlance) with Bonferroni-Holm correction to guard against multiple comparisons. The difference between MS captions and any of the other methods was significant with regards to both dependent variables ($p < .001$ in all cases). This was so for XUI captions ($p < .01$ in all cases). The difference between XUI simple and Human descriptions was non-significant for both dependent variables ($p > .05$). The difference between XUI simple and detailed descriptions was non-significant for both dependent variables. All other comparisons were statistically significant. We conclude therefore that (1) the three types of XUI descriptions are perceived to describe the contents of the UI similarly, but only XUI's 'simple' descriptions are on par with Human descriptions in this regard, and (2) 'simple' descriptions are perceived to be as actionable as Human descriptions.

7 READABILITY ANALYSIS

We further evaluated our XUI method via objective readability measures, by analyzing the five methods assessed in our user evaluation: XUI Caption, XUI simple, XUI detailed, MS Caption, and Human.

[5]Lorem ipsum is a Latin-like placeholder text commonly used in web design, see https://www.lipsum.com/.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

Given the lack of ground-truth UI descriptions,[6] we compute the usual metrics in NLG to assess text *diversity*: running words (number of total words, including duplicates), number of unigrams (unique words), and number of bigrams (sequences of two words). We also compute the *uniqueness* score attained by each method, defined as the proportion of unique UI descriptions generated. This score informs about the text variability of any captioning method, whether human or automatic. We argue that the higher the uniqueness the better since, all things being equal, it indicates that the method was able to generate a diverse set of UI descriptions. A uniqueness score of 100% means that all the generated descriptions were different.

We then compute the Automated Readability Index (ARI) [71] attained by each method. ARI is one of the most popular measures to assess comprehension of reading material. Concretely, ARI was designed to gauge the understandability of any English text by estimating the US grade level required to comprehend such text. ARI has better test-retest reliability than other readability indices [75] and is calculated with the following formula:

$$\text{ARI} = \left\lceil 4.71\frac{C}{W} + 0.5\frac{W}{S} - 21.43 \right\rceil \tag{3}$$

where C, W, S are the number of characters, words, and sentences, respectively, and $\lceil\cdot\rceil$ denotes the ceiling operation. For this analysis, all sentences produced by each method are concatenated so that we can consider a single text source per method, since ARI is not apt for very short texts, let alone single sentences. The results of this experiment are reported in Table 5.

Method	Running words	Unigrams	Bigrams	Uniqueness	Words/sentence	Chars/word	ARI
XUI Caption	4164	72	198	99.26	15.56	4.19	7
XUI simple	6605	143	356	98.88	24.78	4.32	12
XUI detailed	8132	158	364	100.0	30.23	4.09	13
MS Caption	1638	81	135	14.13	6.65	3.74	0
Human	7786	1620	4540	99.13	14.59	4.41	7

Table 5. Readability analysis results. 'MS caption' refers to the descriptions provided by the Microsoft Cognitive Services API.

We can see that XUI descriptions are both diverse and variable, as indicated by the number of generated words, whether total (running words) or unique (unigrams count), as well as the bigrams count in the generated descriptions. We can also see that XUI descriptions perform similarly to human-generated data, although human descriptions are more diverse than any automatic method, which is understandable because XUI generates template-based descriptions and MS captions are rather short. We can conclude, however, that our participants were not influenced by the system-provided descriptions (i.e. they did not serve as a priming effect) and thus our user evaluation procedure was adequate.

The similarity between XUI and human descriptions in terms of text variability is further reflected by the uniqueness proportion scores, which range between 98–100%. Note that none of the evaluated methods optimize for uniqueness or diversity, as it would be straightforward to generate (unintelligible) arbitrary strings and achieve a 100% score.

The unigrams/bigrams ratio has been reported as a proxy measure for evaluating the degree of diversity in generative models in natural language processing [41], with a lower unigrams/bigrams

[6]Even if we could use our crowdsourced human captions, they must undergo manual revision to filter out potentially low-quality contributions and clean up the texts.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

ratio indicating more diversity. The ratio is 0.39 for XUI on average (caption: 0.36, simple: 0.4, detailed: 0.43), whereas it is 0.35 for human descriptions and 0.60 for MS captions. A chi-square test of independence showed a significant association between the type of description generated and the unigrams/bigrams ratio: $\chi^2(4, N = 1518) = 9.59, p < .05, \phi = 0.08$. A post-hoc pairwise test of proportions (Bonferroni-Holm corrected) revealed that the MS captions had a significantly higher ratio than any other method ($p < .05$), indicating that MS captions are less diverse overall. All other comparisons were not statistically significant.

We noticed that the Microsoft Cognitive Services API is unable to actually describe the UI contents, and most of the time returns '*A screenshot of a cell phone*'. These results evidence that state-of-the-art computational models of image captioning do not work for UIs. On the other hand, XUI's underlying template-based NLG engine is quite flexible when it comes to producing distinct descriptions. A chi-square test of independence showed a significant association between the type of description generated and the uniqueness score: $\chi^2(4, N = 1518) = 70.56, p < .001, \phi = 0.21$. A post-hoc pairwise test of proportions (Bonferroni-Holm corrected) revealed that the MS captions had a significantly lower uniqueness score than any of the other methods ($p < .001$). All other comparisons were not statistically significant.

