On the Cost of Interactions in Interactive Visual Machine Learning

Yu Zhang* University of Oxford
Bob Coecke† University of Oxford
Min Chen‡ University of Oxford

ABSTRACT

In interactive visual machine learning (IVML) processes, human-computer interaction provides a crucial means for model-developers to inject their knowledge, especially the knowledge that is not encoded in the training data, into the learning processes. In this paper, we discuss how the cost of interactions can be measured or estimated and how such measurements can be used in evaluating IVML processes. In particular, we present a practical solution for estimating the cost of interactions and demonstrate the use of such estimated measures in comparing different designs of an active learning system for reconstructing data from historical visualization.

Index Terms: Human-centered computing—Visualization—Visualization design and evaluation methods

1 INTRODUCTION

Interactive Visual Machine Learning (IVML) is a class of the practically viable machine learning (ML) methods, with which the human model-developers can use visualization to observe various aspects of the learning process, the training data and the evolving model; and can use interactions to steer the learning process. Since it would not be sensible to deny human children any access to their parents and teachers in their learning process, IVML is not in any way an inferior solution to fully-automated ML. In recent years, a good number of authors have reported their positive experience of adopting the IVML approach in developing analytical models (e.g., [1, 3, 4, 7, 8, 10]).

There are many metrics for evaluating ML models and fully automated ML processes [6]. All these can be applied to models learned using IVML methods and IVML processes. Tam et al. used an information-theoretic metric to estimate the human knowledge available to an IVML process through interactions, and demonstrated that an IVML process can benefit significantly from human knowledge, especially when a sparse or skewed dataset is used in model training [11]. While their work was based on the cost-benefit metric proposed by Chen and Golan [2], they did not measure or estimate the cost of interactions, which would provide a more balanced evaluation of the two IVML processes in [11]. In general, it is necessary to assess both the benefit and the cost of interactive visualization activities. In this short paper, we focus on measuring and estimating the cost of interactive activities.

In recent years, many IVML solutions have emerged in the literature. User interfaces with sophisticated interaction designs and visual representations have been used to support ML methods such as neural networks [8], decision trees and random forests [7, 10], topic modeling [3], and clustering [5]. The evaluation metrics shown in these works include conventional ML performance metrics [3, 8, 10], and user feedback and testimonies [3, 5, 7]. However, to our best knowledge, there has not been any effort for measuring the cost of interactions.

*e-mail: yu.zhang@cs.ox.ac.uk
†e-mail: bob.coecke@cs.ox.ac.uk
‡e-mail: min.chen@eng.ox.ac.uk

2 MEASURING THE COST OF INTERACTIONS

In this paper, we define an *interaction* as a set of actions in relation to a human input action for entering a piece of information. These actions may include for a computer to display some information as a prompt for interaction, and for a user to use an input device to enter some information. Here a sequence of inseparable actions (e.g., moving scroll bar or dragging-and-dropping an object) is considered as a single interaction, while some semantically-related actions (e.g., selecting a radio-button option, and then pressing a [yes] button) are considered as separate interactions.

2.1 Metrics, Direct Measurement, and Challenges

The cost of interactions can be affected by many factors, including:

- the number of input actions;
- the time used for interactions;
- the difficulty level of interactions;
- the cognitive load of interaction (e.g., measured using electroencephalography);
- the cost related to the input modality (e.g., mouse, keyboard, gesture, voice, etc.);
- the cost of learning to perform an interaction;
- the adverse cost of an input error;

It is also necessary to consider the impact of interactions on the overall learning processes and the learned models. In addition to many conventional metrics (e.g., accuracy, precision, recall, F1-score, etc.) for fully automated ML processes and the learned models, one may consider the following aspects of impact in comparison with a fully automated ML process:

- Do interactions reduce or increase the biases of the learned models?
- Do interactions reduce or increase the amount of data required for an ML process, and/or the cost of data labelling?
- Do interactions reduce or increase the convergence rate of the evolving model?
- Do interactions reduce or increase the overall time of a model-development project (from the initial problem specification to the delivery of a working model)?
- Do interactions reduce or increase the overall cost of a model-development project (e.g., human costs for data collection, labelling, model template design, and iterative design refinement, etc.)?
- Do interactions reduce or increase the human users' confidence or trust in the learned model or the associated learning process?

