Huy Viet Le, University of Stuttgart
Sven Mayer, LMU Munich
Niels Henze, University of Regensburg

Deep Learning for Human-Computer Interaction

Insights

- → Deep learning enables novel systems by moving from handcrafted algorithms to data-driven approaches.
- → Determining a solution's qualities is easy for most disciplines but much harder for HCI, as it needs to conduct a user study.
- → The user-centered design process must be adapted by incorporating data collection and iterative model development.

IMAGE BY BY ZEBERW / SHUTTERSTOCK.COM

Early and continuous user involvement has a truly long tradition in human-computer interaction (HCI). Iterative human-centered processes are widely used in academia and industry to ensure users' needs are considered throughout the design and development processes. The user-centered design (UCD) process [1] depicted in Figure 1 is a highly influential example of such a process and has proven to be flexible when new tools, such as study, prototyping, or development methods, become available.

The rise of machine learning, especially deep learning [2], poses new challenges for human-centered design [3]. Deep learning currently changes work in many disciplines, such as computer vision, natural language processing, and brain-machine interfaces. These disciplines went from handcrafted algorithms to data-driven approaches for building novel systems. In expensive iterative processes, machine learning models are trained and fine-tuned, which is possible only because evaluation is very cheap. In contrast to HCI, such models typically do not follow human-centered approaches, as the developed solution can be evaluated using simple metrics.

HCI requires users to determine the quality of a solution, which turns out to be expensive, as this typically means conducting a user study. When using machine learning, improving a solution requires training a new model, as data-driven models cannot be changed the way that handcrafted ones can. Training a machine learning model requires data, which in HCI typically means that UCD's design-solution step suddenly requires studies for data collection.

Over recent years, we used deep learning to develop a large number of prototypes. An example is our work on the design and recognition of gestures on the back of a smartphone using deep learning [4], where we use convolutional neural networks to identify gestures using the raw data of capacitive sensors. Building the system required an additional data-collection study beyond the studies one would typically conduct when following a UCD process. The data collection was followed by iterative and laborious training, using classification accuracy as a proxy for the system's qualities. Accuracy, however, did not tell us about users' efficiency, effectiveness, and satisfaction when using the solution. The true qualities can be inferred only through another study. This is especially true when users can adapt to the system by, for example, holding their phone differently or changing the position of their fingers. This makes training and testing a model on splits of the same dataset, which is ubiquitous in other branches of machine learning, error prone.

In the following, we take a look at the development process used in HCI and in ML. We propose to unify the processes to fully incorporate deep learning into the development of interactive systems.

MACHINE LEARNING IN A USER-CENTERED DESIGN PROCESS

Previous work in HCI developed countless interactive systems based on UCD. The main process consists of four phases of an iterative design and development cycle to develop interactive systems with a focus on usability (Figure 1). The process consists of phases for understanding the context of use, specifying the user requirements, developing a solution, and, finally, evaluating against the initial system requirements. When reaching the last phase, developers can cycle through the process again, adjusting not only the solution but also what was learned about context and requirements. Cycling iteratively through the design process, a solution evolves toward one that satisfies the users' needs. UCD assumes that user requirements can be unambiguously translated into a working prototype. Indeed, previous work commonly identified the needs and requirements of interactive systems and prototyped them using handcrafted algorithms, which range from simple value comparisons, thresholding, and transfer functions through computer-vision techniques to, for example, kinematic models [5].

With the advent of deep learning, complex relationships and patterns in data can be learned by algorithms, outperforming traditional feature-development approaches. Deep learning algorithms learn features directly from data. They do not require domain knowledge about specific sensors yet are still more accurate when enough data is available. Enough data, however, often means truly large amounts of data, which for HCI often means truly large user studies. In addition, deep learning produces black-box models that can hardly be understood by developers. As we do not understand how and why a deep learning model works, the model needs to be trained, tested, and validated through multiple iterations until it achieves the desired result. In contrast, the UCD process describes the design of a solution in a single step without involving potential users, an evaluation of its usability in a subsequent step, and a full refinement in a further iteration. Due to the effort for gathering data and iterative model development, the UCD process needs to be refined to incorporate the iterative development and testing of a model, as well as evaluating the model's usability within the development process. In particular, the designing-solution step needs to incorporate the modeling cycle of a deep learning process and connect it to the usability aspects of the user-centered design.

