

HypoML: Visual Analysis for Hypothesis-based Evaluation of Machine Learning Models

Qianwen Wang¹, William Alexander², Jack Pegg², Huamin Qu¹, and Min Chen² 

¹ Hong Kong University of Science and Technology, Hong Kong, China and ² University of Oxford, UK

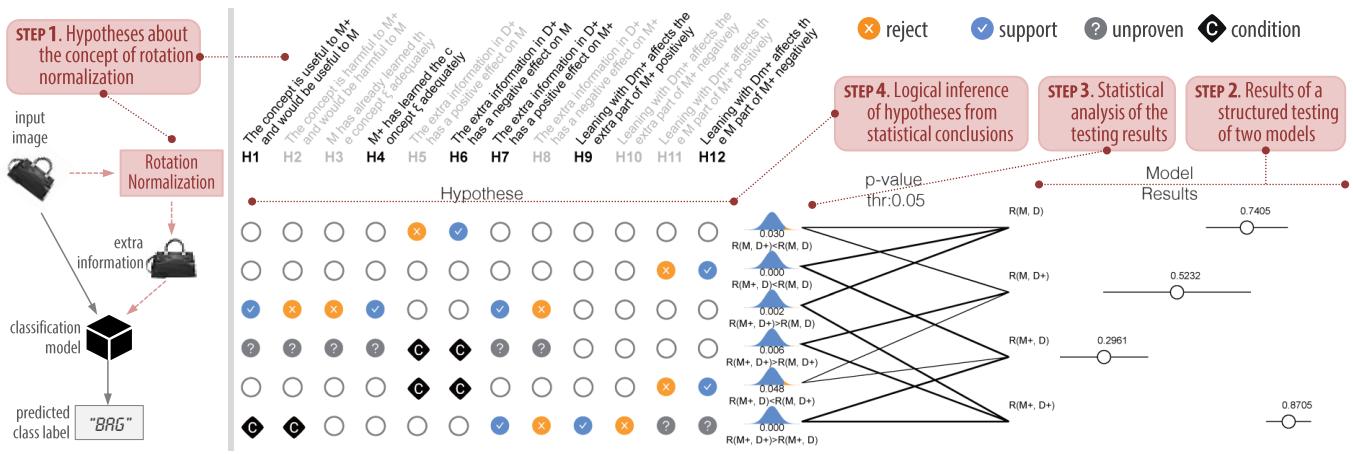


Figure 1: One may wish to know if a classification model is rotation-invariant. If the model is not so, one may use another model that can detect the rotation angle of an object or perform some rotation normalization. The detected rotation angle or normalized image is a piece of extra information about a “concept” that the original model may or may not know. With HypoML, one can conduct a set of structured tests, obtained automated statistical and logical analysis of the results, and visualize the conclusions about the hypothesized concepts.

Abstract

In this paper, we present a visual analytics tool for enabling hypothesis-based evaluation of machine learning (ML) models. We describe a novel ML-testing framework that combines the traditional statistical hypothesis testing (commonly used in empirical research) with logical reasoning about the conclusions of multiple hypotheses. The framework defines a controlled configuration for testing a number of hypotheses as to whether and how some extra information about a “concept” or “feature” may benefit or hinder a ML model. Because reasoning multiple hypotheses is not always straightforward, we provide HypoML as a visual analysis tool, with which, the multi-thread testing data is transformed to a visual representation for rapid observation of the conclusions and the logical flow between the testing data and hypotheses. We have applied HypoML to a number of hypothesized concepts, demonstrating the intuitive and explainable nature of the visual analysis.

1. Introduction

In computer vision, data mining, and machine learning (ML), a *feature* is a measurable variable that characterizes a particular kind of property or attribute of a data object (e.g., an image, a time series, a multivariate record, etc.). Many technical solutions in these fields heavily rely on model-developers’ knowledge about various features and include human-centric feature engineering as a critical process in a model development workflow [EI96, Alp10].

On the other hand, some other technical solutions were designed

to minimize the dependence on the human knowledge of potentially useful features. For example, in deep learning, neural networks are typically expected to learn how to extract a good number of useful features automatically [LBH15]. At the same time, there have also been concerns that some so-called “useful” features may be actually harmful because they contribute towards undesirable biases [ZWW18, KRCP*17]. Inevitably, model-developers have been interested in what features may have or have not been learned by an ML model. A class of visualization techniques, such as neuron

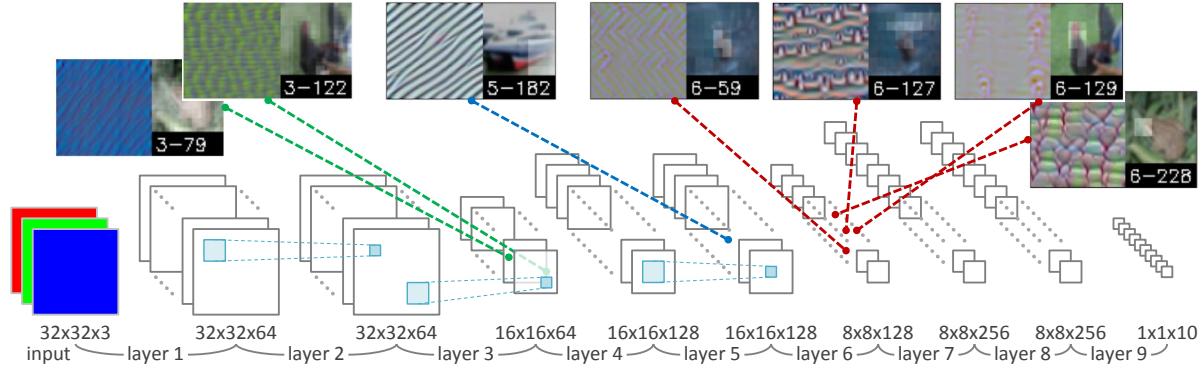


Figure 2: Gradient ascent [SDBR14] can help model-developers observe the pattern that a specified neuron has learned. However, even a small CNN has a huge number of neurons waiting to be inspected, while many patterns shown are not semantically interpretable. Meanwhile, model-developers are often unable to determine whether a pattern is useful or harmful.

activation plot, filter plot, gradient ascent plot [SDBR14], Deconvolution [ZF14], and their variants, have been widely used by developers of neural networks to observe neurons. Since a neural network typically consists of a huge number of neurons, the visual observation may encounter several obstacles, including time demand for viewing all neurons that may reveal some features, subjectivity and memory limitation of an observer, and uncertainty about the semantic meaning of an observed feature. More importantly, while most model-developers have a non-trivial amount of knowledge about features that are potentially useful or harmful, their initiatives are limited to searching for patterns in many thousands of neuron-based plots and speculating if a feature has been learned.

In this work, we propose a new visual analytics approach that enables model-developers to use their knowledge and initiatives in hypothesising and evaluating if any feature may be useful or harmful, if such a feature is learned by a model, and how it may affect a learned model. In particular, we outline a framework for testing such hypotheses systematically, and describe the underlying statistical and logical analysis for inferring conclusions about multiple hypotheses from multiple sets of testing results. Because many model-developers may not be familiar with or remember the underlying statistical and logical analysis, we develop a visual analytics tool, HypoML, for carrying out analysis as well as for depicting the flow of inference (Figure 1), facilitating rapid observation of the conclusions and the logical flow between the testing data and hypotheses. We have made HypoML available as open-source software, a demo is available at <https://hypoml.bitbucket.io/> and the source code is available at <https://bitbucket.org/hypoML/hypoml.bitbucket.io>.

The term “feature” typically implies a piece of information contained in the original input data. Since HypoML can also be used to test a hypothesis about a piece of information that may not be part of the original data, we will use the term “concept-based hypotheses” to describe what to be tested with HypoML.

2. Related Work

While machine learning (ML) has an important role to play in visualization and visual analytics [ERT*17], almost every aspect of

ML processes can benefit from visualization as shown by a recently established ontology VIS4ML [SKKC19]. In general, when model-developers observe some phenomena in an ML process, such as its training and testing data, results, the inner states of a model, and the provenance of the learning process, they acquire new information to inform their various decisions that affect the ML process. As demonstrated quantitatively by Tam et al. [TKC01], a model-developer can contribute a huge amount of knowledge (measured in bits) to an ML process through the use of visualization. This work focuses on the evaluation stage of ML workflows.