As further evidenced by the computed readability indices, XUI descriptions are easy to understand. There is a significant association between the type of description generated and ARI scores: $\chi^2(4, N = 1518) = 13.69, p < .01, \phi = 0.09$. A post-hoc pairwise test (Bonferroni-Holm corrected) revealed that the MS captions are the easiest to read overall ($p < .01$) but no difference was found between MS and XUI captions. However, we should remind the reader that MS captions do not really describe the UI. As a reference, an ARI score of 0 relates to a reading comprehension level for Kindergarten children [71]. An ARI score of 7 relates to seventh grade students (12 years old) whereas an ARI score of 13 relates to a reading level of college students (18 years old). Interestingly, despite the fact that XUI's 'detailed' descriptions have more words per sentence on average than XUI's 'simple' descriptions, their reading difficulty is only one level above. No statistically significant difference between any of the three XUI methods and Human descriptions was found.

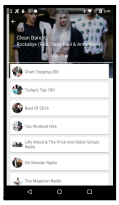
This experiment demonstrates that XUI descriptions are easily understandable, on par with human descriptions. It also validates the need for new technical solutions to provide meaningful descriptions of graphical user interfaces, since state-of-the-art captioning systems are unable to fulfill this goal at present. Of course, all automated readability indices have some limitations, and ARI is not an exception. For example, the intent of the reader[7] is a key factor not taken it into consideration by any readability index. Therefore, a formal user evaluation like the one we conducted in the previous section is recommended to complement any automatic readability analysis.

8 DISCUSSION, LIMITATIONS, AND FUTURE WORK

This work addresses the generation of user interface descriptions more deeply than the casual descriptions a general image captioning model might produce, even with specialized training sets that might appear in the future. To the best of our knowledge, this work is the first UI captioning system that delivers understandable descriptions about what is in the UI (contents) and what can be done (actions), without the need of machine learning datasets. Ultimately, XUI could have interesting applications beyond the examples hinted in the Introduction section, namely: assisting first-time users, ALT text generation, interactive AI, and describing screenshots in technical documentation. For example, since our software is open source, we envision many applications to be developed with XUI or its modules, including the following ones.

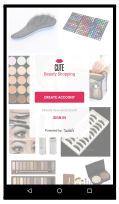
[7] In recreational reading, for example, a low readability index is preferred.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.



XUI Caption: *'A list app having a large image element placed at the top part of the screen.'*
XUI Simple: *'A list app with a large image element placed at the top area. You can see a list of elements, typically arranged in rows.'*
XUI Detailed: *'The screenshot is a list screen. It shows a list of elements, typically arranged in rows. You may notice a large image located at the top area.'*
MS Caption: *'A screenshot of a cell phone.'*
Human: *'List of elements about the song you're hearing.'*

XUI Caption: *'A mediaplayer screen having a large background image element located at the center part of the screen.'*
XUI Simple: *'The interface looks like a mediaplayer app with a large background image component located at the center area of the screen. You can see a music or video playback functionality.'*
XUI Detailed: *'That app must be a mediaplayer screen. It can be noticed a music or video playback functionality. You are likely to see a large background image placed at the center area of the screen.'*
MS Caption: *'Mike Zito talking on a cell phone screen with text.'*
Human: *'It's the screenshot of a mediaplayer where you can see the singer and the image of the album.'*



XUI Caption: *'This screen must be a modal app having a large image element located at the left area.'*
XUI Simple: *'A modal app having a large image component ubicated at the left part. It shows a popup window with additional content.'*
XUI Detailed: *'This interface must be a modal app. You can see a popup window with additional content. You are likely to see a large image placed at the left area of the screen that you can possibly click.'*
MS Caption: *'A close up of text on a white background.'*
Human: *'The images are across the whole of the background of the app - the app is for beauty shopping.'*

XUI Caption: *'This screen must be a gallery app having a large image element located at the right area.'*
XUI Simple: *'The interface looks like a gallery screen having a large image element located at the right part of the screen. It shows a grid-like layout with multiple images and text.'*
XUI Detailed: *'This screenshot must be a gallery app. It shows a grid-like layout with multiple images and text. You are likely to see a large image located at the right area of the screen.'*
MS Caption: *'Taylor Swift, Usher, Jason Aldean are posing for a picture.'*
Human: *'It is clear and nice.'*

XUI Caption: *'That interface is a news screen with a large image element ubicated at the center area.'*
XUI Simple: *'This screen looks like a news app with a large image component situated at the center part. It shows information about events.'*
XUI Detailed: *'The screenshot is a news screen. It shows information about events. You are likely to see a large image situated at the center area of the screen.'*
MS Caption: *'A screenshot of a cell phone screen with text.'*
Human: *'Screenshot of a news screen with large images forming the bulk of each news item.'*