LIMITATIONS OF COMMON DEEP LEARNING PROCESSES

A typical process for developing and evaluating deep learning models consists of three phases, depicted in Figure 2. First, we need a representative dataset that in HCI can be gathered through a data-collection study. The dataset should be representative of a wide variety of users and different contexts to capture a wide range of variations. Afterward, we need to prepare the data and train the model. Training is repeated a hundred or even a thousand times to find the most suitable model structure and hyperparameters that lead to the lowest model error on a test set using trial-and-error or grid search. As thousands of iterations can result in overfitting to the test set, the model's generalizability must be evaluated using previously unseen data to assess whether the chosen model and hyperparameters were overfitted to the validation set or generalize to unseen data.

The deep learning community often uses a training-validation-test split when developing models. While the training set and the validation set are used for iterative model development, the test set is used for a one-time validation of the model. However, traditional machine learning evaluation metrics (e.g., accuracy, precision, recall, and error rate to describe how well the model generalizes to unseen data) alone do not describe the usability of a system. The main focus of the UCD process is to achieve a high usability. Instead of software metrics, factors such as the effect of inference errors on the usability, the model stability, and the usefulness of the investigated system should be considered. This can involve fundamental questions like how good the perceived usability is for a given use case and how impactful errors are and how noisy the estimations over time are for variations in the input data. As systems are used by a wide range of users and in different scenarios, the validation also needs to assess whether the model can generalize beyond the tasks used in a data-collection study. While previous work considered accuracies above 80 percent to be sufficient [6], sufficiency depends on the use case, which can be evaluated only through user feedback—for example, whether the action's consequence is recoverable and how much the consequences affect the user.

In summary, a typical process for deep learning describes the iterative nature of developing and evaluating black-box models. However, the deep learning development process does not consider the usability of the model and thus of the final system. To apply deep learning techniques in interactive systems, we need to refine and combine the UCD process with typical deep learning processes to consider both the iterative development and evaluation of models, as well as their usability within the final system to achieve the objectives of the system.

USER-CENTERED DESIGN PROCESS FOR DEEP LEARNING

In Figure 3, we present the *UCD process for deep learning* (UCDDL), which combines the UCD process with steps required for deep learning. The UCDDL consists of five phases. The first two phases are identical to the traditional UCD process, focusing on understanding users as well as specifying requirements. The next three phases focus on developing a prototype based on deep learning and evaluating not only the system as a whole but also the developed mode in particular. In the following, we describe the UCDDL.

Phase 1 and Phase 2. Phase 1 is about identifying users who will use the system and their tasks, and under which conditions they will use it (e.g., technical and ergonomic constraints). This step often consists of user studies to understand the context of use, such as observation and inquiry studies, or is simply based on findings from previous work. Based on the gained insights, phase 2 specifies the requirements that are the basis for development and evaluation. In this step, the need for deep learning is identified.

Phase 3: Collect data based on user requirements. Training a deep learning model requires a representative and large enough dataset as the ground truth. Gathering this dataset through a user study involves the design and development of an apparatus that runs mockup tasks to cover all expected interactions. The study should be conducted with a representative set of potential users who cover all relevant factors to collect a sufficient amount of data for model training. Instructing potential users to perform certain tasks with the mockups even enables the apparatus to label the collected samples automatically.

In case the final system does not generalize to new users and tasks in phase 5, an additional iteration of phase 3 is required. The dataset is the foundation of the developed system and needs to be refined. In this case, another data-collection study must be conducted, whereas the resulting new dataset can be combined with the already existing dataset. In addition, the data collected in the design-evaluation phase (see Phase 5) could also be used to extend the existing dataset.

An example to collect data for our gesture recognizer based on raw capacitive data would be to design tasks that represent common inputs on mobile touchscreens. A mockup interface that prompts the user to perform different gestures on the back of a smartphone enables collecting data that is representative of common touch input in the real world. A general suggestion would be to simulate the context of use identified in the previous phase. This helps to capture a distribution that covers most common but also edge cases that occur naturally. A concrete example is described in our paper on shortcut gestures [4].

Phase 4: Model development. Based on the dataset, the model-development phase applies deep learning to develop a model that is used by the system. Prior to the actual model training, the dataset often needs to be cleaned or augmented in case producing the desired amount of data is not feasible. Further, we first explore the dataset with techniques such as visual inspection and descriptive and inferential statistics (e.g., finding correlations), as well as applying basic machine learning models such as linear regression and SVMs using simple feature extraction. This step provides an overview of the dataset and helps to choose an initial model and hyperparameters. After data preprocessing and exploration, the dataset needs to be split into a training-validation split to avoid overfitting. Since the same user could generate highly similar data, the dataset should be further split by participants instead of by samples, as commonly applied. Previous work commonly used a rate of