Ideally, one would wish that many of such measurements can be obtained from various forms of empirical studies, or good record-keeping for longer-term measurements (e.g., human effort and costs) in model-development projects. However, there are a number of challenges in these ways of conducting evaluation:

(a) Since one of the main purposes of interaction in IVML is to allow trainers to inject their knowledge into the learning process, replacing them with inexperienced participants in a controlled experiment may make the benefit of knowledge difficult to assess.

(b) Many ML processes are sensitive to the ordering of training data. Interactions may change such ordering, while the effect of interactions may also depend on the ordering. In traditional empirical studies, one typically uses a combination of randomization and repeated measures to alleviate an ordering confounding effect. This approach would require an IVML session to be very short in order to take repeated measures, and would restrict the design space of such studies significantly as most IVML sessions are more complex than typical trials in empirical studies.

(c) A project for developing a working ML model in a practical application typically takes many months from the initial problem specification to model delivery. For such a project to embrace both automated ML and IVML workflows is already a non-trivial demand. Any effort for good record-keeping about the time, cost, and impact during the project period will add further burden and complexity to the management of the project.

(d) One may consider obtaining some qualitative or subjective measures by, for instance, conducting surveys and small group discussions. While these are no doubt useful means of evaluation in general, the opinions gathered can be easily influenced by the characteristics of the data, ML method used, tasks of the model, the knowledge and experience of the trainers and so on.

(e) The records of model-development projects and the results of less-tightly controlled empirical studies (e.g., qualitative studies) can be difficult to reproduce, and this may hinder meaningful comparison with future work;

2.2 Simulation-based Estimation

One alternative to direct measurement using empirical studies and record-keeping is to simulate the interactions that may be used in IVML sessions, and compile statistical measures from the simulation results. Simulation of the cost of interaction is essentially a computable function $F : I \times O$, where I is one or a set of input variables and O is one or a set of output variables. The input variables define the independent variations of the operational environment used for comparing different IVML sessions, including for instance, different designs of the user interfaces and/or visual representations for supporting IVML, different types of information available to the trainers during the IVML sessions, different techniques used for initiating interactions or for reducing the need for interactions, and so forth. The output variables define the estimated measures about the IVML sessions that vary depending on the input variables. The output variables may be selected from those cost measures mentioned in Section 2.1 as well as various conventional measures of ML processes (e.g., accuracy, precision, recall, F1-score, etc.).

For example, one may define I for a supervised ML method as a simple three-value nominal variable:

$$\text{Method: } M = \{\texttt{Manual}, \texttt{ActiveLearning}, \texttt{AutomatedML}\}$$

which represents three settings: (i) fully manual labelling 100% data objects without ML; (ii) active learning with dynamic labelling of 20% data objects during ML and applying the learned model to the rest 80% data objects; and (iii) fully automated ML with pre-labelling of 20% data objects and applying the learned model to the rest 80% data objects. Meanwhile, the output O can be as simple as two basic variables:

$$\text{Accuracy: } Acc \in [0,1], \quad \text{Number of Interactions: } \#Int \in \mathbb{N}.$$

With some further assumptions such as the number of interactions for labelling each data object, it is feasible to write a program to compute the function F in the three scenarios, allowing a quantitative comparison of $(Acc_i, \#Int_i)$, $i = 1,2,3$.

It is not difficult to see that the function F can involve more input and output variables, and can simulate many different settings of IVML sessions.

With such a programmable function, one can address a number of aforementioned challenges in direct measurements through empirical studies and record-keeping. For example:

- Simulation takes much less time to run in comparison with an empirical study and keeping records over a project period.
- Simulation can cover scenarios involving a large number of interactions or longer interaction time that would not be feasible to measure in an empirical study that typically lasts between 30-60 minutes.
- Simulation can cover many settings of IVML sessions that cannot all be implemented in a typical model-development project.
- Simulation can collect measures using repeated measures by, e.g., running differently-ordered sequences of training data, alleviating the confounding effect due to the sampling order.

