This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

CNN EXPLAINER: Learning Convolutional Neural Networks with Interactive Visualization

Zijie J. Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das,
Fred Hohman, Minsuk Kahng, and Duen Horng (Polo) Chau

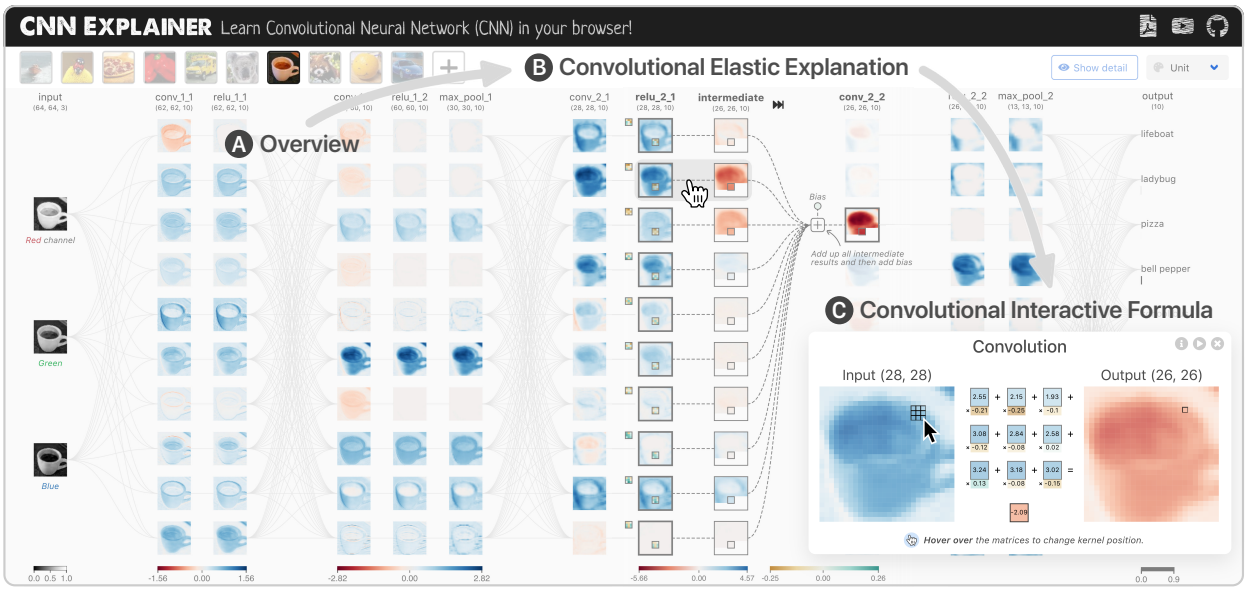


Fig. 1. With CNN EXPLAINER, learners can visually examine how *Convolutional Neural Networks* (CNNs) transform input images into classification predictions (e.g., predicting *espresso* for an image of a coffee cup), and interactively learn about their underlying mathematical operations. In this example, a learner uses CNN EXPLAINER to understand how convolutional layers work through three tightly integrated views, each explaining the convolutional process in increasing levels of detail. **(A)** The *Overview* visualizes a CNN architecture where each neuron is encoded as a square with a heatmap representing the neuron's output. **(B)** Clicking a neuron reveals how its activations are computed by the previous layer's neurons, displaying the often-overlooked intermediate computation through animations of sliding kernels. **(C)** *Convolutional Interactive Formula View* for inspecting underlying mathematics of the dot-product operation core to convolution. For clarity, some annotations are removed and views are re-positioned.

Abstract— Deep learning's great success motivates many practitioners and students to learn about this exciting technology. However, it is often challenging for beginners to take their first step due to the complexity of understanding and applying deep learning. We present CNN EXPLAINER, an interactive visualization tool designed for non-experts to learn and examine convolutional neural networks (CNNs), a foundational deep learning model architecture. Our tool addresses key challenges that novices face while learning about CNNs, which we identify from interviews with instructors and a survey with past students. CNN EXPLAINER tightly integrates a model overview that summarizes a CNN's structure, and on-demand, dynamic visual explanation views that help users understand the underlying components of CNNs. Through smooth transitions across levels of abstraction, our tool enables users to inspect the interplay between low-level mathematical operations and high-level model structures. A qualitative user study shows that CNN EXPLAINER helps users more easily understand the inner workings of CNNs, and is engaging and enjoyable to use. We also derive design lessons from our study. Developed using modern web technologies, CNN EXPLAINER runs locally in users' web browsers without the need for installation or specialized hardware, broadening the public's education access to modern deep learning techniques.