XUI Caption: *'The app looks like a terms app with a large text placed at the center area of the screen.'*
XUI Simple: *'This screen is a terms screen with a large text located at the center part. You can see a wall of text, mostly likely describing legal conditions.'*
XUI Detailed: *'This interface is a terms screen. It can be noticed a wall of text, mostly likely describing legal conditions. There is a large text situated at the center part of the screen.'*
MS Caption: *'A screenshot of a cell phone.'*
Human: *'The screenshot is of the terms and conditions for a trip, and gives different tariffs.'*

Fig. 11. UI description examples from the test partition of our dataset. None of these UIs were used while training XUI's computational models. We also show the descriptions provided by the Microsoft Cognitive Services API and our crowdsourcing participants. All examples were selected pseudo-randomly for a handful of UI topics, removing duplicates, aimed at providing a diverse set of sample outputs for all the systems being compared.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

Interactive captioning: XUI may be useful to understand interactive applications that dynamically change their elements. Since the UI structure is already available at runtime, it is just needed an additional pixel-based representation that can be acquired programmatically (e.g. using the Canvas API, which is available both for Android and iOS apps as well as HTML5 websites).

Improved search and retrieval: Query-by-image retrieval models [10, 48] can be enhanced by incorporating explicit information about the UI design, their contents, or the interactive properties. This information should help disambiguate and improve search.

Smart tutorials: The concept of "UI tours" is very popular in web design [21], as a means to provide contextual help. XUI could be used to describe individual elements and present this information as tooltips, for example, in order to inform the users about the functional role of each element in the UI.

Automatic UI tagging: Designers could query XUI's underlying topic classifier and request e.g. the top-3 most likely labels of a given UI layout design, in order to display that information as inline tags in a web gallery.

Annotation interfaces: Our topic classifier can speed up the annotation process for new UIs [39] by arranging the topics list from higher to lower probability (n-best list), better assisting the human annotator.

Tag cloud visualization: Designers can communicate graphically the n-best list of topics with a tag cloud, where each tag is rendered with a font size proportional to the design topic probability, so that larger tags would indicate higher importance.

Currently XUI is implemented as a command-line interface program written in Python. We believe that XUI could potentially generalize to other graphical interfaces, since in most cases the semantic information of the UI is either available by default (e.g. web pages) or can be easily obtained by traversing the view hierarchy (e.g. mobile apps), usually represented by a document object model tree. It could even be possible to use pixel-based methods to reverse engineer the UI structure [13]. Nevertheless, if the semantic information is not available or cannot be reliably computed, XUI can deliver a fallback description using only information from our topic classifier, which just requires a UI screenshot as input. For example, the 'simple' description example shown in Figure 1 would become: *'This is a menu screen. It has a list of selectable options.'*

One limitation of XUI is that the UI topic predictions are limited to 20 different categories, therefore, if the design layout topic is not classified correctly, a wrong template would be populated with data that eventually could confuse the user. This is an inherent limitation of any system or method relying on a 'chain of responsibility' modules, since, ultimately, cascading errors are likely to negatively impact the final UI descriptions. For XUI to cope with this issue, the verbs are phrased in a way that hopefully inform the user about the likelihood of the descriptions, leaving thus some room for flexibility. Figure 12 shows some examples of what could be considered as bad descriptions.

Relying on templates instead of a learned decoder backbone can be seen as a rather rigid approach to text generation. However, our templates do not rely on a slot-filling mechanism only, but considers a declarative representation that provides more control over the generated text while ensuring diversity. These capabilities were shown to produce understandable descriptions, as evidenced by our readability experiments (Section 7), yet we cannot claim that XUI descriptions would fit everyone. According to the achieved readability scores, XUI descriptions are likely to be understandable to people aged 12–18 and above, so there could be room for improvement regarding

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

XUI Caption: *'A bare screen with a large background image element ubicated at the center part of the screen.'*
XUI Simple: *'This screenshot could be a bare screen with a large background image located at the center part. It has a largely empty area on screen.'*
XUI Detailed: *'The interface might be a bare screen. You should see a largely empty area on screen. You can find a large background image located at the center area.'*

XUI Caption: *'Unknown screenshot.'*
XUI Simple: *'The interface is unknown.'*
XUI Detailed: *'That interface is unknown, however you are likely to see a large background image ubicated at the center area of the screen which you can try to click.'*

Fig. 12. Bad XUI description examples from the test partition of our dataset. Top: the UI is misclassified as a 'bare' screen. Bottom: the UI is classified as 'other', so no topic description is available, however, even in this failure case, the detailed description provides an affordability hint.

how XUI templates are phrased. Further, the templates are fully customizable, so they can be easily translated to any language, thereby providing support for an international audience. We acknowledge, however, that some languages have challenging grammatical inflections so we leave this translation task as an opportunity for future work.

Another exciting line of future work includes investigating few-shot and transfer learning using the latest efforts in image captioning using cross-modal vision-and-language pre-training [29, 44, 51], which have been trained on millions of images with accompanying captions. It should be possible to fine-tune these kind of models with a small dataset such as the crowdsourced annotations we have collected in our user study. We could even try to use the XUI templates for creating better prompts that can guide large language models such as T5 [20] and the GPT family [9, 65].

