Autocompletion as a Basic Interaction Concept for User-Centered AI

Florian Lehmann
florian.lehmann@uni-bayreuth.de
Research Group HCI + AI,
Department of Computer Science,
University of Bayreuth
Bayreuth, Germany

Daniel Buschek
daniel.buschek@uni-bayreuth.de
Research Group HCI + AI,
Department of Computer Science,
University of Bayreuth
Bayreuth, Germany

ABSTRACT

With this position paper we propose that autocompletion can be interpreted as a basic interaction concept in the interaction between humans and systems using artificial intelligence (AI). Autocompletion is well known from text input where the system predicts intended user input, e.g. in search engines. In our research on human-AI collaboration we observe parallels to such textual autocompletion but in different application contexts, such as text generation, mock-up generation, and layout solvers. We compare exemplary related work to highlight autocompletion as a reoccurring and reusable interaction concept. We discuss that identifying underlying interaction primitives in user-centered AI can help to inform concrete design solutions for interactions and user interfaces, and could be a starting point for future research in this area.

CCS CONCEPTS

• **Human-centered computing → HCI theory, concepts and models**; *Interaction design theory, concepts and paradigms.*

KEYWORDS

autocompletion, interaction patterns, human-AI interaction, user-centered AI

1 INTRODUCTION AND BACKGROUND

Autocompletion is well established nowadays, and is most likely used on a daily basis by everyone using modern online services, for instance in search engines. Autocompletion describes the approach to predict the intended user input from textual input (as seen in Figure 1, example 1). Search engines utilise this to optimise search queries. This can improve efficiency and supports users to formulate exact search queries. As the underlying key concept, the system extends partial user input (see Figure 1 blue areas) to make it more complete. The completed input is then presented to the user who can choose from multiple variants (see Figure 1 orange areas).

With this position paper we explicitly focus on autocompletion as a reoccurring interaction concept for interactions with an AI system. Here, we regard "intelligence" as the ability of such systems to reason on user input to rank and provide suggestions. We do not differentiate the technology used to implement such abilities, which could be, for example, neural net-based autocompletion or rule-based completions, such as n-grams.

We reviewed work on autocompletion as part of our research on topics at the intersection of HCI and AI: Already in 1986, Jakobsson [6] reported increased input efficiency compared to coded search queries in the example of searching a library system. More recent work on search queries [9] observed a relation between the likelihood of autocompletion use and the position of the next character on the keyboard. Other work modeled click behaviour to further improve interaction performance [7], and analyzed visual search with an eye-gaze study [4]. Further studies in context of autocompletion include creating XML documents [1, 8] and gestural interaction [2].

We are particularly interested in interactive systems that are "intelligently" informed to generate data. For instance, such systems might generate text [13], GUI prototypes [10], and layouts [3].

In summary, autocompletion is well established, provides interactivity, and intelligently predicts new text. Comparing this to other generative contexts we observe a similarity: Many prior systems share the underlying implied role of AI to make something more complete, which the user had already partially specified, as part of an ongoing interaction. Moreover, we found shared related interaction patterns. Thus, we propose to examine autocompletion as a basic interaction concept in human-AI interaction.

2 AUTOCOMPLETION IN PRACTICE

In this section, we describe the interactivity of generative approaches from related work in detail and the parallels to textual autocompletion. This serves as an analysis of underlying interaction concepts inherent to practical applications of human-AI interaction. We highlight specific interaction and UI patterns across the examples in this section:

- User interface: The interface holds a field for input. Generated objects are placed near the input.
- Workflow: One or more recommendations are generated interactively and are part of the workflow.
- User decision: The user can decide to accept a recommendation or not.
- Editing: The user can further edit the recommended object.
- Information: Partial user input serves as input for the system. The AI's prediction is conditioned on the input to extend it.

