2017

Towards Automated Optimization of Web Interfaces and Application to E-commerce

Aoun Lutfi
University of Wollongong in Dubai, AounLutfi@gmail.com

Stefano Fasciani
University of Wollongong in Dubai, fasciani@uow.edu.au

Publication Details

Lutfi, A. & Fasciani, S. 2017, 'Towards Automated Optimization of Web Interfaces and Application to E-commerce', Computer and Applications (ICCA), 2017 International Conference on, IEEE, United States, pp. 79-84.

Towards Automated Optimization of Web Interfaces and Application to E-commerce

Aoun Lutfi
Faculty of Engineering and Information Sciences
University of Wollongong in Dubai
Dubai, UAE
al702@uowmail.edu.au

Stefano Fasciani
Faculty of Engineering and Information Sciences
University of Wollongong in Dubai
Dubai, UAE
stefanofasciani@stefanofasciani.com

Abstract—**In this paper, we present a system that supports the design of web graphical user interface by finding the optimal placement of interactive elements. The definition of optimal placement is context specific; it aims at maximizing measurable aspects of the user experience, and it is derived using expert knowledge embedded in the system, which is based on HCI principles, user studies, and data analytics. We use a novel modeling technique to represent the layout of web user interfaces. The model is unsupervisedly computed on information extracted by several image processing algorithms. The system identifies the website category, builds a layout model, compares it with the relevant optimal model, and recommends an alternative layout of interactive elements. To prove this concept, we present a study on e-commerce websites where placement of the checkout button has a significant impact on the online sale process conversion rate. The system identifies non-optimal placement, and it recommends an alternative position that is likely to improve the conversion rate. The results will be validated based on the Visa Checkout user experience. The system is implemented as an open-source software, and it currently supports the re-positioning of a single interactive element. In the paper, we discuss a further generalization of this approach to support a larger number of interactive objects in a wider spectrum of scenarios.**

Keywords—User Interface Modeling, User Experience Optimization, Human Computer Interaction, Complex System, E-commerce, Image Segmentation, Patter Recognition, Classification.

I. INTRODUCTION

Providing user-friendly systems is one of the key objectives for designers of interactive computer-based systems. As technology advances, novel interaction modalities emerge, providing new engagement opportunities as well as new challenges in the design process. Previous works on Human-Computer Interaction (HCI) have demonstrated that it is possible to influence the behavior of humans using computer systems [1], [2]. In the past, extensive user studies had to be carried out to collect sufficient data to identify effective strategies for influencing the users. Today, most personal devices are connected to the Internet, and user-activity logs are often collected with anonymous profiles. Large Internet companies collect a significant amount of information and appropriate large data analytics can identify specific users' preferences and patterns. This enabled the definition of context specific HCI design guidelines to improve system usability. The application of this expert knowledge has opened opportunities in a wide spectrum of fields. In this paper, we focus on web interfaces because of their popularity, universal accessibility, and nearly ubiquitous presence. In particular, we propose a system to automate the embedding of expert knowledge in the design of website layout. E-commerce, which is mostly based on web interfaces, has recently seen a tremendous growth. Worldwide sales are expected to double from $1.9 trillion in 2016 to $ 4 trillion by 2020 [3]. This requires that e-commerce platforms continue to evolve to increase the demand. E-commerce companies need to develop more sophisticated strategies to attract and maintain customers. An approach to this issue is to improve the conversion rate, which is defined as the ratio between completed transactions and the total number of transactions; it also accounts for abandoned transactions. Studies from key players in the market suggest that User Experience (UX) and User Interface (UI) have a major impact on the conversion rate. User-friendly UIs and simple UXs ensure the users are engaged and attached to the website [4]–[6], and in the case of e-commerce, this improves the likelihood of a sale. We target e-commerce websites to test and validate our approach because of the availability of data and expert knowledge, the relative simplicity and consistency of layouts, and the presence of a parameter measuring the user experience (i.e. the conversion rate) we aim to maximize.