Index Terms—Deep learning, machine learning, convolutional neural networks, visual analytics

1 INTRODUCTION

Deep learning now enables many of our everyday technologies. Its continued success and potential application in diverse domains has attracted immense interest from students and practitioners who wish to learn and apply this technology. However, many beginners find it challenging to take the first step in studying and understanding deep learning concepts. For example, convolutional neural networks (CNNs), a foundational deep learning model architecture, is often one of the first and most widely used models that students learn. CNNs are often used in image classification, achieving state-of-the-art performance [33]. However, through interviews with deep learning instructors and a survey of past students, we found that even for this "introductory" model, it can be challenging for beginners to understand how inputs (e.g., image data) are transformed into class predictions. This steep learning curve stems from CNN's complexity, which typically leverages many computational layers to reach a final decision. Within a CNN, there are many types of

- *Zijie J. Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, and Duen Horng Chau are with Georgia Tech. E-mail: {jayw|rturko3|oshaikh|haekyu|nilakshdas|fredhohman|polo}@gatech.edu.*
- *Minsuk Kahng is with Oregon State University. E-mail: minsuk.kahng@oregonstate.edu.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

network layers (e.g., fully-connected, convolutional, activation), each with a different structure and underlying mathematical operations. Thus, a student needs to develop a mental model of not only how each layer operates, but also how to choose different layers that work together to transform data. Therefore, a key challenge in learning about CNNs is the intricate interplay between *low-level mathematical operations* and *high-level integration* of such operations within the network.

Key challenges in designing learning tools for CNNs. There is a growing body of research that uses interactive visualization to explain the complex mechanisms of modern machine learning algorithms, such as TensorFlow Playground [50] and GAN Lab [29], which help students learn about dense neural networks and generative adversarial networks (GANs) respectively. Regarding CNNs, some existing visualization tools focus on demonstrating the high-level model structure and connections between layers (e.g., Harley's Node-Link Visualization [20]), while others focus on explaining the low-level mathematical operations (e.g., Karpathy's interactive CNN demo [30]). There is no visual learning tool that explains and connects CNN concepts from both levels of abstraction. This interplay between global model structure and local layer operations has been identified as one of the main obstacles to learning deep learning models, as discussed in [50] and corroborated from our interviews with instructors and student survey. CNN EXPLAINER aims to bridge this critical gap.

Contributions. In this work, we contribute:

- **CNN EXPLAINER, an interactive visualization tool designed for non-experts** to learn about both CNN's high-level model structure and low-level mathematical operations, addressing learners' key challenge in connecting unfamiliar layer mechanisms with complex model structures. Our tool advances over prior work [20, 30], overcoming unique design challenges identified from a literature review, instructor interviews and a survey with past students (Sect. 4).
- **Novel interactive system design** of CNN EXPLAINER (Fig. 1), which adapts familiar techniques such as *overview + detail* and *animation* to simultaneously summarize intricate model structure, while providing context for users to inspect detailed mathematical operations. CNN EXPLAINER's visualization techniques work together through fluid transitions between different abstraction levels (Fig. 2), helping users gain a more comprehensive understanding of complex concepts within CNNs (Sect. 6).
- **Design lessons distilled from user studies** on an interactive visualization tool for machine learning education. While visual and interactive approaches have been gaining popularity in explaining machine learning concepts to non-experts, little work has been done to evaluate such tools [28, 43]. We interviewed four instructors who have taught CNNs and conducted a survey with 19 students who have previously learned about CNNs to identify the needs and challenges for a deep learning educational tool (Sect. 4). In addition, we conducted an observational study with 16 students to evaluate the usability of CNN EXPLAINER, and investigated how our tool could help students better understand CNN concepts (Sect. 8). Based on these studies, we discuss the advantages and limitations of interactive visual educational tools for machine learning.
- **An open-source, web-based implementation** that broadens the public's education access to modern deep learning techniques without the need for advanced computational resources. Deploying deep learning models conventionally requires significant computing resources, e.g., servers with powerful hardware. In addition, even with a dedicated backend server, it is challenging to support a large number of concurrent users. Instead, CNN EXPLAINER is developed using modern web technologies, where all results are directly and efficiently computed in users' web browsers (Sect. 6.7). Therefore, anyone can access CNN EXPLAINER using their web browser without the need for installation or a specialized backend. Our code is open-sourced[1] and