3 A Case Study

In this section, we describe our experience of using simulation to estimate the cost of interactions and to compare different IVML methods with fully manual labelling and fully automated ML.

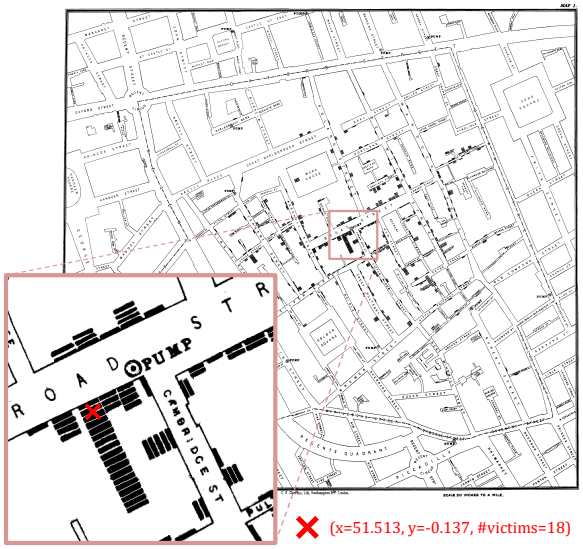

Figure 1: John Snow's Cholera Map [9] contains 321 data points. Each data point is a tuple $(x, y, \#victims)$, representing the latitude and longitude of a location, and the number of victims at this location.

3.1 Data Reconstruction from Historical Visualizations

This case study is concerned with the development of a software system for reconstructing data from historical visualization images. For example, a user may wish to reconstruct the data from John Snow's Cholera Map [9] as shown in Figure 1 by identifying and measuring all data points $(x, y, \#victims)$. One may consider the following four types of systems for supporting the user:

1. No software system support — The user visually identifies all data points and uses the "pen and ruler" approach to measure each $(x, y, \#victims)$.
2. An interactive system — The user would visually identify all data points and use the system interactively to click each point (x, y), and enter its data value $\#victims$.

3. An IVML system — A system adopts an active learning approach, has some built-in image processing utilities, and attempts to learn dynamically to identify all data points and measure $(x, y, \#victims)$. It initiates interactions with the user to seek help, such as labelling imagery components.

4. A conventional ML-based system — The user pre-label a set of training data related to a historical visualization image, and then use a supervised ML method to train an ML model, which is then incorporated into a generic software system for identifying and measuring all data points in the historical visualization image concerned.

We consider that Approach 1 of measuring all 321 data points manually could be rather time-consuming and laborious, while Approach 4 of conventional ML would not be feasible because the training dataset might require more than 321 data objects in order for the ML model to reach a high-level accuracy. We have thus developed an IVML system for supporting data reconstruction.

3.2 Techniques for Reducing Interaction Cost

In IVML, interactions bring about benefit while incurring cost. Hence we have considered a range of techniques for making interactions more cost-effective. This raises the question about how to evaluate and compare different technical solutions. As discussed in Section 2.1, using empirical studies and record-keeping for this purpose would encounter many challenges. We thus adopt the simulation approach, with which we can estimate the cost of interactions routinely whenever a new technical solution arose during the project development.

We adapted the strategy of demanding only simple input actions from the user (such as point and click), and avoiding complex actions (such as drawing a boundary). This allowed us to focus on the "number of interaction" as the core metric in our simulation. Our IVML system has many technical components, and throughout the project, we considered several design options for each component. Hence the combination of different design options for different components resulted in a range of technical solutions for the whole system. We list below some major variations of these solutions that were evaluated and compared through simulation:

- **BF** — This is used to estimate the cost of interactions in the aforementioned Approaches 1 and 2 (i.e., without ML or IVML).
- **ML +** x**%-PL** — The Approach 4 (conventional ML) with x% of data objects pre-labelled before the learning session.
- **IVML + RS** — An IVML solution with active learning and random sampling (RS) of data objects for labelling.
- **IVML + AS** — An IVML solution with active learning and algorithmic sampling (AS) of data objects for labelling. Algorithmic sampling is a technique for selecting more informative data objects to be presented to the user for labelling.
- **IVML + RS + AD +** k**-DL** — An IVML solution that adds an algorithmic default labelling (AD) technique to the **IVML + RS**. The AD technique uses an algorithm (e.g., label propagation or an interim learned model) to guess the possible labels of some unlabelled data objects. With the AD technique, a user interface for active learning can provide the user with default labels (DF) of the presented data objects. If all default labels are correct, the user only needs to confirm all labels with one confirmation interaction. If some default labels are incorrect, the user needs to correct the mistakes before the confirmation. So the number of interactions depends on the error rate of the AD technique as well as the number k of default labels in each iteration of active learning.
- **IVML + AS + AD +** k**-DL** — Similar to the above, except that algorithmic sampling is used in place of random sampling.
- **IVML + RS + RD +** k**-DL** — An IVML solution that adds a random default labelling (RD) method to the **IVML + RS**. By assigning a default label randomly to each data object to be presented to the user, this provides a reference benchmark for evaluating the algorithmic default labelling technique.
- **IVML + AS + RD +** k**-DL** — Similar to the above, except that algorithmic sampling is used in place of random sampling.

It is easily observed that many more alternative solutions can be simulated, e.g., with different x% for pre-labelling, and different k for default labels. During our development, we implemented several versions for various components, such as algorithmic sampling and label propagation. We collected their performance measures (e.g., error rate), and fed them into the simulation software. Furthermore, our IVML system consists of a number of different pipelines, e.g., for detecting blocks, multi-block bars, root positions, lines, curves, polylines, boundaries, etc. Hence the actual number of variations being simulated is much higher. The simulation-based evaluation has been an indispensable method in our development project.

Because the focus of this paper is on evaluation and the page limit of a short paper, it is not possible to describe the details of various technical solutions mentioned here. These have been reported in a separate technique paper [12].

3.3 Cost Estimation and Repeated Measures

Let us first consider the example of block recognition in relation to John Snow's Cholera Map [9]. Figure 2 shows a set of simulation results for comparing six different technical solutions in implementing one of the pipelines, i.e., for recognizing all blocks (i.e., the components of stacked bars) in John Snow's Cholera Map. The six solutions are (i) **BF**, (ii) **ML +** 20%**-PL + NeD + 6-DL**, (iii) **IVML + RS + RD + 6-DL**, (iv) **IVML + AS + RD + 6-DL**, (v) **IVML + RS + AD + 6-DL**, and (vi) **IVML + AS + AD + 6-DL**. The simulation results here are for comparing a specific setting involving a user interface showing six data objects each time for the user to label, a specific algorithm (referred to as ES) for algorithmic sampling, and another specific algorithm (referred to as Interim) for algorithmic default labelling. The tags **IVML** and **6-DL** are omitted in the legend of the plot.

The block recognition pipeline first detects all potential candidates of blocks and then attempts to classify them into blocks and non-blocks. It may terminate the active learning process after a number of iterations, applies the learned model to all unlabelled candidates, and ask the user to confirm the correct answers. Visually, humans are able to recognize 579 rectangular blocks in John Snow's Cholera Map. The approach *BF* would require a minimal of 579 interactions to achieve 100% accuracy.

However, the image processing step of the pipeline cannot recognize the 579 blocks reliably due to the quality of the hand-drawn visualization. In order to ensure that the candidate set contained as many true positives as possible, the image processing step was tuned towards a lower threshold, yielding 4416 candidates in total. Even with such a low threshold, 46 blocks were not among the 4416 candidates. Hence, there was a minimal overhead of 46 interactions for quality assurance if one demands 100% accuracy.

For a conventional ML process with pre-labelling 20% candidates (i.e., **ML +** 20%**-PL**) would require 883 pre-labelling interactions using a basic UI that labels objects one each time. However, with the anticipation of more false positives than true positives, one can design a UI that allows a user to label k data objects in one go. These data objects can all be labelled "non-block" by default (i.e., labelled as NeD (Negative Defaults) in Figure 2). Hence about 12% of the default labels need correction. If the UI is designed to label six data