Methods for evaluating ML models can be categorized into two main classes: black-box analysis and white-box analysis. Here we focus our review of the previous works on model evaluation that feature visualization techniques. More comprehensive surveys on using visualization for ML can be found in the works of Zhang and Zhu [ZZ18] and Hohman et al. [HPRC20].

Black-box analysis enables users to investigate and evaluate ML models without knowing the internal working mechanism. Statistic metrics (e.g., accuracy, recall), ROC curve, and confusion matrices are widely-used black-box analysis and have commonly been provided as built-in functions in machine learning environments. To aid the aggregated statistical analysis, researchers recently proposed visualization techniques to support black-box evaluation of ML models [ACD*15, KPN16, RAL*17, ZWM*19]. For example, Squares [RAL*17] juxtaposes a set of histograms to present an instance-level visualization for models in multi-class classification problems. Manifold [ZWM*19] employs a scatterplot-based visual technique to assist in the comparison between multiclass classifiers. However, these techniques focus mainly on visualizing model performances and offer limited support for model-developers to ask in-depth questions about the model or the experiment results, or to evaluate specific hypotheses in a statistically-meaningful way.

White-box analysis, on the contrary, opens the black box and displays the internal states of ML models. A number of visualization tools have been proposed to support white-box analysis of different ML models, including MLP [RFFT17], CNNs [LSL*17, KAKC18, PHVG*18, RFFT17, LCJ*18], deep generative models [LSC*18, WGYS18, KTC*19], and RNNs [MCZ*17, SGPR18]. Although these tools have utilized some of the most sophisticated

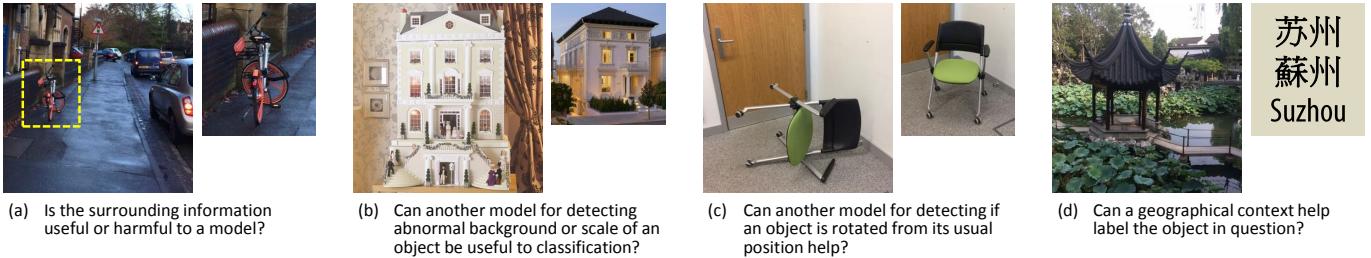


Figure 3: During the development of ML models, model-developers usually have many hypotheses about whether certain extra information (e.g., location, date, time, etc.) or certain preprocessing methods (e.g., cropping, histogram normalization, etc.) would help a model.

visual representations and have assisted model-developers in evaluating, understanding, and explaining their models, comprehending a huge number of high-dimensional internal states is naturally challenging for humans.

In addition, researchers proposed techniques to summarize information about internal variables and present the summary information visually. Salience-based methods, such as CAM [ZKL^{*}16], Grad-CAM [SCD^{*}17], and guided back propagation [SDBR14], identify discriminative regions in the input image and thus highlight important features for a certain prediction. However, these salience-based methods can only offer explanations for specific predictions but cannot confirm whether or not a concept has been learned. To offer instance-independent explanation, Yosinski et al. employed gradient ascent plots [YCF15] to depict the patterns that an individual neuron has learned. Figure 2 illustrates a small selection of gradient ascent plots being observed in conjunction with a CNN. However, even for such a simple model, there are a huge number of neurons, it is impossible for model-developers to conduct a full examination. Moreover, the depicted pattern would largely be a hunch, but not a proof that a certain concept is useful or not to the classification task. Perhaps the most relevant to our work is TCVA [KWG^{*}18], which learns human-friendly concepts from an already trained model and conducts hypothesis testing. However, TCVA requires a time-consuming process to label the concept across the whole dataset.

In this work, we propose a novel ML-testing framework that combines black-box and white-box analysis. Whether an ML model has learned a concept or feature is a typical “internal problem” that is to be investigated using white-box analysis. The new framework allows model-developers to investigate “internal problems” in a manner of black-box analysis.

3. Concept-Based Testing of ML Models

Let M be a machine learned (ML) model that transforms an input data object $d \in D$ to an output decision that may be of a classification label, or a prediction. A *concept* ξ is a variable that is not explicitly defined in d_i , but is hypothesized by an ML model-developer that ξ would be useful or harmful to the quality of the output decision should M be able to access some extra information about ξ . Figure 3 shows several examples of concepts. We can observe that some concepts may be extracted from the original data

objects using known techniques, while it may be almost impossible to infer some other concepts from the data objects.

As long as M has a finite number of constructs (e.g., neurons or tree nodes) or receives input data with finite informative dimensions, there will always be some concepts that M cannot learn. Inevitably, most model-developers will have questions about some concepts in relation to a learned model M . For example, considering the examples in Figure 3, one may ask:

- Would having an extended field of view be useful for recognizing an object captured from a less ideal viewing angle?
- Would another model for detecting an anomalous background or some scale inconsistency be useful to differentiate a toy from a real building?
- Would another model that is able to detect an object in an unusual position and estimate the rotation angle be useful to the recognition of the object?
- Would having additional information about a geographical context improve the accuracy of building recognition?

One can easily imagine many other questions about different types of extra information, such as different meta-data, multiple data capture modalities, and various pre-processing techniques. All these questions are essentially hypotheses. Just as in psychology, healthcare, social science, and many other disciplines, one can conduct experiments to evaluate such hypotheses. Indeed, one can test ML models against many thousands of data objects in comparison with tens of stimuli in typical empirical studies.

Because model testing is a routine operation in ML, it is desirable to establish a structured method such that many model-developers can adopt the same method and produce comparable testing results. Because the above definition of *concept* is relatively broad, developers of different ML models in various applications can benefit from open source software or commercial systems for supporting such a structured testing method.

Figure 4 illustrates the framework for concept-based hypothesis testing. Given an ML process and a training and testing dataset, a model-developer is interested to know how some extra information about a concept may affect the ML process and the learned model. The framework thus requires the developer to invoke two ML processes that receive two pieces of input data. As shown on the left of Figure 4, both processes take the original training data D_m as one piece of the data. For the other piece, one process takes random