[1]Code: https://github.com/poloclub/cnn-explainer

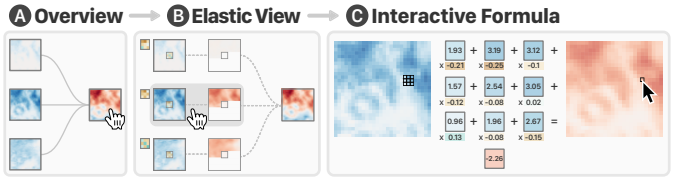


Fig. 2. In CNN EXPLAINER, tightly integrated views with different levels of abstractions work together to help users more easily learn about the intricate interplay between a CNN's high-level structure and low-level mathematical operations. **(A)** the *Overview* summarizes connections of all neurons; **(B)** the *Elastic View* animates the intermediate convolutional computation of the user-selected neuron in the *Overview*; and **(C)** *Interactive Formula* interactively demonstrates the detailed calculation on the selected input in the *Elastic View*.

CNN EXPLAINER is available at the following public demo link: https://poloclub.github.io/cnn-explainer.

Broadening impact of visualization for AI. In recent years, many visualization systems have been developed for deep learning, but very few are designed for non-experts [20, 29, 44, 50], as surveyed in [23]. CNN EXPLAINER joins visualization research that introduces beginners to modern machine learning concepts. Applying visualization techniques to explain the inner workings of complex models has great potential. We hope our work will inspire further research and development of visual learning tools that help democratize and lower the barrier to understanding and applying artificial intelligent technologies.

2 BACKGROUND FOR CONVOLUTIONAL NEURAL NETWORKS

This section provides a high-level overview of convolutional neural networks (CNNs) in the context of image classification, which will help ground our work throughout this paper.

Image classification has a long history in the machine learning research community. The objective of supervised image classification is to map an input image, X, to an output class, Y. For example, given a cat image, a sophisticated image classifier would output a class label of "cat". CNNs have demonstrated state-of-the-art performance on this task, in part because of their multiple layers of computation that aim to learn a better representation of image data.

CNNs are composed of several different layers (e.g., convolutional layers, downsampling layers, and activation layers)—each layer performs some predetermined function on its input data. Convolutional layers "extract features" to be used for image classification, with early convolutional layers in the network extracting low-level features (e.g., edges) and later layers extracting more-complex semantic features (e.g., car headlights). Through a process called backpropagation, a CNN learns kernel weights and biases from a collection of input images. These values also known as parameters, which summarize important features within the images, regardless of their location. These kernel weights slide across an input image performing an element-wise dot-product, yielding intermediate results that are later summed together with the learned bias value. Then, each neuron gets an output based on the input image. These outputs are also called activation maps. To decrease the number of parameters and help avoid overfitting, CNNs downsample inputs using another type of layer called pooling. Activation functions are used in a CNN to introduce non-linearity, which allows the model to learn more complex patterns in data. For example, a Rectified Linear Unit (ReLU) is defined as $\max(0, x)$, which outputs the positive part of its argument. These functions are also often used prior to the output layer to normalize classification scores, for example, the activation function called Softmax performs a normalization on unscaled scalar values, known as logits, to yield output class scores that sum to one. To summarize, compared to classic image classification models that can be over-parameterized and fail to take advantage of inherent properties in image data, CNNs create spatially-aware representations through multiple stacked layers of computation.

3 Related Work

3.1 Visualization for Deep Learning Education

Researchers and practitioners have been developing visualization systems that aim to help beginners learn about deep learning concepts. Teachable Machine [9] teaches the basic concept of machine learning classification, such as overfitting and underfitting, by allowing users to train a deep neural network classifier with data collected from their own webcam or microphone. The Deep Visualization Toolbox [58] also uses live webcam images to interactively help users to understand what each neuron has learned. These deep learning educational tools feature direct model manipulation as core to their experience. For example, users learn about CNNs, dense neural networks, and GANs through experimenting with model training in ConvNetJS MNIST demo [30], TensorFlow Playground [50], and GAN Lab [29], respectively. Beyond 2D visualizations, Node-Link Visualization [20] and TensorSpace [3] demonstrate deep learning models in 3D space. Inspired by Chris Olah's interactive blog posts [44], interactive articles explaining deep learning models with interactive visualization are gaining popularity as an alternative medium for education [10, 39].