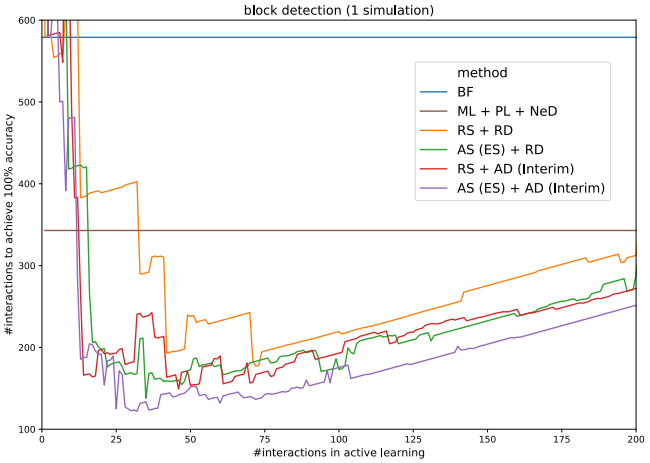

Figure 2: Comparing six technical solutions for detecting blocks in John Snow's Cholera Map. The x-axis indicates the number of interactions have been performed after each iteration of active learning, while the y-axis shows the total number of interactions that would be required to achieve 100% accuracy in block recognition.

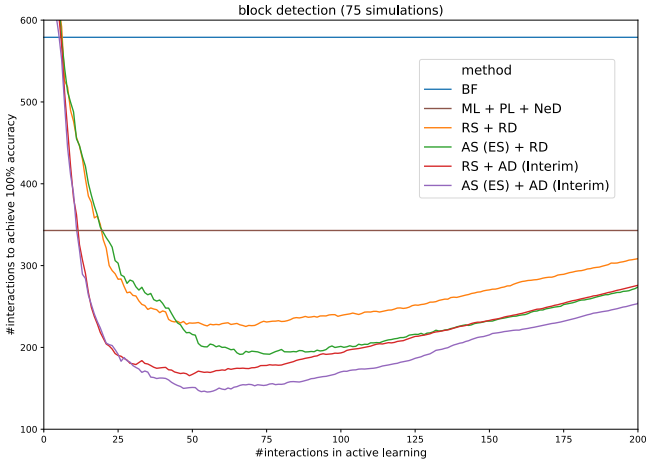

Figure 3: Using 75 repeated measures, the simulation results are must less cluttered than Figure 2.

objects each time, 883 data objects would need 108 interactions for correction and 148 interactions for confirmation. This does not include the 46 interactions for correcting those false negative cases.

As the ML model learned from the 883 training data objects is unlikely to achieve 100% accuracy, further interactions for quality assurance will be necessary (in addition to the aforementioned 46 ones). Our testing of the learned model with 20% pre-labelling (averaging 75 simulations) showed that the model achieved about 98.837% accuracy, with 0.702% false positives and 0.461% false negatives. We can thus estimate that 41 ($\approx (0.702+0.461)\% \times (4416-883)$) interactions would be required for correcting these false positives and false negatives. The total number of interactions for pre-labelling and quality assurance is thus $108+148+41+46=343$.

With the four IVML solutions in Figure 2, the system dynamically selects k data objects in each iteration using random or algorithmic sampling, assigns their labels using random or algorithm default labelling, and asks the user to correct any labelling errors. The system then attempts to update the interim model using the k training data objects. In our simulation, we can apply the interim model to the unlabelled data objects, estimate the number of interactions required for quality assurance should the interim model were to be deployed. In Figure 2, the x-axis indicates the number of interactions performed after each iteration of active learning, while the y-axis shows the total number of interactions that have already been performed for dynamic labelling and for quality assurance in order to achieve 100% accuracy of block recognition.

Since we have implemented these four IVML solutions, we have called the real implementations to obtain the perform measures of each interim model. Because each iteration may add 1-7 interactions, and the labels may occasionally weaken the performance of the interim model, we can observe the step patterns in Figure 2. After some iterations in active learning, further interactions for dynamic labelling may not improve the learned model significantly. We can observe that the total number of interactions increases because the number of interactions required for quality assurance remains more or less unchanged while those for dynamic labelling continue to increase.