A. System Overview

The process of changing an existing web UI to a more appealing one is often time consuming and cumbersome, because it involves several subjects and it requires several stages, such as assessment of current interface, change proposal, approval, implementation, verification, and deployment. The system we propose addresses this issue by reducing the effort of designers in implementing web layout improvements, allowing them to focus on more challenging and higher-level aspects of the UX process. This also provides an opportunity to combine HCI principles with image and text processing techniques into the field of automated UI design. To address this issue, we propose a system that:

- identifies the type of website;

- builds a model of the website layout;
- compares the computed model against optimal reference models, providing a quantitative evaluation for the current design;
- and provides a recommendation to improve the UI based on the expert rules.

In the case of e-commerce, we focus on the placement of the checkout button, which has a significant impact on the conversion rate. The checkout button's optimal placement follows different expert rules (aka. rule set) based on the type of goods being sold online. The above scenario allows for a complex system that combines various subsystems to achieve a certain target. The system uses a classifier, pattern matching algorithm, image modelling algorithm, and an optimization algorithm. The system is partitioned in to the following modules:

1. Website capture.
2. HTML code extraction.
3. Screenshots capture.
4. HTML based website classification
5. UI elements identification via image segmentation and pattern recognition.
6. UI model generation.
7. Evaluation of the current layout.

Major contributions of this work are in modules 4, 5, 6, and 7. In the fourth module we include a lightweight classifier that helps identify the type of website using only the HTML code of the website. The image pre-processing module requires combining various segmentation and pattern matching algorithms to achieve a comprehensive module that can identify and locate all the relevant interactive UI elements in the website layout. For the image modeling, we developed a novel modeling technique and algorithm that are based on human visual perception. Finally, the evaluation module involves complex model matching and comparison techniques to compute the UI score and provide recommendations for improvement.

The system we developed is generic and may find application in a wider spectrum of scenarios. The system can be extended to model and adjust specific features of any graphical interface. The role and nature of the image acquisition, classification based on metadata, image processing, and model generation would not differ, as they can be used to describe the relevant elements in other interactive system. It is important to note that the description is based on human visual perception rather than computational models. The system also relies on expert knowledge that we encode in the last two modules of the system.

II. Related Works

When supporting the optimization of distinct types of website following heterogeneous design rules, it is essential to compare the website layout against the correct reference model. This is automatically achieved using a two stages classifier, using keywords from the HTML content to describe the webpage. First, the website is scrapped to obtain the keywords, which are then used to perform the classification. For the scrapping, the Document Object Model (DOM) tree based approach [7] provides satisfactory results, while we select the bag-of-words model for the text based classification [8], [9]. Since the early 1990s, it has been shown that any approach that involves text classification requires hard categorization of keywords [10]. A bag-of-words approach would satisfy this criterion because it results in a histogram that describes all the words in the text, along with their count.

Also, the identification of a website's interactive elements, such as buttons and forms, requires two steps. First, the layout is segmented and then we use pattern recognition to find which segment is associated with UI elements. For segmentation we found that Felzenszwalb's algorithm [11] provides the better performances among those considered in a preliminary study targeting websites images. This is visible in Fig. 1, where the segmentation results are provided also for Simple Linear Iterative Clustering (SLIC) and Watershed segmentation algorithms [12], [13]. It is evident that Felzenszwalb's algorithm segments the image along edges and borders that describe UI elements, whereas SLIC does not identify the background and a large element, resulting in an excessive segmentation of the image. After segmenting, pattern recognition is used to identify each segment.

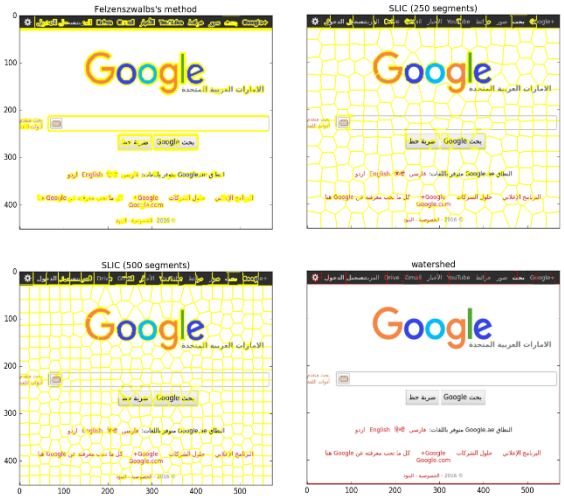

Figure 1. Comparison of three segmentation techniques: Felzenszwalb's algorithm (top left) performs better than SLCI (top right and bottom left) and better than Watershed algorithm (bottom right). The segments borders are shown in yellow.