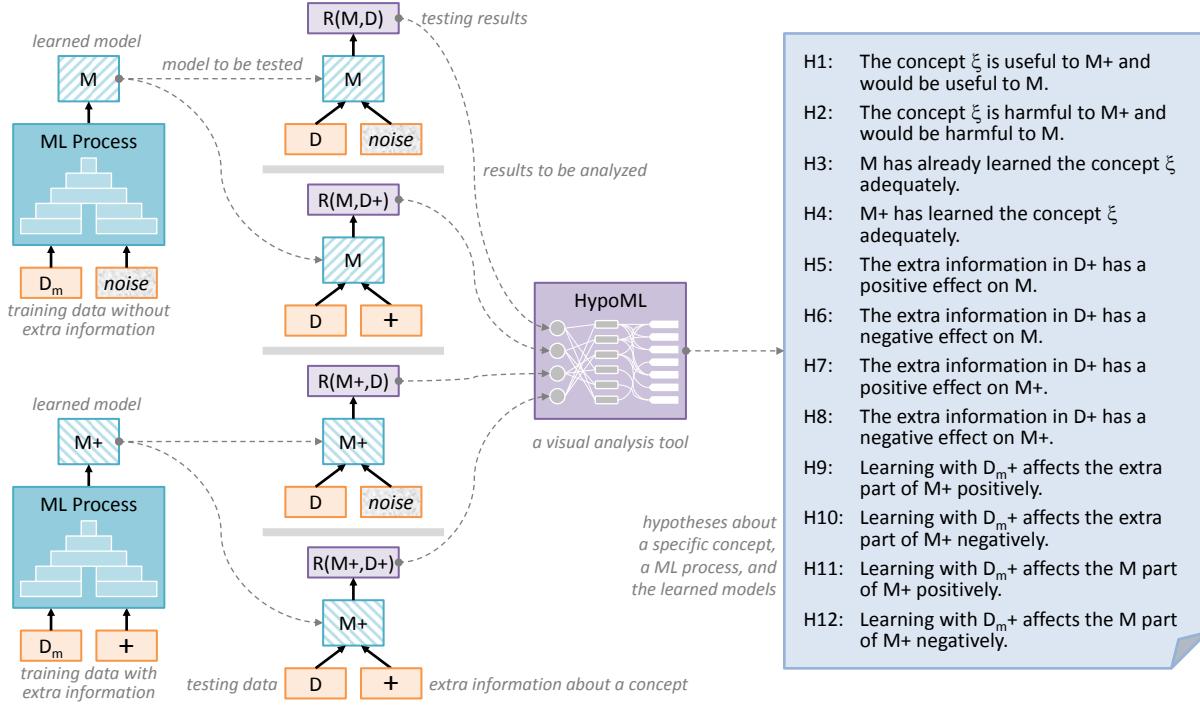


Figure 4: An illustration of the structured testing method proposed in this work. The concept to be tested is encoded as extra information to accompany the original data. Two models, M+ and M, are trained with and without extra information. Both models are then tested using two different types of testing data, one with extra information and one without. The four sets of results are then analyzed by the HypoML tool against 12 hypotheses. HypoML presents the analytical conclusions using visualization as shown in Figure 1.

noise as its input, while the other takes extra information about a concept (denoted by the sign “+”).

Following the same procedure for model training, the two ML processes generate two learned models, M and M+, respectively. The framework then requires the model-developer to test each model with two runs. As illustrated in the middle column of Figure 4, one testing run uses testing data D that does not have extra information, while the other run uses testing data D+ that include extra information. The two runs with M thus produce two sets of results, R_{M,D} and R_{M,D+}, while the two runs with M+ produce R_{M+,D} and R_{M+,D+}. Because evaluating an ML model typically involves testing many thousands of data objects, some computational analysis of the four sets of results will be necessary.

HypoML is designed to support the computational analysis. In particular, it provides statistical and logical analysis for evaluating a set of hypotheses. The statistical analysis is based on the well-established method for hypothesis testing, while the logical analysis is formulated in this work for reasoning about the intertwining relationships between 12 hypotheses and 6 statistical conclusions drawn from different pairs of results. To assist users in understanding such complex relationships, HypoML provides a purposely-designed visual representation, which enables users to trace the conclusion of each hypothesis to related statistical analysis and to the corresponding testing results.

The 12 hypotheses are listed on the right of Figure 4. The first two hypotheses, H₁ and H₂, are about whether the concept con-

cerned is useful (or harmful) to M+, and would be useful (or harmful) to M. Although the conclusions for these two hypotheses cannot in principle be both positive, each can also be inconclusive. We thus follow the convention of hypothesis testing by listing them as separate hypotheses, each can be independently confirmed, rejected, or unproven (inconclusive).

H₃ hypothesizes that model M has already learned the concept adequately, while H₄ hypothesizes that model M+ has learned the concept adequately. For H₃, the adverb “adequately” implies that the concept can be learned by a model, such as M, without the need for any extra information about the concept. For H₄, the adverb “adequately” implies that M+ would perform worse without the extra information of the concept.

In general, model M has not been trained with extra information. It is thus not expected to be affected by any extra information during testing. However, as a scientific exercise, one cannot take this assumption for granted since one cannot assume that a model template (e.g., an untrained neural network) has always been configured correctly or a training method has always been implemented correctly. H₅ and H₆ are thus designed to examine whether M is affected positively or negatively by the extra information during testing. Because there exists an inconclusive state, they are kept as two separate hypotheses, in a way similar to H₁ and H₂.

When model M+ is trained with extra information, the model may learn new capability from the extra information, while losing some capability that would be learned without the extra informa-

tion. The vice versa could also be true. \mathbf{H}_9 , \mathbf{H}_{10} , \mathbf{H}_{11} , and \mathbf{H}_{12} are for investigating the trade-off between different parts of $M+$ in the development of its intelligence. Depending on the design of the model template or architecture, the parts of $M+$ for handling the extra information (+ part) and the original information (D) can be quite separated as well as rather integrated. When the two parts are more integrated, one should consider two parts as functional units rather than geometric or topological regions. Similarly, we separate \mathbf{H}_9 from \mathbf{H}_{10} , and separate \mathbf{H}_{11} from \mathbf{H}_{12} because of the inconclusive state in each case. We also anticipate that more testing and analysis methods may be developed in the future, which may support or reject those apparently-paired hypotheses asymmetrically. Having separate hypotheses will not hinder such advancement.

4. Statistical and Logical Reasoning of Hypotheses

As shown in Figure 4, HypoML receives four sets of results, namely $R_{M,D}$, $R_{M,D+}$, $R_{M+,D}$, and $R_{M+,D+}$. Each set of results is a list of tuples, each of which consists of:

- **id** — the unique identifier of a data object. The data object may be an image, a feature vector, a multivariate data record, or a more complex data record.
- **ground truth** — a ground truth label, which can be a nominal value, an integer, a real number, a range, or a data record of a more complex data type (e.g., a time series).
- **ML label** — a label generated by an ML model. The label must be of the same data type as ground truth.
- **ML uncertainty** — an optional value indicating the uncertainty estimated by an ML model equipped with a self-assessment capacity. It is a real number in the range $[0, 1]$ with 1 being the most uncertain. Many ML models may not have any self-assessment capacity, and in such a case, this entry takes the default value 0. Some ML models may return a confidence value, which can easily be converted to uncertainty.
- **correctness** — This is a value in the range of $[0, 1]$ with 1 indicating absolutely correct, and 0 indicating absolutely incorrect. The value is mostly computed based on ground truth and ML label using a user-defined function. The simplest function can be true (1) if ground truth equals ML label, or 0 otherwise. A more complicated function may feature a distance or similarity metric.
- **correctness with uncertainty** — This is used by the statistical analysis and is defined as $ML\ uncertainty \times correctness$.

Given two sets of results, R_a and R_b , we assume that the tuples in the two lists are paired, i.e., the id entries are in the same order exactly. We can compare R_a and R_b with their accuracy, i.e., the average of correctness with uncertainty. As testing in ML often shows small variations of accuracy, it is necessary to measure the statistical significance. HypoML uses paired, two-tail t -test for this purpose. Let us introduce the following notation to denote the possible outcomes of the statistical analysis.

- $R_a \lessapprox R_b$ — It is statistically significant that R_a is lower than R_b .
- $R_a \gtrapprox R_b$ — It is statistically significant that R_a is higher than R_b .
- $R_a \approx R_b$ — It is statistically insignificant that R_a is higher or lower than R_b .
- $R_a \lesssim R_b$ — $R_a \lessapprox R_b$ or $R_a \approx R_b$, but not $R_a \gtrapprox R_b$.
- $R_a \gtrsim R_b$ — $R_a \gtrapprox R_b$ or $R_a \approx R_b$, but not $R_a \lessapprox R_b$.

Analysis	Condition	Hypothesis
$A_1: R_{M+,D+} v. R_{M,D}$		$\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3, \mathbf{H}_4, \mathbf{H}_7, \mathbf{H}_8$
$A_2: R_{M+,D+} v. R_{M,D+}$	$\mathbf{H}_5, \mathbf{H}_6$	$\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3, \mathbf{H}_4, \mathbf{H}_7, \mathbf{H}_8$
$A_3: R_{M+,D+} v. R_{M+,D}$	$\mathbf{H}_1, \mathbf{H}_2$	$\mathbf{H}_7, \mathbf{H}_8, \mathbf{H}_9, \mathbf{H}_{10}, \mathbf{H}_{11}, \mathbf{H}_{12}$
$A_4: R_{M+,D} v. R_{M,D}$		$\mathbf{H}_{11}, \mathbf{H}_{12}$
$A_5: R_{M+,D} v. R_{M,D+}$	$\mathbf{H}_5, \mathbf{H}_6$	$\mathbf{H}_{11}, \mathbf{H}_{12}$
$A_6: R_{M,D+} v. R_{M,D}$		$\mathbf{H}_5, \mathbf{H}_6$

Table 1: The relations between statistical analysis and hypotheses.