Code – We consider code to be a very structured and formal type of text. Code completion is a core feature of modern development

MuC'20 Workshops, Magdeburg, Deutschland

https://doi.org/10.18420/muc2020-ws111-328

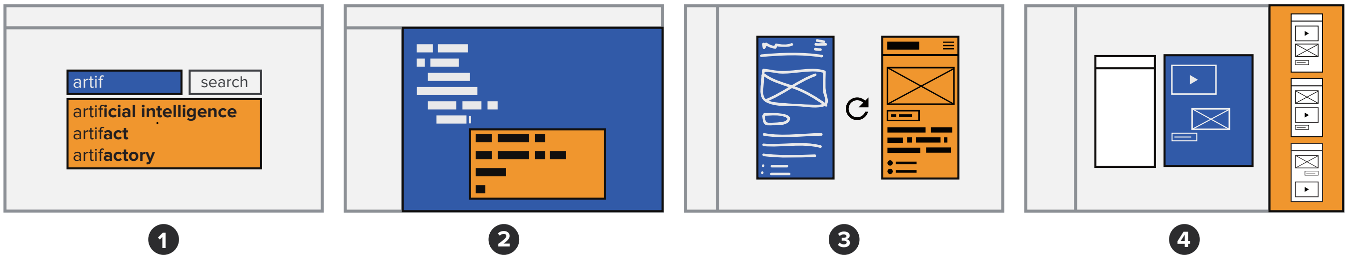

Figure 1: Visualisation of 1) search query autocompletion and three examples that share the same underlying interaction concepts, namely 2) code completion, 3) mock-up generation from sketches, and 4) layout solving. Coloured areas highlight similarities in the user interface. Blue areas with white symbols indicate fields for user input, orange areas with dark grey symbols indicate fields for completed input by the AI. Example 3 is inspired by [10], Example 4 is inspired by [3].

environments and close to the well-known textual autocompletion. However, research introduced AI-assisted code completions to predict intended input based on neural nets [11]. This is used by services like TabNine[1] and Kite[2]. These services do not only complete lines of text, they are able to complete whole functions by only inputting signatures:

Users enter code in a textarea in a code editor or IDE. While editing, a list widget appears near the cursor, e.g. below of it. The widget contains a list of recommendations. Depending on the chosen technique, list items are sorted, e.g. by prediction probabilities. The user can decide to accept a recommendation via keyboard shortcut or click. This then completes/inserts e.g. a variable name, method signature, or a whole function. See Figure 1, example 2.

Graphical User Interfaces (GUIs) – In contrast to code completion, image generation systems have not been widely implemented in current image editing or GUI design tools yet. However, research shows promising progress in this area: For example, partially drawn sketches can be recognised and completed [12]. Based on sketches, high fidelity examples can be retrieved [5]. Moreover, related tools are able to transform a low fidelity sketch into a medium fidelity mock-up [10], which could be used to design a GUI starting with sketches:

GUI sketches and text are added to a canvas with a pen or brush tool. While sketching, an icon appears next to the canvas. By clicking the icon, the sketch is used to derive a medium fidelity mock-up. This is presented on another canvas, next to the sketch. The mock-up consists of editable vector graphics. Here only one version is recommended, however, it would be possible to generate multiple variants of the mock-up, e.g. by intelligently combining elements. The user can decide to accept a recommendation by clicking on an accept icon, or by simply working in the mock-up's canvas field. See Figure 1, example 3.

Layout – Layout solving can be considered a special problem within designing a GUI. Elements in an UI can be arranged in various ways. Some variants work better than others. In particular it is a time consuming manual task and a tool called GRIDS was introduced to solve layouts interactively [3].

In such a tool, interactive layouting might be realised by selecting and placing loose UI elements on a workspace. Next to it is a canvas which is initially empty. By clicking on a button, the tool suggests possible layout solutions in a sidebar list which spans the complete height of the application. The user can scroll through the suggested layouts and can save or further edit them in the canvas. See Figure 1, example 4.

3 DISCUSSION AND CONCLUSION

Focusing on AI in generative roles, we found that key aspects of the well established autocompletion idea for text input can also be found across a wide range of application contexts and AI modelling goals in interactive systems. Based on this observation, we consider autocompletion as an example for an atomic interaction pattern.

Beyond the specific case of autocompletion examined here, we assume that identifying and making explicit other underlying interaction concepts can help to envision new applications and make research on interactions and UIs for human-AI systems more concrete and systematic. In this way we can substantiate research and might be able to adapt already existing methods for analysis. Such methods might be used as starting points to create models for evaluation, and to derive best practices for practitioners. For example, we could use basic concepts as a baseline for novel tools to prototype human-AI interactions to complement other approaches, such as planning for adaptive interfaces based on probability thresholds provided by intelligent systems [14]. This could help to standardise the design and interaction of human-AI interaction.