Scale Invariant Feature Transform (SIFT) method is a key-point matching algorithm that is scale and orientation invariant and capable of matching key-points across multiple images [14]. Although faster pattern recognition algorithms exist, such as Speeded-Up Robust Features (SURF) [15], which is more suitable for real-time applications, SIFT provides higher accuracy, which is the key selection criteria. However, in some cases, key-point matching does not perform well, especially when the template lacks in feature complexity and in complex edges. In such case, alternative

methods include Optical Character Recognition (OCR) of text (if text exists) [16] or analysis of color histograms.

For UI modeling, there are graph based approaches [17] that are easily combined with pattern matching. Deep Neural Networks with Markov Random Fields [18], and Long Short-Term memory [19] have been used in image modeling, but they have been shown to work best in regenerative image processing rather than visual based modeling. Other works in image modeling include the Fisher Vector representation, which uses a set of low level descriptors to generate a global model [20], [21]. In the literature, there is lack of image modeling based on visual perspective, which can be employed in our system. The importance of the visual based model is essential in this work. The UI design highly depends on the human perception of the website layout, which if considered for, would invalidate the model itself and the overall approach.

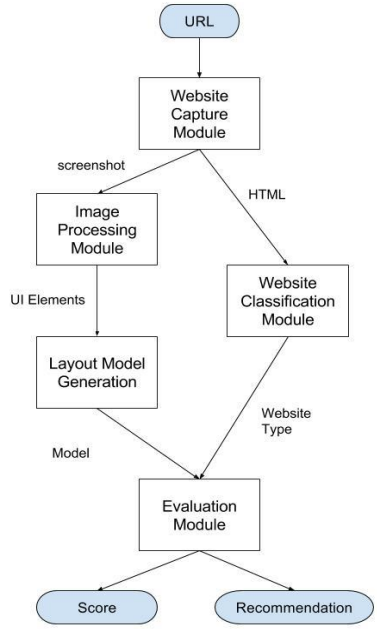

Figure 2. Flowchart illustrating the dataflow and dependencies between the five key modules of the system.

TABLE I. E-COMMERCE CLASSIFICATION CATEGORIES AND KEYWORDS

Category	Keyword
Airlines	*airline, boarding, ticket, tickets, travel, airplane, fight, flights, booking*
Hotels	*hotel, hotels, room, rooms, night, nights, booking*
Tickets	*ticket, tickets, cinema, show, shows, performance, performances, movie, movies*
Food	*food, delivery, meal, meals, order, combo*
Generic	*generic, electronics, flowers, fashion, kids, delivery, phone, TV, computer, toy, toys, flower, florist*

As mentioned earlier there is a strong correlation between a proper UI layout and usability [1], [2], [4]–[6]. Previous works have proposed fuzzy logic to control the UI and dynamically mutate the layout in real-time [22], [23]. These demonstrate that fuzzy logic control is suitable to compare a UI design given a set of design rules. The importance of controlling the UI can also be derived from the three paradigms of HCI [24].

In particular, the first principle states that human computer interaction is a classical cognitivism and information processing problem which requires the understanding of how one influences the other to properly understand the interaction between both. Understanding this interaction allows some systems to automatically change to adapt, in real-time or offline, as we propose in this work.

III. INTERFACE OPTIMIZATION ALGORITHM

The entry point of the system, as visible in the dataflow of Fig. 2, is the URL of a website. This represents the input for the website capture, which includes both the HTML code (i.e. the meta-data) and the screenshot image. These represent respectively the input of the classifier module and the image processing module. The image processing module produces a list of relevant elements in the image, which are fed to the model generation module. The computed visual model and the website type are sent to both the evaluation module and optimization module, which are also provided with a set of website type-dependent rules on the optimal layout properties. The result of the system is a quantitative evaluation of the original layout, and a set of recommendations for improvement based on the input rule set.