With four sets of results, there are six pairs of statistical comparison, which are labelled as A_1, A_2, \dots, A_6 . Each analytical conclusion A_i may support or reject some of the 12 hypotheses $\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_{12}$, but not all. For example the analysis A_1 , which compares $R_{M+,D+}$ and $R_{M,D}$, can inform the evaluation of \mathbf{H}_1 and \mathbf{H}_2 . If $R_{M+,D+}$ is statistically better than $R_{M,D}$, i.e., $R_{M+,D+} \gtrapprox R_{M,D}$, we can draw a conclusion that A_1 supports \mathbf{H}_1 and rejects \mathbf{H}_2 . If $R_{M+,D+} \lessapprox R_{M,D}$, A_1 supports \mathbf{H}_2 and rejects \mathbf{H}_1 . If $R_{M+,D+} \approx R_{M,D}$, A_1 returns an unproven (inconclusive) verdict about \mathbf{H}_1 and \mathbf{H}_2 .

With some careful reasoning, we can observe that A_1 can also inform the evaluation of $\mathbf{H}_3, \mathbf{H}_4, \mathbf{H}_7$, and \mathbf{H}_8 . While A_2 can inform the evaluation of $\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3, \mathbf{H}_4, \mathbf{H}_7$, and \mathbf{H}_8 , but it can only do so subject to that some other hypotheses have already been confirmed or rejected. Table 1 summaries the relations between the six sets of statistical analysis A_1, A_2, \dots, A_6 and the 12 hypotheses $\mathbf{H}_1, \mathbf{H}_2, \dots, \mathbf{H}_{12}$.

Clearly, reasoning about these relations is time consuming and error prone. In order to support the frequent analytical tasks of the developers in testing their ML models, HypoML provides automated logical analysis as well as statistical analysis. To help describe the logical analysis, we employ some additional notations. They are:

- $\top(S)$ — The statement S is true.
- $\perp(S)$ — The statement S is false.
- $\ast(S)$ — The statement S is unproven.
- \wedge — Logical conjunction.
- \vee — Logical (inclusive) disjunction.

We can now specify the logical inference from A_1 as:

$A_1: R_{M+,D+} v. R_{M,D}$ may conclude:

- $R_{M+,D+} \gtrapprox R_{M,D} \implies \top(\mathbf{H}_1) \wedge \top(\mathbf{H}_4) \wedge \top(\mathbf{H}_7) \wedge \perp(\mathbf{H}_2) \wedge \perp(\mathbf{H}_3) \wedge \perp(\mathbf{H}_8)$. This reads as $\mathbf{H}_1, \mathbf{H}_4$, and \mathbf{H}_7 are all true, and $\mathbf{H}_2, \mathbf{H}_3$, and \mathbf{H}_8 are all false.
- $R_{M+,D+} \lessapprox R_{M,D} \implies \top(\mathbf{H}_2) \wedge \top(\mathbf{H}_4) \wedge \perp(\mathbf{H}_1) \wedge \perp(\mathbf{H}_3)$.

Analysis A_2 cannot draw conclusions about \mathbf{H}_5 and \mathbf{H}_6 , but its conclusion may depend on them. In general, there is a common-sense assumption that neither \mathbf{H}_5 nor \mathbf{H}_6 is likely to be true.

$A_2: R_{M+,D+}$ vs. $R_{M,D+}$ may conclude:

- $R_{M+,D+} \gtrapprox R_{M,D+} \implies$
 - (i) **if** $\perp(\mathbf{H}_6)$ **then** $\top(\mathbf{H}_1) \wedge \top(\mathbf{H}_4) \wedge \top(\mathbf{H}_7) \wedge \perp(\mathbf{H}_2) \wedge \perp(\mathbf{H}_3) \wedge \perp(\mathbf{H}_8)$; **or**
 - (ii) **if** $\ast(\mathbf{H}_6)$ **then** $\top(\mathbf{H}_1) \wedge \top(\mathbf{H}_4) \wedge \top(\mathbf{H}_7) \wedge \perp(\mathbf{H}_2) \wedge \perp(\mathbf{H}_3) \wedge \perp(\mathbf{H}_8)$; **or**
 - (iii) **if** $\top(\mathbf{H}_6)$. This offers an explanation but it is against a

common-sense assumption that \mathbf{H}_6 is unlikely to be true, and should be treated cautiously.

- $R_{M+,D+} \not\approx R_{M,D+} \implies$
 - (i) **if** $\perp(\mathbf{H}_5)$ **then** $\top(\mathbf{H}_2) \wedge \top(\mathbf{H}_4) \wedge \top(\mathbf{H}_8) \wedge \perp(\mathbf{H}_1) \wedge \perp(\mathbf{H}_3) \wedge \perp(\mathbf{H}_7)$; or
 - (ii) **if** $\ast(\mathbf{H}_5)$ **then** $\top(\mathbf{H}_2) \wedge \top(\mathbf{H}_4) \wedge \top(\mathbf{H}_8) \wedge \perp(\mathbf{H}_1) \wedge \perp(\mathbf{H}_3) \wedge \perp(\mathbf{H}_7)$; or
 - (iii) **if** $\top(\mathbf{H}_5)$. This offers an explanation but it is against a common-sense assumption that \mathbf{H}_5 is unlikely to be true, and should be treated cautiously.

Because analysis A_3 does not compare $M+$ with M , the conclusion is limited to the context of $M+$. Mathematically, it is possible for A_3 to conclude that the concept is useful in the context of $M+$, while A_1 or A_2 concludes that the concept is harmful or is neither useful nor harmful. Considering this limitation, it is unsafe for this analysis to draw a conclusion about \mathbf{H}_1 and \mathbf{H}_2 . Meanwhile the analysis depends on the conclusions of \mathbf{H}_1 and \mathbf{H}_2 in a small way.

$A_3: R_{M+,D+}$ v. $R_{M+,D}$ may conclude:

- $R_{M+,D+} \not\approx R_{M+,D} \implies$
 - (i) **if** $\top(\mathbf{H}_1)$, **then** $\top(\mathbf{H}_7) \wedge \top(\mathbf{H}_9) \wedge \perp(\mathbf{H}_8) \wedge \perp(\mathbf{H}_{10})$; or
 - (ii) **if** $\ast(\mathbf{H}_1)$, **then** $\top(\mathbf{H}_9) \wedge \perp(\mathbf{H}_{10})$; or
 - (iii) **if** $\perp(\mathbf{H}_1)$, **then** $\top(\mathbf{H}_{12}) \wedge \perp(\mathbf{H}_{11})$.
- $R_{M+,D+} \not\approx R_{M,D} \implies$
 - (i) **if** $\top(\mathbf{H}_2)$, **then** $\top(\mathbf{H}_8) \wedge \top(\mathbf{H}_{10}) \wedge \perp(\mathbf{H}_7) \wedge \perp(\mathbf{H}_9)$; or
 - (ii) **if** $\ast(\mathbf{H}_2)$, **then** $\top(\mathbf{H}_{10}) \wedge \perp(\mathbf{H}_9)$; or
 - (iii) **if** $\perp(\mathbf{H}_2)$, **then** $\top(\mathbf{H}_{10}) \wedge \perp(\mathbf{H}_9)$. This conclusion is against a common-sense assumption that a useful concept normally should not affect the extra part of $M+$ negatively, and should be treated cautiously.

Analysis A_4 is relatively easy to reason, and it is useful for investigating if the part of model $M+$ for handling the original data D becomes less capable due to the training with extra information.

$A_4: R_{M+,D}$ v. $R_{M,D}$ may conclude:

- $R_{M+,D} \not\approx R_{M,D} \implies \top(\mathbf{H}_{11}) \wedge \perp(\mathbf{H}_{12})$.
- $R_{M+,D} \not\approx R_{M,D} \implies \top(\mathbf{H}_{12}) \wedge \perp(\mathbf{H}_{11})$.