Most existing educational resources focus on explaining either the high-level model structures and training process or the low-level mathematics, but not both. However, we found that one key challenge for beginners learning about deep learning models is the difficulty connecting unfamiliar layer mechanisms with complex model structures (discussed in Sect. 4). For example, TensorFlow Playground [50], one of the few yet popular deep learning educational tools, focuses on helping users develop intuition about the effects of different *dense neural network* architectures, but does not explain the underlying mathematical operations. TensorFlow Playground also operates on synthetic 2D data, which can be challenging for users to transfer newly learned concepts to more realistic data and models. In comparison, our work explains both model structure and mathematics of *CNNs*, a more complex architecture, with real image data.

3.2 Algorithm Visualization

Before deep learning started to attract interest from students and practitioners, visualization researchers have been studying how to design algorithm visualizations (AV) to help people learn about dynamic behavior of various algorithms [7, 26, 48]. These tools often graphically represent data structures and algorithms using interactive visualization and animations [7, 14, 18]. While researchers have found mixed results on AV's effectiveness in computer science education [8, 13, 16], growing evidence has shown that student engagement is the key factor for successfully applying AV in education settings [25, 42]. Naps, et al. defined a taxonomy of six levels of engagement[2] at which learners can interact with AVs [42], and studies have shown higher engagement level leads to better learning outcomes [13, 19, 32, 47].

Deep learning models can be viewed as specialized algorithms comprised of complex and stochastic interactions between multiple different computational layers. However, there has been little work in designing and evaluating visual educational tools for deep learning in the context of AV. CNN Explainer's design draws inspiration from the guidelines proposed in AV literature (discussed in Sect. 5); our user study results also corroborate some of the key findings in prior AV research (discussed in Sect. 8.3). Our work advances AV's landscape in covering modern and pervasive machine learning algorithms.

3.3 Visual Analytics for Neural Networks & Predictions

Many visual analytics tools have been developed to help deep learning experts analyze their models and predictions [5, 15, 23, 27, 36, 37]. These tools support many different tasks. For example, recent work such as Summit [24] uses interactive visualization to summarize what features a CNN model has learned and how those features interact and attribute to model predictions. LSTMVis [54] makes long short-term memory (LSTM) networks more interpretable by visualizing the model's hidden states. Similarly, GANVis [56] helps experts to interpret what a trained generative adversarial network (GAN) model has learned. People also use visual analytics tools to diagnose and monitor the training process of deep learning models. Two examples, DGMTracker [35] and DeepEyes [46], help developers better understand the training process of CNNs and GANs, respectively. Also, visual analytics tools recently have been developed to help experts detect and interpret the vulnerability in their deep learning models [12, 34]. These existing analytics tools are designed to assist experts in analyzing their model and predictions, however, we focus on non-experts and learners, helping them more easily learn about deep learning concepts.

[2]Six engagement categories: *No Viewing*, *Viewing*, *Responding*, *Changing*, *Constructing*, *Presenting*.

4 Formative Research & Design Challenges

Our goal is to build an interactive visual learning tool to help students gain understanding of key CNN concepts to design their own models. To identify the learning challenges faced by the students, we conducted interviews with deep learning instructors and surveyed past students.

Instructor interviews. To inform our tool's design, we recruited 4 instructors (2 female, 2 male) who have taught CNNs in a large university. We refer to them as T1-T4 throughout our discussion. One instructor teaches computer vision, and the others teach deep learning. We interviewed them one-on-one in a conference room (3/4) and via a video-conferencing software (1/4); each interview lasted around 30 minutes. Through these semi-structured interviews, we learned that (1) instructors currently rely on simple illustrations with toy examples to explain CNN concepts, and an interactive tool like TensorFlow Playground with real image inputs would be highly appreciated; and (2) key challenges exist for instructors teaching and students learning about CNNs, which informed us to design a student survey.

Student survey. After the interviews, we recruited students from a large university who have previously studied CNNs to fill out an online survey. We received 43 responses, and 19 of them (4 female, 15 male) met the criteria. Among these 19 participants, 10 were Ph.D. students, 3 were M.S. students, 5 were undergraduates, and 1 was a faculty member. We asked participants what were "the biggest challenges in studying CNNs" and "the most helpful features if there was a visualization tool for explaining CNNs to beginners". We provided pre-selected options based on the prior instructor interviews, but participants could write down their own response if it was not included in the options. The aggregated results of this survey are shown in Fig. 3.

Together with a literature review, we synthesized our findings from these two studies into the following five design challenges (C1-C5).