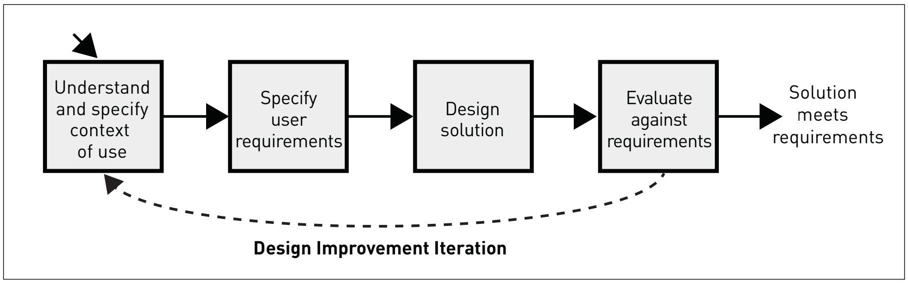
Understand and specify context of use
Specify user requirements
Design solution
Evaluate against requirements
Solution meets requirements
Design Improvement Iteration

Figure 1. The UCD process as described in ISO 9241-210 [1].

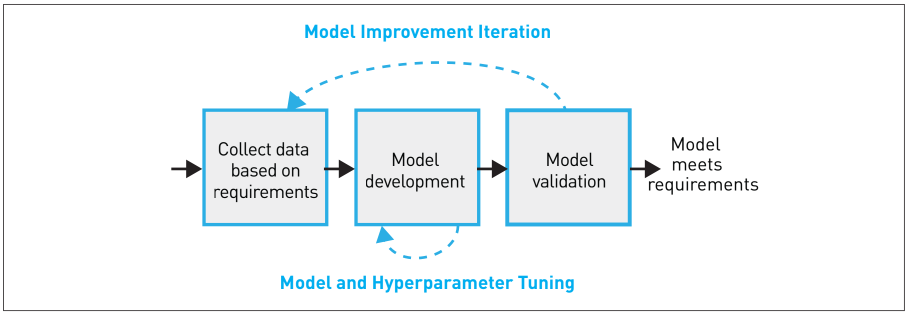
Model Improvement Iteration
Collect data based on requirements
Model development
Model validation
Model meets requirements
Model and Hyperparameter Tuning

Figure 2. The machine learning train-validation-test cycle.

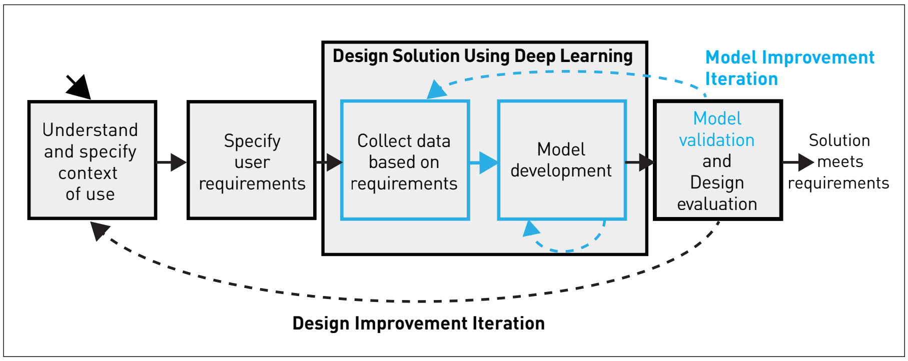
Design Solution Using Deep Learning
Model Improvement Iteration
Understand and specify context of use
Specify user requirements
Collect data based on requirements
Model development
Model validation and Design evaluation
Solution meets requirements
Design Improvement Iteration

Figure 3. Adapted UCD process for deep learning in the context of interactive systems in HCI.

80 percent/20 percent for a training-test split, and a 70 percent/20 percent/10 percent split for a training-validation-test split. While the deep learning community commonly uses a training-validation-test split to detect overfitting, the UCDDL process replaces the third set with a user study in the next phase. This has two advantages: First, the full new dataset can be used to train the model and test it based on the validation set. Second, the validation user study with new participants can serve as a new dataset. More important, the model's usability (and also the accuracy) can be evaluated in a realistic scenario based on feedback from potential users. This is not possible with a training-validation-test split, which focuses only on the modeling aspect.

Continuing the example described above, we would randomly assign each participant to either a training or validation set (with a common ratio of, for example, 80 percent/20 percent). Assuming we recruited 28 participants, we would use the data of 22 participants to train the model and the remaining six participants to validate the model. Those six participants would represent new users who were unseen during the training. The validation set can be used for hyperparameter sweeps, since any overfitting can be discovered in the subsequent evaluation study.