A_5 cannot draw conclusions about \mathbf{H}_5 and \mathbf{H}_6 , but its conclusion may depend on them. In general, there is a common-sense assumption that neither \mathbf{H}_5 nor \mathbf{H}_6 is true.

$A_5: R_{M+,D+}$ v. $R_{M,D+}$ may conclude:

- $R_{M+,D} \not\approx R_{M,D+} \implies$
 - (i) **if** $\perp(\mathbf{H}_6)$ **then** $\top(\mathbf{H}_{11}) \wedge \perp(\mathbf{H}_{12})$; or
 - (ii) **if** $\ast(\mathbf{H}_6)$ **then** $\top(\mathbf{H}_{11}) \wedge \perp(\mathbf{H}_{12})$; or
 - (iii) **if** $\top(\mathbf{H}_6)$. This offers an explanation but it is against a common-sense assumption that \mathbf{H}_6 is unlikely to be true, and should be treated cautiously.
- $R_{M+,D} \not\approx R_{M,D+} \implies$
 - (i) **if** $\perp(\mathbf{H}_5)$ **then** $\top(\mathbf{H}_{12}) \wedge \perp(\mathbf{H}_{11})$; or
 - (ii) **if** $\ast(\mathbf{H}_5)$ **then** $\top(\mathbf{H}_{12}) \wedge \perp(\mathbf{H}_{11})$; or
 - (iii) **if** $\top(\mathbf{H}_5)$. This offers an explanation but it is against a common-sense assumption that \mathbf{H}_5 is unlikely to be true, and should be treated cautiously.

Analysis A_6 is the only comparison that may inform the evaluation of \mathbf{H}_5 nor \mathbf{H}_6 . In general, there is a common-sense assumption

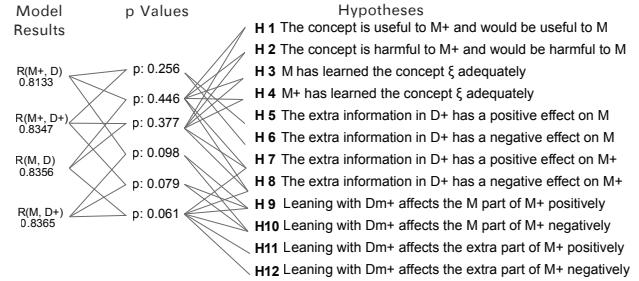


Figure 5: The analytical workflow from testing results to statistical analysis and then logical inference of hypothesis. As a basic visual design, it has a number of shortcomings.

that neither \mathbf{H}_5 nor \mathbf{H}_6 is true if the model template or architecture was correctly defined, the correct ML method was followed, and the correct ML process was executed. When \mathbf{H}_5 or \mathbf{H}_6 is confirmed, it usually suggests some imperfection of the model template or learning process. Therefore the conclusions of A_6 should not be interpreted as their face values. However, the evaluation of \mathbf{H}_5 nor \mathbf{H}_6 is necessary since A_2 and A_5 depend on them.

$A_6: R_{M,D+}$ vs. $R_{M,D}$ may conclude:

- $R_{M,D+} \not\approx R_{M,D} \implies \top(\mathbf{H}_5) \wedge \perp(\mathbf{H}_6)$;
- $R_{M,D+} \not\approx R_{M,D} \implies \top(\mathbf{H}_6) \wedge \perp(\mathbf{H}_5)$.

Because the dependency among the six sets of analysis, the computation of the logical inference must follow an appropriate order, which is summarized as follows:

STEP 0: Initialise the indicator of each hypothesis to 0.

STEP 1: Compute the six comparative values, i.e., A_1, A_2, \dots, A_6 , in terms of $\not\approx$, $\not\gtrless$, and \approx , based on statistical analysis.

STEP 2: Compute the logical inference (i.e., in terms of \top, \perp, \ast) based on A_1, A_4, A_6 . For each true statement, i.e., $\top(\mathbf{H}_i)$, add +1 to the indicator of \mathbf{H}_i . For each false statement, i.e., $\perp(\mathbf{S})$, add -1 to the indicator of \mathbf{H}_i .

STEP 3: Compute the indicators based on A_2, A_5 .

STEP 4: Compute the indicators based on A_3 .

STEP 5: Then display each indicator based on positive or negative values. HypoML displays each hypothesis according to its indicator in three states: >0 (confirmed), 0 (unproven), <0 (rejected).

5. Visual Analysis of Hypotheses

Figure 5 shows a typical workflow of the proposed hypotheses testing. To start with, model-developers conduct experiments and obtain four sets of results, i.e., $R_{M,D}$, $R_{M,D+}$, $R_{M+,D}$, and $R_{M+,D+}$. HypoML then performs six sets of statistical analysis by comparing each pair of the results. Based on the statistical analysis, HypoML makes logical inference about the twelve hypotheses, deciding whether a hypothesis should be supported or rejected.

It is helpful for model-developers to make quick observation about the analysis and conclusions. It will also be useful for the

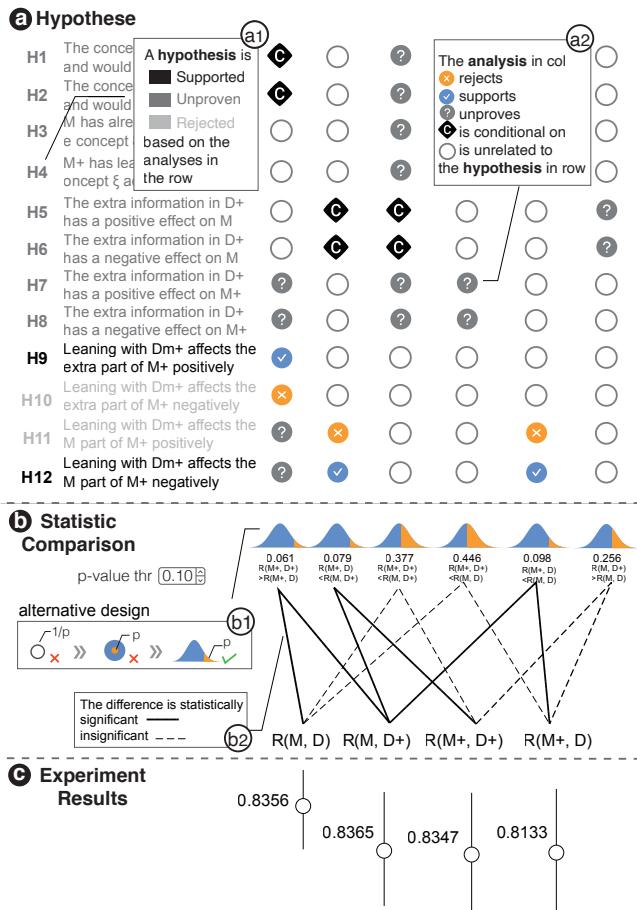


Figure 6: The vertical version of the HypoML interface. In comparison with the basic design in Figure 5, it is much easier for a user to have an overview of the analytical flow, while acquiring quickly the conclusions of different hypotheses. A horizontal version, which is more suitable for wide-screen monitors, is shown in Figure 1.

model-developers to convey the outcomes of the test to other stakeholders, such as users of the ML models being evaluated. It can be difficult for some model-developers and many of ML users to remember and reason the complicated relationships among experiment results, statistic and logical analysis, and multiple hypotheses. Therefore, an effective visual representation is necessary. The bipartite graph shown in Figure 5 is a straightforward solution but it exhibits several shortcomings that hinder efficient information acquisition and effective information dissemination.