We conclude that common human-AI applications share similarities and that identifying these in existing applications provides one way to systematically approach the design of future interactions and UIs. Such reusable interaction patterns ideally might be transferred across domains. Ultimately, we could derive reusable design components for practitioners, and path the way for novel tools that combine the capabilities of humans and AI. Further empirical and experimental research is needed to find out how interaction patterns can best be identified, conceptually analysed, and potentially shared across application domains in human-AI interaction.

REFERENCES

[1] Serge Abiteboul, Yael Amsterdamer, Tova Milo, and Pierre Senellart. 2012. Autocompletion learning for XML. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data (SIGMOD '12)*. ACM Press, New York, NY, USA, 669–672. https://doi.org/10.1145/2213836.2213928

[2] Mike Bennett, Kevin McCarthy, Sile O'Modhrain, and Barry Smyth. 2011. SimpleFlow: Enhancing Gestural Interaction with Gesture Prediction, Abbreviation

[1] TabNine: https://www.tabnine.com, last accessed 10th June 2020.
[2] Kite: https://www.kite.com, last accessed 10th June 2020.

and Autocompletion. In *Human-Computer Interaction – INTERACT 2011 (Lecture Notes in Computer Science)*, Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler (Eds.). Springer, Berlin, Heidelberg, 591–608. https://doi.org/10.1007/978-3-642-23774-4_47

[3] Niraj Ramesh Dayama, Kashyap Todi, Taru Saarelainen, and Antti Oulasvirta. 2020. GRIDS: Interactive Layout Design with Integer Programming. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. ACM Press, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376553

[4] Kajta Hofmann, Bhaskar Mitra, Filip Radlinski, and Milad Shokouhi. 2014. An Eye-tracking Study of User Interactions with Query Auto Completion. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM '14)*. ACM Press, New York, NY, USA, 549–558. https://doi.org/10.1145/2661829.2661922

[5] Forrest Huang, John F. Canny, and Jeffrey Nichols. 2019. Swire: Sketch-based User Interface Retrieval. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM Press, New York, NY, USA, 1–10. https://doi.org/10.1145/3290605.3300334

[6] M. Jakobsson. 1986. Autocompletion in full text transaction entry: a method for humanized input. *ACM SIGCHI Bulletin* 17, 4 (April 1986), 327–332. https://doi.org/10.1145/22339.22391

[7] Yanen Li, Anlei Dong, Hongning Wang, Hongbo Deng, Yi Chang, and ChengXiang Zhai. 2014. A two-dimensional click model for query auto-completion. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (SIGIR '14)*. ACM Press, New York, NY, USA, 455–464. https://doi.org/10.1145/2600428.2609571

[8] Chunbin Lin, Jiaheng Lu, Tok Wang Ling, and Bogdan Cautis. 2012. LotusX: A Position-Aware XML Graphical Search System with Auto-Completion. In *2012 IEEE 28th International Conference on Data Engineering*. IEEE, Washington, DC, USA, 1265–1268. https://doi.org/10.1109/ICDE.2012.123 ISSN: 2375-026X.

[9] Bhaskar Mitra, Milad Shokouhi, Filip Radlinski, and Katja Hofmann. 2014. On user interactions with query auto-completion. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval (SIGIR '14)*. ACM Press, New York, NY, USA, 1055–1058. https://doi.org/10.1145/2600428.2609508

[10] Vinoth Pandian and Sarah Suleri. 2020. BlackBox Toolkit: Intelligent Assistance to UI Design. In *CHI'20, Workshop on Artificial Intelligence for HCI: A Modern Approach*.

[11] Alexey Svyatkovskiy, Ying Zhao, Shengyu Fu, and Neel Sundaresan. 2019. Pythia: AI-assisted Code Completion System. In *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. ACM Press, New York, NY, USA, 2727–2735. https://doi.org/10.1145/3292500.3330699

[12] Caglar Tirkaz, Berrin Yanikoglu, and T. Metin Sezgin. 2012. Sketched symbol recognition with auto-completion. *Pattern Recognition* 45, 11 (Nov. 2012), 3926–3937. https://doi.org/10.1016/j.patcog.2012.04.026

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Long Beach, California, USA, 6000–6010.

[14] Qian Yang, John Zimmerman, Aaron Steinfeld, and Anthony Tomasic. 2016. Planning Adaptive Mobile Experiences When Wireframing. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems - DIS '16*. ACM Press, New York, NY, USA, 565–576. https://doi.org/10.1145/2901790.2901858