A. Website Capture Module

The website capture module is the entry point to the system. It takes a website URL, and it fetches the HTML code of the website using an HTTP request. The code is then passed to the website classification module. We use a webdriver (such as Selenium webdriver) to render the HTML and obtain an image of the webpage. Then the webdriver produces a screenshot of the webpage with a parametric geometry and passes it on to the following image processing module. This approach shows its main limitation with websites that dynamically generate some of the elements after delivering the HTML webpage. This would result in these elements not being captured in the initial HTML request and resulting in high likelihood of misclassification.

B. Web Classification Module

The classification algorithm developed in this system is based on the bag-of words approach. This approach was chosen because it provides high accuracy with text based classification. Before applying the bag-of-words approach, the HTML text is cleaned from tags, scripts, and styles. It is then passed to a natural language processing toolkit to remove simple words such as “and”, “or”, “with”, “is”. These words are known as stop-words and are irrelevant for classification purposes. The bag-of-words approach generates a histogram of frequencies for each keyword. Based on this histogram, a score is given to each category based on the number of words assigned to each category. The categories requiring different layout optimization rules in our proof-of-concept application to e-commerce website are shown in Table 1 together with the related keywords

used for classification. These categories were also chosen because the vast majority of e-commerce websites fall into one of these. As for the keywords, they were selected based on experimental results with a website test set, and are sufficient to describe and correctly classify websites of each category. The classifier then returns the type of website and passes it to the evaluation and optimization modules.

C. *Image Processing Module*

This component of the system processes the screenshot of the website (checkout page in particular) captured in the previous stage. At first, the algorithm segments the image using Felzenszwalb's algorithm, as in Fig. 3. Then, each segment is analyzed separately to identify whether it contains any UI relevant element. Pattern matching is used to identify elements of unique layout, such as the Visa Checkout button which is used for validation. The matching process is based on SIFT to find key-points and on Random Sample Consensus (RANSAC), which uses Homography to find the most likely correct key-points. The module reiterates over each segment and tests the different templates. To pass the test, templates need at least 10 matched key-points and a statistically sufficient number of good matched key-points defined by Lowes ration test [14]. Lowe's test indicates that more than 70% of the key-point descriptor mask must match. Then the segment is localized in the image frame and we extract the features listed in Table 2.

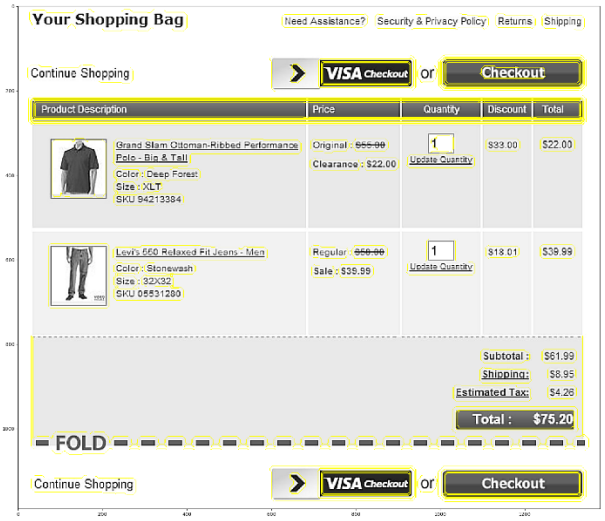

Figure 3. A sample e-commerce website's checkout page segmented using Felzenszwalb's algorithm.

TABLE II. FEATURES EXTRACTED FROM THE SEGMENTS KEY DESCRIPTORS AND THE METHOD OF EXTRACTION

Feature	**Method of Extraction**
Type	From template type
Boundaries	Maximum and minimum key-descriptors positions
Center Coordinates	Calculated from boundaries
Dimensions	Calculated from boundaries
Text (if applicable)	OCR
Colors	Color histogram analysis

The key-points based approach, works only with elements that contain a larger number of key-descriptors, which are usually detected in edges-rich segments. For simpler elements, such as basic buttons, we take another approach. First, a search for rectangular boxes is performed. Boxes have been shown to represent the majority of buttons and one of the most suitable shapes for a button [4]–[6]. After discarding all segments that do not contain such elements, the remaining elements are analyzed using OCR, looking for the strings: "Checkout", "Continue", and "Payment". These words usually identify UI elements for progressing towards the completion of the transaction [4]–[6]. For product-related images, we take a different approach, performing a color variance analysis in each segment. Those with high variance likely contain an image of the product being purchased. After performing the analysis on each segment, we extract the features listed in Table 2.