One main shortcoming is the cluttered links between the six statistical comparisons and the twelve hypotheses. These links have no obvious or memorable structures and are difficult to track by eye. One can add additional visual encoding to these links to depict three types of conclusions (i.e., reject, support, unproven) and conditional dependency. However, such encoding would further worsen the cluttering of the bipartite graph. To address this issue, we designed a matrix-based visualisation for HypoML as shown in Fig-

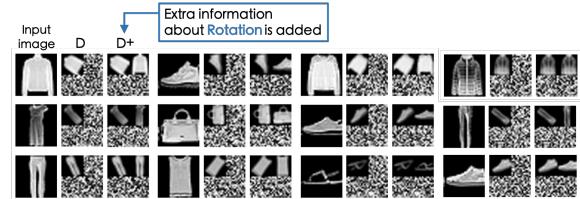


Figure 7: Samples of the training data for testing the concept of rotation correction. For each sample, the left image shows the original object. The middle image shows the corresponding stimulus in the testing dataset D, where the object has been arbitrarily rotated. The right image shows the stimulus in the testing dataset D+ where the rotated object is accompanied by an up-right view of the object.

ure 6(a), where four types of icons (a2) are introduced to indicate reject, support, unproven, and conditional dependency.

The second shortcoming is that simply listing numerical values (e.g., the accuracy of experiment results, the p -value of statistical comparisons) incurs a fair amount of cognitive load upon users who have to compare and analyse them numerically. Therefore, we thus visually encoded these values while maintaining the numerical representations. In particular, HypoML depicts experiment results with positions, since position is considered to be the most effective visual channel [Mun14]. As shown in Figure 6(c), the position of the circle indicates the average accuracy while the line indicates the 95% confidence interval.

We decided to encode p -value using a glyph, and considered several alternative designs as shown in Figure 6(b1). With the first design option, the area of a circle is used to encode the level of statistical significance, i.e., the inverse of a p -value. The less the p -value, the more significant the difference, and the larger the circle. However, in an informal pilot study, this design was found to be “confusing” due to the reverse encoding. With the second design option, the p -value is encoded using the area of an orange circle, which is inside a large blue circle of a fixed size. While this design enables direct observation of statistical significant through the blue area as well as the p -value through the orange area, it was found to be “unintuitive” for those who were unfamiliar with the definition of p -value. We finally settled down on the third design based on a widely-used illustration for explaining the concept of statistical hypothesis testing. In this design, the whole shape represents a normal distribution and the area in orange coarsely encodes the p -value. The normal distribution curve can quickly remind users of the meaning of p -value.

The third shortcoming is that while depicting the reasoning flow from data to conclusion as in Figure 5 correctly represents the temporal order of the computation, it would slow users down when they wish to find out the conclusions quickly. We thus reverse the order of the workflow in both the vertical and horizontal versions of the visual user interface (see Figure 1 and Figure 6). The horizontal design is more suitable for wide-screen displays, while the vertical design can be used on portable devices and high-resolution monitors. Users may benefit from having both designs available.

Both versions of the interface were designed and developed by following an iterative design process with regular feedback from

potential users, including model-developers and ML model users. Through such feedback, we discovered that most users would prefer to observe the conclusions of the hypotheses as soon as the testing results were loaded into HypoML. They could then decide whether it would be necessary to track back to the statistical comparison and experiment results for detailed reasoning. We also discovered that double encoding used for the p -value and hypotheses had enhanced users' perception of the information and enable them to switch between overview (through visual encoding) and details on demand (through numerical values) rapidly by simply changing their visual attention. While each p -value is already encoded using the glyph and numerical value, we further encode it through its links with the testing results. The link width indicates the reverse of the p -value and the link style (i.e., solid or dashed) shows whether the difference between two sets of results is statistical significant or not (Figure 6(b2)).

While the decision state of a hypothesis is already encoded using icons in the matrix, we double encode it using black and two grey-scale values to the levels of support to the hypothesis (Figure 6(a1)). The black color draws users' attention quickly to those hypotheses that have been confirmed.

HypoML supports a set of interactions. Users are allowed to modify the threshold of p -value, which may lead to changes in the conclusions of the hypotheses and dynamical update of the whole visualization. By hovering on a p -value, users can highlight the two corresponding sets of results.

6. Results and Discussions

The testing reported in this section is primarily for testing HypoML to see if HypoML can make correct transformation from four sets of results $R_{M,D}$, $R_{M,D+}$, $R_{M+,D}$, and $R_{M+,D+}$ to visual representations of the conclusions about 12 hypotheses. The examples shown are not intended to establish the truth about the goodness of any particular ML technique, but to demonstrate the practical uses of HypoML. If a developer suspects an ML model may have a shortcoming, HypoML can help the developer confirm or reject such a hypothesis. With convolution neural networks (CNN), a common wisdom is that the deeper and the larger a CNN is, more likely a concept will be learned by the CNN. When our tests show that a particular CNN model has not learned a concept adequately, it does

not necessarily mean that a more complicate CNN model would not be able to learn the concept either. This is indeed what testing is for in software engineering. The goal of testing is to discover the shortcoming of a model or a piece of software in order to improve the model or software.

We used the Fashion MNIST dataset [XRV17] to train a CNN model for classification. The model was specified using Keras and Tensorflow in Python, and was trained and tested using the Google Colaboratory server. We use the same CNN structure as that in the official example of Keras. This CNN consists of the following layers: convolution (3x3x32, RELU), convolution (3x3x64, RELU), max pooling (2x2), dropout (25%), flatten, dense (128, RULE), dropout(50%), and dense(10, softmax). We refer readers to [ker14] for more details.

In each training session, a model is trained using 40,000 training images. With batch sizes of 128 and 50 epochs, convergence occurs in around 5 minutes. In each test, a model is tested against 6,666 test images. These images are all of 28×28 8-bit pixels. The class labels are: (0) T-shirt/top, (1) Trouser, (2) Pullover, (3) Dress, (4) Coat, (5) Sandal, (6) Shirt, (7) Sneaker, (8) Bag, and (9) Ankle boot.

The original images in the Fashion MNIST dataset feature all fashion objects in an upright position. This naturally leads to a speculation that a trained model may not be rotation invariant. One possible way to address the need for rotation-invariance is to train a model with images featuring randomly rotated objects, which is widely employed in data augmentation techniques [SSP03]. As humans can determine easily if a fashion object is in an upright position or not, one may hypothesize that a classification model may benefit from the extra information from another model that can detect the rotation angle or perform rotation normalization.

Following the workflow depicted in Figure 4, we constructed two types of data. We applied random rotation to each image in the training and testing data. This resulted in a new training dataset D_m and testing dataset D . We then created the $+$ part of the data by simply reusing the original upright images, by presuppose the existence of a rotation normalization model. As illustrated in Figure 7, each group of three images shows an original image (left), an image in D_m or D (middle), and an image in D_m+ or $D+$ (right). The middle image contains only the rotated image, together with

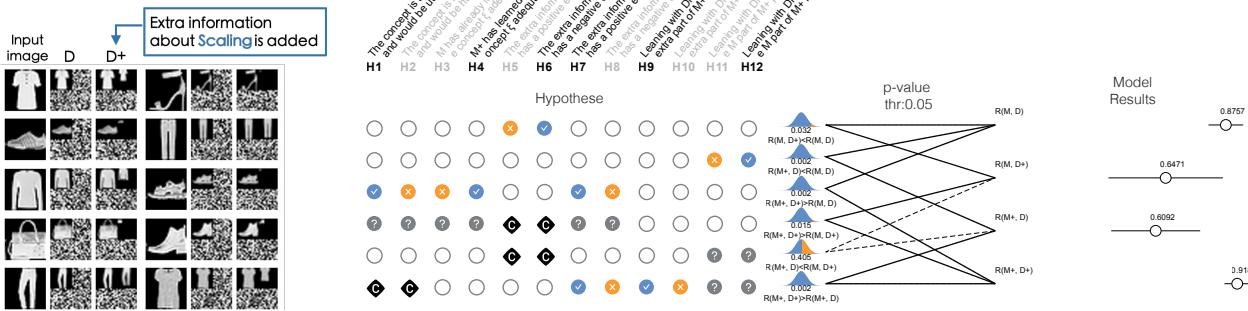


Figure 8: Data samples and the visualization of the testing results for testing the concept of scaling correction. For each sample, the stimulus in D contains an object of a “maximized” size. The stimulus in $D+$ contains an extra object of a “relative” size.