We believe there is potential for our method to serve more specific target users, such as, for example, the elderly or visually impaired users. As a first step, we deliberately decided to provide the most generic application of our technique, which is high-level and task-free. Previous work has investigated what do people with vision impairments expect to find in descriptions of natural images [57, 74], however the UI domain has been largely overlooked. Additionally, we note there is only a single screenshot to describe a given UI, however humans may provide different (and better) captions when the available information increases. For example, users or developers of an application have more context than a single screenshot to describe more informative UI captions. Therefore, for future work, we plan to run different formative engagement tasks with specific user groups in order to help us adapt XUI to their specific needs.

Finally, we plan to improve the actionability hints of XUI descriptions, since currently they only inform whether an element is clickable/tappable, which may be insufficient information for some users. Providing additional interactions with the UI, for example, could promote understandability further. Additionally, describing the actual contents of an image could be important for contextualizing an interaction as well, thereby providing the user with more precise information about the tasks that can be performed with the UI. Going forward, maybe XUI can be extended to take actual eye fixations into account, in case a user is actively looking at the interface and the system can interactively react on that kind of fixation, delivering an description that is tailored to the user's visual context, in real-time.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

9 CONCLUSION

We have introduced XUI, a novel automatic captioning method to create informative descriptions of UIs that is fully controllable and does not rely on training datasets. Being able to describe any UI screenshot in natural language is important to several user groups and applications, yet until now it was not possible to do so successfully.

XUI builds upon computational models for topic classification and element saliency prediction ("what to say") and template-based natural language generation ("how to say it"). Together, these capabilities allows XUI to provide high-level and task-free descriptions, making it thus the first workable general-purpose technique to describe UIs in natural language automatically.

We found that the descriptions delivered by XUI are highly readable and perceived to accurately describe the UI. Overall, based on our results, XUI's 'simple' descriptions represent the best compromise solution between descriptive accuracy (Section 6) and readability (Section 7). Nevertheless, we believe that we have barely scratched the surface of what is possible with XUI. Our software, data, and computational models are available at https://luis.leiva.name/xui/.

ACKNOWLEDGMENTS

We acknowledge the computational resources provided by the Aalto Science-IT project. We thank Homayun Afrabandpey, Daniel Buschek, Jussi Jokinen, and Jörg Tiedemann for reviewing an earlier draft of this article. This work has been supported by the Horizon 2020 FET program of the European Union through the ERA-NET Cofund funding (grant CHIST-ERA-20-BCI-001), the European Innovation Council Pathfinder program (SYMBIOTIK project), and the Academy of Finland (grants 291556, 318559, 310947).

REFERENCES

[1] A. Adadi and M. Berrada. 2018. Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI). *IEEE Access* 6 (2018).

[2] American Psychological Association. 2020. *Publication Manual of the American Psychological Association* (7th ed.). American Psychological Association (APA).

[3] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *Proc. CVPR*.

[4] G. Angeli, P. Liang, and D. Klein. 2010. A simple domain-independent probabilistic approach to generation. In *Proc. EMNLP*.

[5] N. Banovic, T. Grossman, J. Matejka, and G. Fitzmaurice. 2012. Waken: reverse engineering usage information and interface structure from software videos. In *Proc. UIST*.

[6] J. P. Bigham, C. Jayant, H. Ji, G. Little, A. Miller, R. C. Miller, R. Miller, A. Tatarowicz, B. White, S. White, and T. Yeh. 2010. VizWiz: Nearly Real-Time Answers to Visual Questions. In *Proc. UIST*.

[7] J. P. Bigham, R. S. Kaminsky, R. E. Ladner, O. M. Danielsson, and G. L. Hempton. 2006. WebInSight: Making Web Images Accessible. In *Proc. ASSETS*.

[8] M. Borenstein. 2009. Effect sizes for continuous data. In *The handbook of research synthesis and meta-analysis* (2nd ed.), H. Cooper, L. V. Hedges, and J. C. Valentine (Eds.). Sage Foundation.

[9] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. 2020. Language Models are Few-Shot Learners. In *Proc. NeurIPS*.

[10] B. Deka, Z. Huang, C. Franzen, J. Hibschman, D. Afergan, Y. Li, J. Nichols, and R. Kumar. 2017. Rico: A Mobile App Dataset for Building Data-Driven Design Applications. In *Proc. UIST*.

[11] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proc. CVPR*.

[12] M. Dixon and J. Fogarty. 2010. Prefab: Implementing Advanced Behaviors Using Pixel-based Reverse Engineering of Interface Structure. In *Proc. CHI*.