Phase 5: Model validation and design evaluation. To avoid potential overfitting, this phase of the system's evaluation has to be conducted with participants who did not participate in prior data-collection studies conducted for building the model (e.g., in Phase 3). The evaluation focuses on three aspects: 1) a model validation to achieve the same results as the commonly used training-validation-test approach (combined with training and test of the previous phase), 2) evaluating the model usability (and optionally also the model error) in a realistic but controlled scenario to focus on individual aspects, and 3) evaluating the system within a common use case (as specified in Phase 2) to assess the practicality of the system and the perceived usability of the model in an uncontrolled scenario.

The model validation replaces the test set based on tasks similar to those used in the data-collection study (Phase 4). In particular, data is collected with the same tasks, which, at the same time, can also be used to introduce participants into the system. This prepares the participants for the usability evaluation within realistic scenarios, which consists of a set of tasks that resemble a realistic use case. These tasks have to be designed to be controlled enough to enable a focus on individual aspects of the system (e.g., recognition accuracy and usability of certain classes of the model).

In our concrete example of gesture recognition, we would start with tasks that are very similar to the ones used in the data-collection study. This enables us to collect a test set that is comparable to the training and validation set but comes from previously unseen users. In addition, we would deploy our model in a proof-of-concept application that represents a use case that we identified in the first two phases. We instruct users to solve a realistic task while we capture the raw data as well as model predictions. This enables us to validate the model while we provide feedback to the user's input. This is especially interesting since the live model feedback allows us to study how users adapt to the model to achieve a higher level of accuracy.

SUMMARY

Deep learning opens up opportunities for researchers and practitioners. Myriad freely available libraries enable their users to train models with only a few lines of code. This is a promising alternative to handcrafting complex interactions but comes with many new challenges and pitfalls. We need large datasets that are representative, carefully designed studies that evaluate the interaction between user and model, and clear metrics that define when our system satisfies the user requirements. We presented a necessary adaption of the UCD process, which adds two new steps to build and validate systems on a data-driven basis. While best practices in deep learning suggest rigorous tests based on an existing dataset, we need user studies to understand many more factors that affect the user experience. How do we know how well a model performs in a realistic scenario without evaluating it with use cases? How do we know how users adapt to the model without testing it with potential users? The presented UCD process is the first step toward a user-centered method of building and evaluating interactive systems with deep learning.

Endnotes

1. International Standardization Organization (ISO). ISO 9241-210:2010 - Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems. 2010.
2. LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature 521*, 7553 (2015), 436–444; https://doi.org/10.1038/nature14539
3. Dove, G., Halskov, K., Forlizzi, J., and Zimmerman, J. UX design innovation: Challenges for working with machine learning as a design material. *Proc. of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, New York, 2017, 278–288; https://doi.org/10.1145/3025453.3025739
4. Le, H.V., Mayer, S., Weiß, M., Vogelsang, J., Weingärtner, H., and Henze, N. Shortcut gestures for mobile text editing on fully touch sensitive smartphones. *ACM Trans. Comput.-Hum. Interact. 27*, 5, (Aug. 2020), Article 33; https://doi.org/10.1145/3396233
5. Nancel, M., Aranovskiy, S., Ushirobira, R., Efimov, D., Poulmane, S., Roussel, N., and Casiez, G. Next-point prediction for direct touch using finite-time derivative estimation. *Proc. of the 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, 2018, 793–807. DOI: https://doi.org/10.1145/3242587.3242646
6. Kostakos, V. and Musolesi, M. Avoiding pitfalls when using machine learning in HCI studies. *Interactions 24*, 4 (Jul.–Aug. 2017), 34–37; https://doi.org/10.1145/3085556

Huy Viet Le is a researcher who received his Ph.D. in human-computer interaction from the University of Stuttgart. His research focus lies at the intersection of interactive systems and applied machine learning. He invents, builds, and evaluates new interaction techniques that notably improve the user experience.
→ mail@huyle.de

Sven Mayer is an assistant professor of HCI at LMU Munich. In his research, he uses machine learning tools to design, build, and evaluate future human-centered interfaces. He focuses on hand- and body-aware interactions in contexts such as large displays, augmented and virtual reality, and mobile scenarios.
→ info@sven-mayer.com

Niels Henze is a professor of media informatics at the University of Regensburg. He is interested in machine learning for interactive systems, new tools enabled by mixed reality, and the effects of avatars on users.
→ nhenze@googlemail.com

DOI: 10.1145/3436958