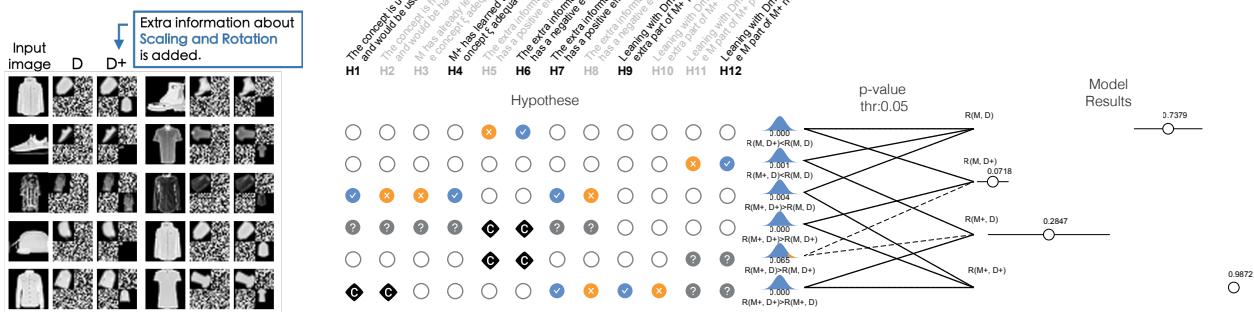


Figure 9: Testing the combined concept of rotating and scaling correction. For each sample, the stimulus in D contains an object of a rotated and “maximized” size. The stimulus in D+ contains an extra object of a “relative” size in an up-right view.

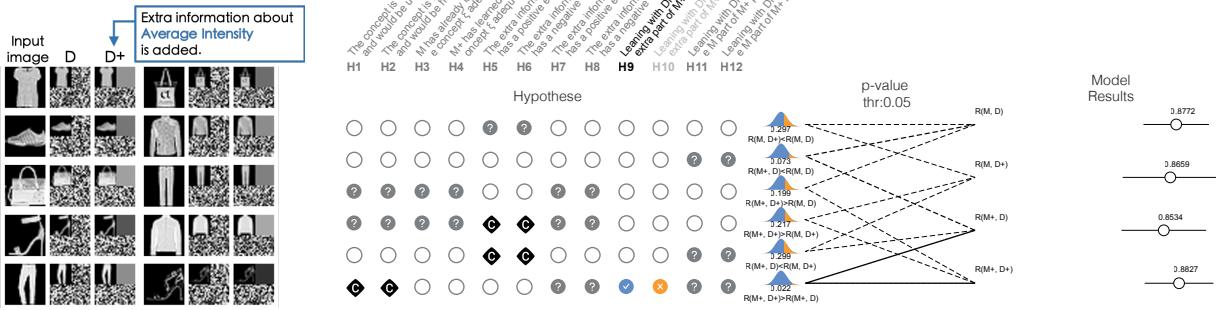


Figure 10: Testing the combined concept of average intensity. For each sample, the stimulus in D contains an original object. The stimulus in D+ contains an extra piece of information about the average intensity of the object.

noise in the other three quadrants. The right image contain both the rotated image and the normalized image, together with noise in the two lower quadrants.

We then trained two models M and M+, and tested each of them using two datasets D and D+ according to the workflow in Figure 4. From the four sets of testing results, HypoML carries out statistical and logical analysis and displays the results as shown in Figure 1. In Figure 1, we can observe that six hypotheses have been confirmed. They indicate:

- **H₁:** The concept of rotation normalization is useful to M+ and would be useful to M.
- **H₃:** M+ has learned from the concept of rotation normalization adequately.
- **H₆:** The extra information in D+, when it is fed to M, has a negative effect on M. Although M has only learned from noise the upper-right quadrant of the stimuli, when non-noise information appears in that area, it still affects M, in a negative way.
- **H₇:** The extra information in D+ (upper-right quadrant) has a positive effect on M+.
- **H₉:** Learning with D_{m+} affects the extra part of M+ positively. This is somehow anticipated because **H₁** is confirmed.
- **H₁₂:** Learning with D_{m+} affects the M part of M+ negatively,

that is, if the extra information is unavailable, M+ performs worse than M, which has not learned with the extra information.

When working with the dataset, we also noticed that the fashion objects in all images are maximized within the boundary of the image. We wondered if this would introduce some biases to a trained model. As humans can usually perceive the size of an everyday object fairly quickly, we hypothesized that a model that can remap a maximized object to a more realistic size may help the classification of such an object. As shown in Figure 8, we conducted another test by following the same workflow illustrated in Figure 4. In this case, the extra information features a scaled object on the upper-right quadrant. We measured typical sizes of fashion objects in each category and defined a relative range for the category accordingly. For the extra information, we randomly selected a scaling factor within the range defined for the corresponding category, and used the factor to scale the image. The analytical result is shown on the right of Figure 8. The conclusions are more or less the same as the hypothesis rotation normalization.

To demonstrate a slightly more complex design of a test, we combined the above two tests to examine the combined effects of the two concepts, namely rotation normalization and relative scaling. As shown in Figure 9, we used the upper-left quadrant for the rotated object as the information present in all training and testing

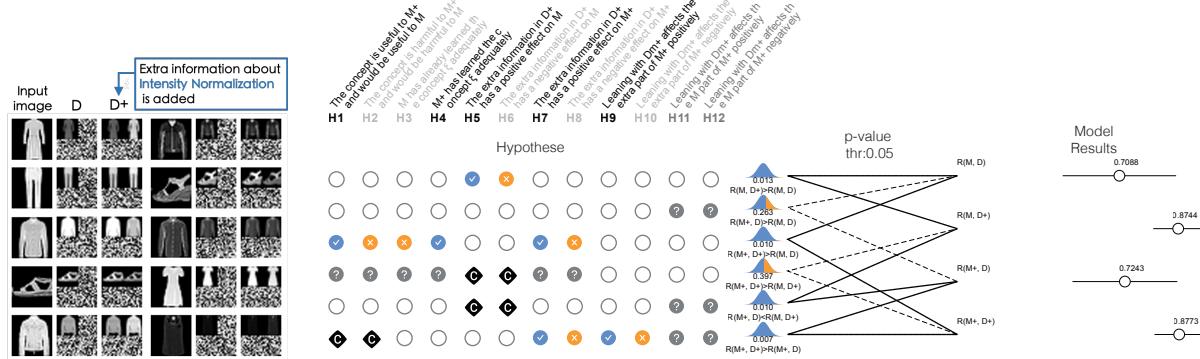


Figure 11: Testing the concept of intensity normalization. For each sample, the stimulus in D contains an original object whose intensity has been arbitrarily re-scaled. The stimulus in D+ contains an extra object featuring the original intensity as a form of normalization.

data. We placed rotation-normalized and relatively-scaled object at the lower-right quadrant. As perhaps expected, the test confirmed the same set of hypotheses as the two tests mentioned before.

In general, a CNN is expected to learn features about some aggregated properties (e.g., mean, median, or mode). We thus conducted a test to see whether providing such a feature as a piece of extra information is useful. As shown in Figure 10, we introduced the average intensity value of an object as a single-colored square in the upper-right quadrant. The analysis of the test results indicates that most hypotheses are unproven. In other words, we cannot be sure if this extra piece of information is useful or harmful. The only hypothesis that has been confirmed is **H₉**, i.e., learning with D_m+ affects the extra part of M+ positively. However, this does not translate to a confirmation of **H₇** about the overall positive impact to M+. By observing the details about how this hypothesis (i.e., **H₉**) was confirmed, we can see that it is confirmed only within the context of M+, without involving any tests about M.

Considering further about the intensity of the images, one common idealized requirement in computer vision is lighting invariance, i.e., a model can recognize the same object under different lighting conditions. We thus hypothesized that another model for normalizing the intensity of an image may help a classification model. Using a similar strategy as in the first test (random rotation), we randomly change the intensity of the original images to create the benchmark datasets D_m and D. We then use the original images as the extra information, presupposing that the original images were the results of intensity normalization.

Figure 11 shows that the extra information is useful to M+ (**H₁**), and M+ has learned the concept adequately (**H₄**). While the test confirms **H₇** and **H₉**, it is inconclusive about **H₁₁** and **H₁₂**. Interestingly, the test confirms **H₅** unexpectedly, i.e., the extra information in D+ has a positive effect on M. This is in some way related to the failure to confirm **H₁₂** as in some earlier tests. For each image in D+, the signals in the extra information (i.e., the upper-right quadrant), which in many ways is similar to those in D (i.e., upper-left quadrant). One possible explanation is the signals in the upper-right quadrant somehow strengthen the signals in the upper-left quadrant, even though M has not learned to use the extra information.