D. *Model Generation Module*

After receiving the UI image elements from the image processing, this module generates a graph based model that describes the relation between the image elements. The model can be described as a set of nodes connected by links. Each node in the model will describe an element in the UI. The node structure will be as such:

- Node Type: (Checkout or Continue or Image) (text)
- Node ID: (number)
- Node Coordinates: (x, y) (number, number)
- Node Boundaries: (up, down, left, right) (number, number, number, number)
- Node Dimensions: (h, w) (number, number)
- Node Text: (text)
- Node colour: (number)

The connections between each note will consist of the distances between each centre and the direction, which results in a 2-dimensional vector. Any node can have any arbitrary number of links to all the other nodes describing the relative placement of each. Consequently, comparing any two models will involve comparing the links and nodes. As such, comparing two links can only be done if both links connect two nodes of the same type. The comparator computes differences in the distance and direction (a difference vector). The comparator also calculates the differences between nodes as well; two "checkout" nodes can be compared and the result would also be a comparison between the different node components (text, dimensions, coordinates, and color). Fig. 4 shows the resulting model after the segmentation performed as in Fig. 3.

As visible in Fig. 4, the links describe the relation between each relevant element whereas each node describes each relevant element. The website modelling technique we propose here represents only elements that are relevant to a human observer, which is a significant advantage over those

proposed in literature. However, extracting such information can be challenging: experiments, tests and data analytics could indicate which elements are more relevant to a human observer in certain situations. In e-commerce in particular and when addressing the conversion rate problem, it has been shown that the layout of the checkout button, continue shopping button, back button, and the shopping items have the greatest impact on the conversion rate [1], [2], [4]–[6].

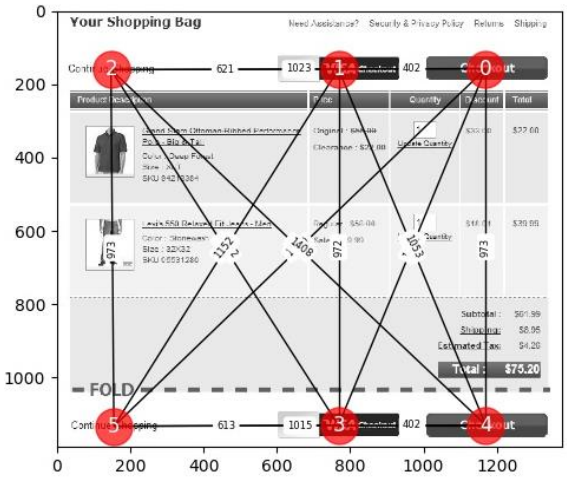

Figure 4. A sample graphical representation of the model based on the segments in Fig. 3

E. Scoring and Evaluation Module

After computing the nodes and links of the model, the evaluation module processes it according to the website type and according to the provided expert rules. These rules are parsed according to the follow syntax:

1. *Colour intensity* of X *with respect* to Y
2. *Location* of X *relative to* Y (above, below, right of, left of)
3. *Text* of X *contains* "text"
4. *Distance between* X and Y
5. X *alignment with respect to* Y (center align, right align, left align)
6. *Size* of X *with respect to* Y (same, larger, smaller)

X and Y in the rules indicate two different nodes (elements) identified by the node type and optionally ID, the keywords in bold help the parser identify the two nodes in question, and the words in italic help identify the property being measured. Both X and Y use the node type as a method of identification since generally in e-commerce websites elements, such as the checkout button, are not redundant in the UI. The evaluation module utilizes these rules to assess the generated model. This is done comparing each parsed rule with all nodes and links, and checking if these nodes and links match the rule. If so, a score is incremented. Nonetheless, the system also requires a small margin of error to account for minor inaccuracies in the system. The evaluation is presented with a score out of 100.