With the simulation approach, one is able to consider a variety of independent variables that could affect the cost of interaction as well as the quality of the learned models. However, the more independent variables are involved, the more complex it will be to interpret the simulation results. For example, one may wish to simulate the potential labelling errors in all four approaches mentioned in 3.1. The learned model can be sensitive to which data objects are labelled incorrectly. With the random sampling (**RS**) technique mentioned in Section 3.2, a specific sampling sequence may lead to a learned model that may be more or less accurate as many ML methods are sensitive to the ordering of the training data. The up and down patterns in Figure 2 are caused by such sensitivity.

To alleviate the issues of such sensitivity, which are similar to confounding effects in empirical studies, we can use "repeated measures" that are commonly used in empirical studies for alleviating confounding effects. Figure 3 shows a less cluttered plot, where each of the four IVML curves is the average of 75 repeated measures. Note that in empirical studies, one typically uses 3-5 repeated measures (i.e., 3-5 stimulus for representing the same condition). Most studies cannot afford the time for many repeated measures. In this respect, there is an overwhelming advantage to use simulation.

4 Conclusion

In this short paper, we have articulated the need for measuring the cost of interactions for evaluating IVML solutions, and have reasoned about the merits of using simulation to derive different measures of such cost. We have shown a case study where the number of interactions was used as a metric for evaluating and comparing different IVML solutions, and have discussed the need for repeated measures in order to produce more informative simulation results. The research on measuring the benefit [11] and the cost of interactions in the context of IVML is still in an early stage. We hope to carry out more case studies in the future.

References

[1] E. T. Brown, J. Liu, C. E. Brodley, and R. Chang. Dis-function: Learning Distance Functions Interactively. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology*, pp. 83–92. IEEE, oct 2012. doi: 10.1109/VAST.2012.6400486

[2] M. Chen and A. Golan. What May Visualization Processes Optimize? *IEEE Transactions on Visualization and Computer Graphics*, 22(12):2619–2632, dec 2016. doi: 10.1109/TVCG.2015.2513410

[3] M. El-Assady, R. Sevastjanova, F. Sperrle, D. Keim, and C. Collins. Progressive Learning of Topic Modeling Parameters: A Visual Analytics Framework. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):382–391, jan 2018. doi: 10.1109/TVCG.2017.2745080

[4] H. Kim, J. Choo, H. Park, and A. Endert. InterAxis: Steering Scatterplot Axes via Observation-Level Interaction. *IEEE Transactions on*

Visualization and Computer Graphics, 22(1):131–140, jan 2016. doi: 10.1109/TVCG.2015.2467615

[5] B. C. Kwon, B. Eysenbach, J. Verma, K. Ng, C. De Filippi, W. F. Stewart, and A. Perer. Clustervision: Visual Supervision of Unsupervised Clustering. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):142–151, jan 2018. doi: 10.1109/TVCG.2017.2745085

[6] T. Mitchell. *Machine learning*. McGraw-Hill Education, 1997.

[7] T. Muhlbacher, L. Linhardt, T. Moller, and H. Piringer. TreePOD: Sensitivity-Aware Selection of Pareto-Optimal Decision Trees. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):174–183, jan 2018. doi: 10.1109/TVCG.2017.2745158

[8] N. Pezzotti, T. Hollt, J. Van Gemert, B. P. Lelieveldt, E. Eisemann, and A. Vilanova. DeepEyes: Progressive Visual Analytics for Designing Deep Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):98–108, jan 2018. doi: 10.1109/TVCG.2017.2744358

[9] J. Snow. *On the Mode of Communication of Cholera*. John Churchill, 1855.

[10] G. K. L. Tam, H. Fang, A. J. Aubrey, P. W. Grant, P. L. Rosin, D. Marshall, and M. Chen. Visualization of Time-Series Data in Parameter Space for Understanding Facial Dynamics. *Computer Graphics Forum*, 30(3):901–910, jun 2011. doi: 10.1111/j.1467-8659.2011.01939.x

[11] G. K. L. Tam, V. Kothari, and M. Chen. An Analysis of Machine- and Human-Analytics in Classification. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):71–80, jan 2017. doi: 10.1109/TVCG.2016.2598829

[12] Y. Zhang, M. Chen, and B. Coecke. MI3: Machine-Initiated Intelligent Interaction for Dynamic Data Annotation and Data Reconstruction. 2019. under review.