We have also conducted several other tests about the randomly-sized class labels and images with incorrect labels. HypoML has also shown to be useful for support such hypothesis testing.

7. Conclusions

In this paper, we propose a novel testing framework to aid the evaluation of ML models. In particular, this framework tests a set of hypotheses about a concept, checking whether extra information about the concept can benefit an ML model, and if so, how the extra information affects the model. The testing framework is underpinned by statistical analysis of the experiment results as well as logical inferences about the relations between six statistical conclusions and twelve hypotheses. Through an implementation of this framework HypoML, we demonstrate that with a purposely-designed visual representation, model-developers can visualize the conclusions about the twelve hypotheses as soon as the four sets of testing result data become available. This approach complements the traditional way of observing various plots for monitoring neuron activities, such as activation plots and gradient ascent plots. Model-developers, who observe any interesting patterns or failed to find desired patterns, can now formulate a concept-based hypothesis and carry out a structured test to evaluate their hypotheses.

We recognize that HypoML is only one of the many steps towards an ultimate goal of developing a powerful testing suite for evaluating, understanding, and explaining ML models. There is a need for further theoretical and practical developments in this direction, including, for instance, formulating more detailed logical analysis for sub-group analysis of the testing results, designing an advanced user interface for supporting detailed observation of sub-group analysis, and integrating with other visualization techniques for observing, understanding, and explaining ML models.

References

- [ACD*15] AMERSHI S., CHICKERING M., DRUCKER S. M., LEE B., SIMARD P., SUH J.: Modeltracker: Redesigning performance analysis tools for machine learning. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea, 2015), ACM, pp. 337–346. 2

- [Alp10] ALPAYDIN E.: *Introduction to Machine Learning*, 2nd ed. The MIT Press, 2010. 1
- [El96] ELDER IV J. F.: Machine learning, neural, and statistical classification. *Journal of the American Statistical Association* 91, 433 (1996), 436–439. 1
- [ERT*17] ENDERT A., RIBARSKY W., TURKAY C., WONG B. L. W., NABNEY I., BLANCO I. D., ROSSI F.: The state of the art in integrating machine learning into visual analytics. *Computer Graphics Forum* 36, 8 (2017), 458–486. 2
- [HPRC20] HOHMAN F., PARK H., ROBINSON C., CHAU D. H. P.: SUMMIT: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE Transactions on Visualization and Computer Graphics* 26, 1 (2020). 2
- [KAKC18] KAHNG M., ANDREWS P. Y., KALRO A., CHAU D. H.: ActiVis: visual exploration of industry-scale deep neural network models. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 88–97. 2
- [ker14] Keras CNN examples. https://keras.io/examples/mnist_cnn/, 2014. Accessed: 2019-12-03. 8
- [KPN16] KRAUSE J., PERER A., NG K.: Interacting with predictions: Visual inspection of black-box machine learning models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, CA, USA, 2016), ACM, pp. 5686–5697. 2
- [KRCP*17] KILBERTUS N., ROJAS-CARULLA M., PARASCANDOLO G., HARDT M., JANZING D., SCHÖLKOPF B.: Avoiding discrimination through causal reasoning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA, 2017), Curran Associates, pp. 656–666. 1
- [KTC*19] KAHNG M., THORAT N., CHAU D. H., VIÄL'GAS F. B., WATTENBERG M.: GAN Lab: Understanding complex deep generative models using interactive visual experimentation. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 310–320. 2
- [KGW*18] KIM B., WATTENBERG M., GILMER J., CAI C., WEXLER J., VIEGAS F., SAYRES R.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). *35th International Conference on Machine Learning* (2018). [arXiv:1711.11279](https://arxiv.org/abs/1711.11279). 3
- [LBH15] LECON Y., BENGIO Y., HINTON G.: Deep learning. *Nature* 521, 7553 (2015), 436–444. 1
- [LCJ*18] LIU D., CUI W., JIN K., GUO Y., QU H.: Deeptracker: Visualizing the training process of convolutional neural networks. *ACM Transactions on Intelligent Systems and Technology* (2018). 2
- [LSC*18] LIU M., SHI J., CAO K., ZHU J., LIU S.: Analyzing the training processes of deep generative models. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 77–87. 2
- [LSL*17] LIU M., SHI J., LI Z., LI C., ZHU J., LIU S.: Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 91–100. 2
- [MCZ*17] MING Y., CAO S., ZHANG R., LI Z., CHEN Y., SONG Y., QU H.: Understanding hidden memories of recurrent neural networks. *arXiv preprint arXiv:1710.10777* (2017). 2
- [Mun14] MUNZNER T.: *Visualization analysis and design*. AK Peters/CRC Press, 2014. 7
- [PHVG*18] PEZZOTTI N., HÖLLT T., VAN GEMERT J., LELIEVELDT B. P., EISEMANN E., VILANOVA A.: Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 98–108. 2
- [RAL*17] REN D., AMERSHI S., LEE B., SUH J., WILLIAMS J. D.: Squares: Supporting interactive performance analysis for multiclass classifiers. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 61–70. 2
- [RFFT17] RAUBER P. E., FADEL S. G., FALCAO A. X., TELEA A. C.: Visualizing the hidden activity of artificial neural networks. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 101–110. 2
- [SCD*17] SELVARAJU R. R., COGSWELL M., DAS A., VEDANTAM R., PARikh D., BATRA D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In *International Conference on Computer Vision (ICCV)* (2017), IEEE, pp. 618–626. 3
- [SDBR14] SPRINGENBERG J. T., DOSOVITSKIY A., BROX T., RIEDMILLER M.: Striving for simplicity: The all convolutional net, 2014. [arXiv:1412.6806](https://arxiv.org/abs/1412.6806). 2, 3
- [SGPR18] STROBELT H., GEHRMANN S., PFISTER H., RUSH A. M.: LSTMVis: A tool for visual analysis of hidden state dynamics in recurrent neural networks. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 667–676. 2
- [SKKC19] SACHA D., KRAUS M., KEIM D. A., CHEN M.: VIS4ML: An ontology for visual analytics assisted machine learning. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 385–395. 2
- [SSP03] SIMARD P. Y., STEINKRAUS D., PLATT J.: Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition* (August 2003), Institute of Electrical and Electronics Engineers, Inc. 8
- [TKC01] TAM G. K. L., KOTHARI V., CHEN M.: An analysis of machine- and human-analytics in classification. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2011), 71–80. 2
- [WGYS18] WANG J., GOU L., YANG H., SHEN H.: Ganviz: A visual analytics approach to understand the adversarial game. *IEEE Transactions on Visualization and Computer Graphics* 24, 6 (2018), 1905–1917. 2
- [XRV17] XIAO H., RASUL K., VOLLMGRAF R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. [arXiv:1708.07747](https://arxiv.org/abs/1708.07747). 8
- [YCFL15] YOSINSKI J., CLUNE J., FUCHS T., LIPSON H.: Understanding neural networks through deep visualization. In *Proceedings of the 32nd International Conference on Machine Learning* (Lille, France, 2015). 3
- [ZF14] ZEILER M. D., FERGUS R.: Visualizing and understanding convolutional networks. In *Proc. 13th European Conference on Computer Vision*. Springer, 2014, pp. 818–833. 2
- [ZKL*16] ZHOU B., KHOSLA A., LAPEDRIZA A., OLIVA A., TORRALBA A.: Learning deep features for discriminative localization. In *Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), IEEE, pp. 2921–2929. 3
- [ZWM*19] ZHANG J., WANG Y., MOLINO P., LI L., EBERT D. S.: Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 364–373. 2
- [ZWW18] ZHANG L., WU Y., WU X.: Achieving non-discrimination in prediction. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI (Stockholm, Sweden, 2018)*, ijcai.org, pp. 3097–3103. 1
- [ZZ18] ZHANG Q.-S., ZHU S.-C.: Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering* 19, 1 (Jan 2018), 27–39. URL: <http://dx.doi.org/10.1631/FITEE.1700808>, doi:10.1631/fitee.1700808. 2