The evaluation module also computes the differences between the desired reference model generated by the rules and the actual model and then uses the differences to generate a set of recommendations to improve the UI. The recommendations indicate the rule broken and show the required change to satisfy the rule.

IV. Implimentation and Results

The system discussed above was implemented as open-source software written in Python[1]. The system has been tested on both Windows 10 and Linux Ubuntu 14.04.1. In the partially-optimized current implementation modules are executed as parallel threads when possible, and verbosity level of the screen output can be controlled to reduce the execution time. For the website capture we use the Python libraries urllib2 to retrieve the html code and Selenium webdriver to get a screenshot of the website rendering. However, due to the dynamic scope and different configurations of individual websites, the automatic navigation to the checkout page was not always possible. Therefore, we include an extra input parameter which represents the URL of the checkout page.

In the classification module, the HTML code obtained is pre-processed using bs4 BeautifulSoup, re, collections, and nltk. Bs4 and re are used to clean the code from tags, scripts, and styles. And the natural language processing toolkit ntlk is employed to remove stop-words. The histogram is generated using the collections Counter method. The implementation of this module has shown a 100% accuracy with a test set of 20 valid URLs and 3 outlier URLs. As mentioned before, the limitation of this approach is the incompatibility with dynamic HTML documents. Any change in the HTML after the code is fetched is not considered for evaluation.

The image processing module is the most computationally intensive module. The average execution time is about 90 seconds across all websites. The module integrates OpenCV (for SIFT and Homography), PIL Image (for image transformations), and skimage (for image segmentation). The OCR functionality is implemented using the tesseract-ocrwrapper of the Tesseract OCR engine.

TABLE III. IMAGE PROCESSING AND MODELING RESULTS

Sample	**Correct number of Elements**	**Obtained number of Elements**	**Model Accuracy**	**False Positives**
Sample 1	3	3	***100%***	-
Sample 2	4	4	***100%***	-
Sample 3	3	3	***100%***	-
Sample 4	6	4	***100%***	-
Sample 5	2	3	66%	***1***
Sample 6	4	7	57%	***3***
Sample 7	2	2	***100%***	-
Sample 8	1	1	***100%***	-
Sample 9	4	3	50%	***1***

Image processing produced an overall accuracy of 95%. This is due to some text elements being identified as buttons.

[1] Available on https://github.com/aounlutfi/E-commerce-Opimization

The pattern matching could detect all Visa Checkout buttons with 100% accuracy. The accuracy of the computed model with respect to the website layout is strictly dependent on the performance of the element identification in the image processing module. Table 3 shows the model accuracy for each of the 9 samples used to assess the system. As it can be seen in the table, most of the samples were 100% accurate. Three samples present false positive mostly due to elements (especially text) visually similar to buttons. These false buttons can be considered constructively towards the UI improvement goal. In fact, these suggest that the website may contains visual elements that can be mistakenly identified by some user. The data set used to assess the image processing and modeling algorithms is composed of 9 sample screenshots, one of which is an ideal implementation (template) of the Visa Checkout button.

After computing the model of the website's layout, the system evaluates it against the following rule set (provided just for validation):

- element 1 is visa checkout
- element 2 is checkout
- element 1 right of element 2
- element 1 less than 600px from element 2
- element 2 contains "checkout"

As expected, the template Visa Checkout implementation scored 100%, where as other implementations resulted in a lower score. It is interesting to note that despite some implementations were correct for their own website type, when compared to this specific ruleset for the "generic" website type, they produced a lower score. This highlights the importance of the classifier to identify the type of website using the correct set of rules for that type. The novelty of our approach and the lack of similar modelling techniques of graphical layout make it difficult to compare our system against others. However, our evaluation is reliable as it was carried out using reference data assessed by experts.