[13] M. Dixon, D. Leventhal, and J. Fogarty. 2011. Content and Hierarchy in Pixel-based Methods for Reverse Engineering Interface Structure. In *Proc. CHI*.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

[14] M. Dixon, A. Nied, and J. Fogarty. 2014. Prefab Layers and Prefab Annotations: Extensible Pixel-based Interpretation of Graphical Interfaces. In *Proc. UIST*.
[15] P. L. Dognin, I. Melnyk, Y. Mroueh, J. Ross, and T. Sercu. 2019. Adversarial Semantic Alignment for Improved Image Captions. In *Proc. CVPR*.
[16] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *Proc. ICLR*.
[17] A. Dutta, Y. Verma, and C. V. Jawahar. 2018. Automatic Image Annotation: The Quirks and What Works. *Multimedia Tools Appl.* 77, 24 (2018).
[18] H. Fang, S. Gupta, F. Iandola, R. Srivastava, L. Deng, P. Dollar, J. Gao, X. He, M. Mitchell, J. Platt, L. Zitnick, and G. Zweig. 2015. From captions to visual concepts and back. In *Proc. CVPR*.
[19] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth. 2010. Every Picture Tells a Story: Generating Sentences from Images. In *Proc. ECCV*.
[20] W. Fedus, B. Zoph, and N. Shazeer. 2021. Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity. arXiv:2101.03961. (2021).
[21] J. Garcia. 2011. *Ext JS in Action* (2nd ed.). Manning Publications.
[22] A. Gatt and E. Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.* 61 (2018).
[23] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. 2019. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *Proc. ICML*.
[24] C. Gleason, A. Pavel, E. McCamey, C. Low, P. Carrington, K. M. Kitani, and J. P. Bigham. 2020. Twitter A11y: A Browser Extension to Make Twitter Images Accessible. In *Proc. CHI*. 1–12.
[25] J. Gu, J. Cai, G. Wang, and T. Chen. 2018. Stack-captioning: Coarse-to-fine learning for image captioning. In *Proc. AAAI*.
[26] J. Harel, C. Koch, and P. Perona. 2007. Graph-Based Visual Saliency. In *Proc. NIPS*.
[27] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. CVPR*.
[28] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga. 2019. A Comprehensive Survey of Deep Learning for Image Captioning. *ACM Comput. Surv.* 51, 6 (2019).
[29] X. Hu, X. Yin, K. Lin, L. Wang, L. Zhang, J. Gao, and Z. Liu. 2021. VIVO: Visual Vocabulary Pre-Training for Novel Object Captioning. In *Proc. AAAI*.
[30] X. Hua and L. Wang. 2019. Sentence-Level Content Planning and Style Specification for Neural Text Generation. In *Proc. EMNLP*.
[31] J. Huang and M. B. Twidale. 2007. Graphstract: Minimal Graphical Help for Computers. In *Proc. UIST*.
[32] T. Intharah, D. Turmukhambetov, and G. J. Brostow. 2017. Help, It Looks Confusing: GUI Task Automation Through Demonstration and Follow-up Questions. In *Proc. IUI*.
[33] R. Kimchi. 1992. Primacy of wholistic processing and the global/local paradigm: a critical review. *Psychol. Bull.* 112 (1992).
[34] R. Kondadadi, B. Howald, and F. Schilder. 2013. A Statistical NLG Framework for Aggregated Planning and Realization. In *Proc. ACL*.
[35] A. Kraskov, H. Stögbauer, and P. Grassberger. 2004. Estimating mutual information. *Phys. Rev. E* 69 (2004).
[36] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F.-F. Li. 2017. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *Int. J. Comput. Vision* 123 (2017).
[37] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg. 2011. Baby Talk: Understanding and Generating Image Descriptions. In *Proc. CVPR*.
[38] R. Lebret, D. Grangier, and M. Auli. 2016. Neural Text Generation from Structured Data with Application to the Biography Domain. In *Proc. EMNLP*.
[39] L. A. Leiva, A. Hota, and A. Oulasvirta. 2020a. Enrico: A High-quality Dataset for Topic Modeling of Mobile UI Designs. In *Proc. MobileHCI*.
[40] L. A. Leiva, Y. Xue, A. Bansal, H. R. Tavakoli, T. Köroğlu, N. R. Dayama, and A. Oulasvirta. 2020b. Understanding Visual Saliency in Mobile User Interfaces. In *Proc. MobileHCI*.
[41] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proc. NAACL*.
[42] J. Li, D. Li, C. Xiong, and S. Hoi. 2022. BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation. arXiv:2201.12086. (2022).
[43] S. Li, G. Kulkarni, T. L. Berg, A. C. Berg, and Y. Choi. 2011. Composing simple image descriptions using web-scale N-grams. In *Proc. ACL*.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

[44] X. Li, X. Yin, C. Li, P. Zhang, X. Hu, L. Zhang, L. Wang, H. Hu, L. Dong, F. Wei, Y. Choi, and J. Gao. 2020. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In *Proc. ECCV*.

[45] Y. Li, G. Li, L. He, J. Zheng, H. Li, and Z. Guan. 2020. Widget Captioning: Generating Natural Language Description for Mobile User Interface Elements. In *Proc. EMNLP*.

[46] P. Liang, M. Jordan, and D. Klein. 2009. Learning Semantic Correspondences with Less Supervision. In *Proc. ACL/IJCNLP*.

[47] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. 2014. Microsoft COCO: Common Objects in Context. In *Proc. ECCV*.

[48] T. F. Liu, M. Craft, J. Situ, E. Yumer, R. Mech, and R. Kumar. 2018. Learning Design Semantics for Mobile Apps. In *Proc. UIST*.

[49] J. Lu, C. Xiong, D. Parikh, and R. Socher. 2017. Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning. In *Proc. CVPR*.

[50] W. Luo, Y. Li, R. Urtasun, and R. Zemel. 2016. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *Proc. NIPS*.

[51] Z. Luo, Y. Xi, R. Zhang, and J. Ma. 2022. VC-GPT: Visual Conditioned GPT for End-to-End Generative Vision-and-Language Pre-training. arXiv:2201.12723. (2022).

[52] K. R. McKeown. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press.

[53] S. W. McRoy, S. Channarukul, and S. S. Ali. 2000. YAG: A Template-Based Generator for Real-Time Systems. In *Proc. INLG*.

[54] T. Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.* 267 (2019).

[55] V. S. Morash, Y.-T. Siu, J. A. Miele, L. Hasty, and S. Landau. 2015. Guiding Novice Web Workers in Making Image Descriptions Using Templates. *ACM Trans. Access. Comput.* 7, 4 (2015).

[56] R. Moriyon, P. Szekely, and R. Neches. 1994. Automatic generation of help from interface design models. In *Proc. UIST*.

[57] M. R. Morris, A. Zolyomi, C. Yao, S. Bahram, J. P. Bigham, and S. K. Kane. 2016. "With Most of It Being Pictures Now, I Rarely Use It": Understanding Twitter's Evolving Accessibility to Blind Users. In *Proc. CHI*.

[58] D. Navon. 1977. Forest before trees: The precedence of global features in visual perception. *Cogn. Psychol.* 9, 3 (1977).

[59] J. Novikova, O. Dušek, and V. Rieser. 2017. The E2E Dataset: New Challenges For End-to-End Generation. In *Proc. SIGDIAL*.

[60] S. Pangoli and F. Paternó. 1995. Automatic generation of task-oriented help. In *Proc. UIST*.

[61] S. Pareddy, A. Guo, and J. P. Bigham. 2019. X-Ray: Screenshot Accessibility via Embedded Metadata. In *Proc. ASSETS*.

[62] B. A. Plummer, L. Wang, C. M. Cervantes, J. C. Caicedo, J. Hockenmaier, and S. Lazebnik. 2017. Flickr30K Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models. *Int. J. Comput. Vis.* 123, 1 (2017).

[63] D. Powers. 2011. Evaluation: from Precision, Recall and F-measure to ROC, Informedness, Markedness and Correlation. *J. Mach. Learn. Technol.* 2, 1 (2011).

[64] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *Proc. ICML*.

[65] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. *Language Models are Unsupervised Multitask Learners*. Technical Report. OpenAi.

[66] K. Ramnath, S. Baker, L. Vanderwende, M. El-Saban, S. N. Sinha, A. Kannan, N. Hassan, M. Galley, Y. Yang, D. Ramanan, A. Bergamo, and L. Torresani. 2014. AutoCaption: Automatic caption generation for personal photos. In *Proc. WACV*.

[67] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. 2016. Sequence level training with recurrent neural networks. In *Proc. ICLR*.

[68] E. Reiter and R. Dale. 2000. *Building natural language generation systems*. Cambridge University Press.

[69] H. Schielzeth, N. J. Dingemanse, S. Nakagawa, D. F. Westneat, H. Allegue, C. Teplitsky, D. Réale, N. A. Dochtermann, L. Z. Garamszegi, and Y. G. Araya-Ajoy. 2020. Robustness of linear mixed-effects models to violations of distributional assumptions. *Methods Ecol. Evol.* 11, 9 (2020).

[70] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. 2019. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *Int. J. Comput. Vision* 128, 2 (2019).

[71] R. J. Senter and E. A. Smith. 1967. *Automated Readability Index*. Technical Report AMRL-TR-6620. Wright-Patterson Air Force Base.

[72] C. Shen and Q. Zhao. 2014. Webpage Saliency. In *Proc. ECCV*.

[73] K. Simonyan and A. Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proc. ICLR*.

[74] A. Stangl, M. R. Morris, and D. Gurari. 2020. "Person, Shoes, Tree. Is the Person Naked?" What People with Vision Impairments Want in Image Descriptions. In *Proc. CHI*.

[75] G. Thomas, R. D. Hartley, and J. P. Kincaid. 1975. Test-Retest and Inter-Analyst Reliability of the Automated Readability Index, Flesch Reading Ease Score, and the Fog Count. *J. Lit. Res.* 7, 2 (1975).