V. CONCLUSION

In this paper, we presented a generic method to optimize the design of web interfaces by finding the placement of interactive elements that maximize measurable usability parameters. The method has been implemented as an open-source software and customized to automatically optimize the position of the checkout button in several types of e-commerce websites using Visa Checkout guidelines. The algorithm is based on the combination of several image processing techniques and a novel perceptually-relevant visual model of the website's layout. Results have shown that the system operates at a high degree of accuracy. In future work, we will address technical improvements of the system. The current image processing module can be enhanced to improve accuracy, reduce execution time, and detect more complex elements such as tables, fields, and text blocks. This can be achieved by optimizing the current implementation or by integrating additional image processing techniques. The evaluation module will be extended to automatically modify the HTML code to implement the repositioning of the interactive element following the expert knowledge embedded in the system. Finally, the current implementation can be extended to support the simultaneous optimization of multiple interactive elements, suitable for more complex application scenarios and advanced interactive systems.

REFERENCES

[1] J. D. Gould and C. Lewis, "Designing for usability: key principles and what designers think," *Communications of the ACM*, vol. 28, no. 3, pp. 300–311, 1985.

[2] B. Shneiderman and C. Plaisant, *Designing the User Interface: Strategies for Effective Human Computer Interaction*, Fourth. Pearson Addison Wesley, 2004.

[3] "Worldwide Retail Ecommerce Sales Will Reach $1.915 Trillion This Year," *eMarketer*. [Online]. Available: https://www.emarketer.com/Article/Worldwide-Retail-Ecommerce-Sales-Will-Reach-1915-Trillion-This-Year/1014369. [Accessed: 05-Mar-2017].

[4] C. Harshman, "A/B Test Ideas for E-Commerce Call to Action Buttons," *Optimizely Blog*, 07-Dec-2013.

[5] C. Holst, "Fundamental Guidelines of E-Commerce Checkout Design," *Smashing Magazine*, April 2011.

[6] P. Laja, "How to Design an Ecommerce Checkout Flow That Converts," *ConversionXL*, February 2014. .

[7] V. B. Kadam and G. K. Pakle, "A Survey on HTML Structure Aware and Tree Based Web Data Scraping Technique," *International Journal of Computer Science and Information Technologies*, vol. 5, no. 2, pp. 1655–1658, 2014.

[8] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2–3, pp. 146–162, 1954.

[9] Y. Zhang, R. Jin, and Z.-H. Zhou, "Understanding bag-of-words model: a statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1–4, pp. 43–52, 2010.

[10] F. Sebastiani, "Machine Learning in Automated Text Categorization," *ACM Computing Surveys*, Oct. 2001.

[11] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[12] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[13] Jianbo Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[15] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, 2006, pp. 404–417.

[16] R. Smith, "An overview of the Tesseract OCR engine," in *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, 2007, vol. 2, pp. 629–633.

[17] P. Foggia, G. Percannella, and M. Vento, "Graph Matching and Learning in Pattern Recognition in the Last 10 Years," *International Journal of Pattern Recognition & Artificial Intelligence*, vol. 28, no. 1, p. 1, Feb. 2014.

[18] Z. Wu, D. Lin, and X. Tang, "Deep Markov Random Field for Image Modeling," in *European Conference on Computer Vision*, 2016, pp. 295–312.

[19] L. Theis and M. Bethge, "Generative image modeling using spatial lstms," in *Advances in Neural Information Processing Systems*, 2015, pp. 1927–1935.

[20] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *International journal of computer vision*, vol. 105, no. 3, pp. 222–245, 2013.

[21] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv:1312.6034*, 2014.

[22] A. Agah and K. Tanie, “Intelligent graphical user interface design utilizing multiple fuzzy agents,” *Interacting with computers*, vol. 12, no. 5, pp. 529–542, 2000.

[23] S.-M. Chen and J.-M. Tan, “Handling multicriteria fuzzy decision-making problems based on vague set theory,” *Fuzzy Sets and Systems*, vol. 67, no. 2, pp. 163–172, Oct. 1994.

[24] S. Harrison, D. Tatar, and P. Sengers, “The three paradigms of HCI,” in *Alt. Chi. Session at the SIGCHI Conference on Human Factors in Computing Systems San Jose, California, USA*, 2007, pp. 1–18.