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

[76] D. Todorovic. 2008. Gestalt principles. *Scholarpedia* 3, 12 (2008).
[77] K. Tran, X. He, L. Zhang, J. Sun, C. Carapcea, C. Thrasher, C. Buehler, and C. Sienkiewicz. 2016. Rich Image Captioning in the Wild. In *Proc. CVPR*.
[78] S. Wiseman, S. Shieber, and A. Rush. 2017. Challenges in Data-to-Document Generation. In *Proc. EMNLP*.
[79] S. Wiseman, S. M. Shieber, and A. M. Rush. 2018. Learning Neural Templates for Text Generation. In *Proc. EMNLP*.
[80] T. Yeh, T.-H. Chang, and R. C. Miller. 2009. Sikuli: Using GUI Screenshots for Search and Automation. In *Proc. UIST*.
[81] T. Yeh, T.-H. Chang, B. Xie, G. Walsh, I. Watkins, K. Wongsuphasawat, M. Huang, L. S. Davis, and B. B. Bederson. 2011. Creating Contextual Help for GUIs Using Screenshots. In *Proc. UIST*.
[82] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo. 2016. Image Captioning with Semantic Attention. In *Proc. CVPR*.
[83] S. Zedeck (Ed.). 2013. *APA Dictionary of Statistics and Research Methods* (1st ed.). American Psychological Association (APA).
[84] X. Zhang, L. de Greef, A. Swearngin, S. White, K. I. Murray, L. Yu, Q. Shan, J. Nichols, J. Wu, C. Fleizach, A. Everitt, and J. P. Bigham. 2021. Screen Recognition: Creating Accessibility Metadata for Mobile Applications from Pixels. In *Proc. CHI*.

A ALTERNATIVE CAPTIONING MODELS

In the following we analyze and provide example outputs of other image captioning models that could be used instead of the Microsoft Cognitive Services API. All these models have reported state-of-the-art results on a wide range of vision-language tasks, including image captioning, therefore they are considered strong baselines worth of investigation for UI captioning.

A.1 Models overview

A.1.1 BLIP. This is a new Vision-Language Pre-training (VLP) model which transfers flexibly to both vision-language understanding and generation tasks [42]. BLIP effectively utilizes a large-scale but noisy web dataset by bootstrapping the captions, as explained next.

BLIP is jointly trained on 129M images with three vision-language objectives: image-text contrastive learning (captioning), image-text matching, and image-conditioned language modeling. To improve captioning performance, a captioner produces synthetic captions given web images, and a filter removes noisy captions from both the original web texts and the synthetic texts.

A.1.2 CLIP. Contrastive Language-Image Pre-Training (CLIP) is a deep generative model trained on a variety of <image,text> pairs [64]. It can be instructed in natural language to predict the most relevant text snippet, given an image, without directly optimizing for the task, similarly to the zero-shot capabilities of the GPT famiy [65] and GPT-3 [9].

CLIP is trained on 400M image-text pairs from the internet. The implementation we have tested uses a dual encoder for representation learning comprising a ResNet-based image encoder and a Transformer-based text encoder. Then, it uses GPT-2 [65] as the language model decoder to generate captions for a given input image.

A.1.3 VC-GPT. Visual Conditioned GPT (VC-GPT) is a new model for generative vision-and-language using pre-trained models [51]. It is fine-tuned to the Visual Genome image-caption corpus, which comprises 110k distinct images [36]. The implementation we have tested uses a Vision Transformer (ViT) [16] as the image encoder and GPT-2 [65] as the language model decoder. This model has shown to outperform previous state-of-the-art image captioning models such as BUTD [3] and Oscar [44].

A.2 Readability analysis

We performed the same analysis as in Section 7, to better compare the performance of these alternative captioning systems with the results provided in Table 5. That is, we captioned all the UIs in the same test partition of the Enrico dataset from our topic classification experiments (Section 5.1) and computed the same evaluation metrics.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

Method	Running words	Unigrams	Bigrams	Uniqueness	Words/sentence	Chars/word	ARI
BLIP	2616	400	821	87.36	9.74	3.75	2
CLIP	2736	460	965	98.51	10.18	3.71	2
VC-GPT	2719	167	379	50.93	9.83	3.75	2

Table 6. Readability analysis results of the alternative image captioning models.

As can be observed, the three captioning systems produce more diverse and unique outputs than MS Captions, however, as shown in the next section, they are unable to describe properly what is in the interface (what role does the UI plays for interaction) and what the user can do with it (what does the UI afford). The ARI of these systems is 2, suggesting that the generated captions are suitable to first grade students. Remember that Human captions had an ARI of 7, whereas XUI's ARI ranged between 7 and 13.

A.3 Sample outputs

In the following we provide a series of output examples for different UI topics in the Enrico dataset, aimed at covering a wide range of the UI classes we have considered in our work; see Figure 6. All examples were chosen at random, using 42 as RNG seed and shuffling the test partition of the Enrico dataset to get the UI identifiers. Then, the images corresponding to these UI identifiers were captioned with the three systems previously described. Since the resulting automated captions were all lowercased, we applied the following post-processing: (1) uppercase the first letter of the first word in the caption and (2) ensure there the last word the caption ends with a dot. As a reference, in the following figures we provide the output of 'XUI Simple', to ease comparisons.

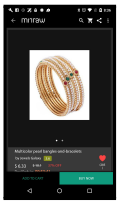
This is a preprint for personal use only. The published paper may be subject to some form of copyright.

BLIP: *'A cell phone with a text message on the screen.'*
CLIP: *'A picture of a person using a phone.'*
VC-GPT: *'A computer screen with a picture of a penguin on it.'*

XUI Simple: *'A chat screen with an icon component placed at the top-left part. It shows a messaging functionality to chat with friends.'*

BLIP: *'A cell phone with a bunch of news on it.'*
CLIP: *'A picture of a person using a phone.'*
VC-GPT: *'A laptop computer sitting on top of a desk.'*

XUI Simple: *'The app is a news app having a text placed at the center area of the screen. It has information about events.'*

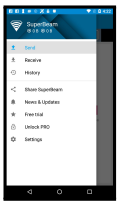


BLIP: *'A pair of gold bracelets on a cell phone.'*
CLIP: *'A display of a variety of sweet and savory treats.'*
VC-GPT: *'A poster with a picture of a cat on it.'*

XUI Simple: *'This screenshot looks like an editor app with a large image component located at the center part of the screen. You may notice a functionality to edit or create content.'*

BLIP: *'A music player with a guitar in front of a lake.'*
CLIP: *'A man playing a guitar in front of a distorted picture.'*
VC-GPT: *'A man is playing a video game on a tv.'*

XUI Simple: *'The interface looks like a mediaplayer app with a large background image component located at the center area of the screen. You can see a music or video playback functionality.'*



BLIP: *'A cell phone with the text super beam send on it.'*
CLIP: *'A picture of a bunch of items on a desk.'*
VC-GPT: *'A computer screen with a picture of a person on it.'*

XUI Simple: *'A menu app having a large image component located at the top area. You may notice a list of selectable options.'*

BLIP: *'A cell phone with the name wandelo on it.'*
CLIP: *'A picture of a computer screen with a price tag.'*
VC-GPT: *'A computer screen with a picture of a person on it.'*

XUI Simple: *'A login screen having a large text button located at the center area. You may notice some input fields that you must fill in to access the app.'*

BLIP: *'A close up of a cell phone on a table.'*
CLIP: *'A black computer keyboard with a screen of a computer.'*
VC-GPT: *'A black and white tv sitting on top of a table.'*

XUI Simple: *'A camera app having a large image placed at the bottom part of the screen. It shows a functionality to take photos.'*

Fig. 13. Examples of alternative captioning models. As a reference, the output of 'XUI Simple' is also provided in each case, to ease comparisons.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

BLIP: *'A cell phone with a picture of a woman's lips.'*
CLIP: *'A series of photographs showing a woman brushing her teeth.'*
VC-GPT: *'A collage of photos of a woman using a cell phone.'*

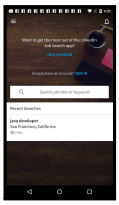
XUI Simple: *'This interface must be a gallery app having a large image located at the top area. You may notice a grid-like layout with multiple images and text.'*

BLIP: *'The colourgo pro app on a smartphone.'*
CLIP: *'A meter with numbers on it that say "F." and "K."'*
VC-GPT: *'A computer screen with a picture of a penguin on it.'*

XUI Simple: *'The interface looks like a form screen having a text component located at the top area of the screen. You may notice some input fields to enter data.'*

BLIP: *'A cell phone with a text message on it.'*
CLIP: *'A picture of a man on a computer.'*
VC-GPT: *'A blurry picture of a person typing on a computer.'*

XUI Simple: *'That screen must be a terms app with a large image located at the center part. You may notice a wall of text, mostly likely describing legal conditions.'*



BLIP: *'A wooden table with a keyboard and a phone.'*
CLIP: *'A picture of a keyboard and a menu.'*
VC-GPT: *'A computer screen with a picture of a person on it.'*

XUI Simple: *'That app must be a search screen having a large background image situated at the center part of the screen. You may notice a search functionality.'*

BLIP: *'A cell phone with a message on the screen.'*
CLIP: *'A picture of a coffee cup with a picture of a person on it.'*
VC-GPT: *'A computer screen with a picture of a person on it.'*

XUI Simple: *'A tutorial screen with a large image component situated at the left part. It shows introductory information about the app.'*

BLIP: *'A picture of a city street at night.'*
CLIP: *'A cell phone with a clock and arrow keys.'*
VC-GPT: *'A blurry photo of a city street.'*

XUI Simple: *'That screen must be a dialer screen having a text placed at the top area of the screen. It shows a keypad to enter numbers.'*

BLIP: *'A picture of a red sports car in a garage.'*
CLIP: *'A picture of a cell phone with a picture of a person holding it.'*
VC-GPT: *'A close up of a picture of a refrigerator.'*

XUI Simple: *'That screenshot looks like a profile screen having a large image component ubicated at the center area. You can see information about a user or a product.'*

Fig. 14. Examples of alternative captioning models (cont.) As a reference, the output of 'XUI Simple' is also provided in each case, to ease comparisons.