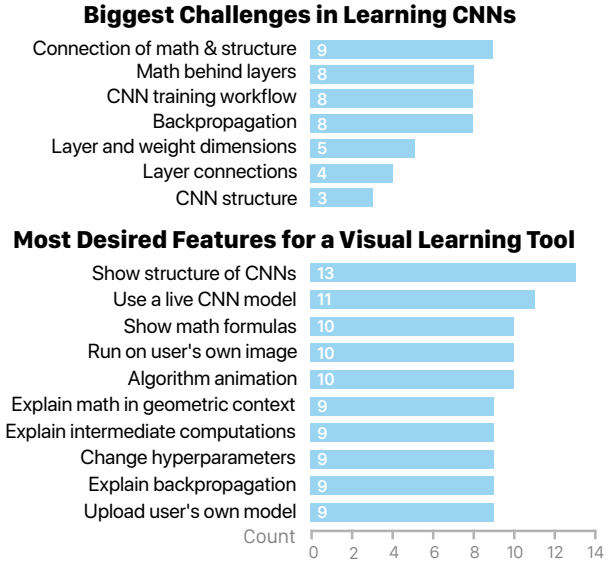


Fig. 3. Survey results from 19 participants who have previously learned about CNNs. **Top:** Biggest challenges encountered during learning. **Bottom:** Desired features for an interactive visual learning tool for CNNs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

C1. Intricate model structure. CNN models consist of many layers, each having a different structure and underlying mathematical functions [33]. Fewer past students listed CNN structure as their biggest challenge, but most of them believe a visual learning tool should explain the structure (Fig. 3), as the complex construction of CNNs can be overwhelming, especially for beginners who just started learning. T2 said "*It can be very hard for them [students with less knowledge of neural networks] to understand the structure of CNNs, you know, the connections between layers.*"

C2. Complex layer operations. Different layers serve different purposes in CNNs [17]. For example, convolutional layers exploit the spatially local correlations in inputs—each convolutional neuron connects to only a small region of its input; whereas max pooling layers introduce regularization to prevent overfitting. T1 said, "*The most challenging part is learning the math behind it [CNN model].*" Many students also reported that CNN layer computations are the most challenging learning objective (Fig. 3). To make CNNs perform better than other models in tasks like image classification, these models have complex and unique mathematical operations that many beginners may not have seen elsewhere.

C3. Connection between model structure and layer operation. Based on instructor interviews and the survey results from past students (Fig. 3), one of the cruxes to understand CNNs is understanding the interplay between low-level mathematical operations (**C2**) and the high-level model structure (**C1**). Smilkov et al., creators of the popular dense neural network learning tool Tensorflow Playground [50], also found this challenge key to learning about deep learning models: "*It's not trivial to translate the equations defining a deep network into a mental model of the underlying geometric transformations [change of feature representations].*" In other words, in addition to comprehending the mathematical formulas behind different layers, students are also required to understand how each operation works within the complex, layered model structure.

C4. Effective algorithm visualization (AV). The success of applying visualization to explain machine learning algorithms to beginners [9,29,50] suggests that an AV tool is a promising approach to help people more easily learn about CNNs. However, AV tools need to be carefully designed to be effective in helping learners gain an understanding of algorithms [13]. In particular, AV systems need to clearly explain the mapping between the algorithm and its visual encoding [40], and actively engage learners [32].

C5. Challenge in deploying interactive learning tools. Most neural networks are written in deep learning frameworks, such as TensorFlow [4] and PyTorch [45]. Although these libraries have made it much easier to create AI models, they require users to understand key concepts of deep learning in the first place [53]. Can we make understanding CNNs more accessible without installation and coding, so that everyone has the opportunity to learn and interact with deep learning models?

The above design challenges cover most of the desired features (Fig. 3). We assessed the feasibility to also support explaining backpropagation in the same tool, and we concluded that its effective explanation will necessitate designs that are hard to be unified (e.g., backpropagation Algorithm [2]) . Indeed, T1 commented that "*Deriving backpropagation is applying a series chain rules [...] It doesn't really make sense to visualize the gradients [in our tool].*" Supporting the training process would require client-side in-browser computation on many data examples, which incur both high amount of data download and slow convergence ([29,30]). Therefore, as the first prototype, we decided for CNN EXPLAINER to focus on explaining inference after a model has been trained. We plan to support the explanation for backpropagation and training process as future work (Sect. 9).

5 DESIGN GOALS

Based on the identified design challenges (Sect. 4), we distill the following key design goals (**G1G5**) for CNN EXPLAINER, an interactive visualization tool to help students more easily learn about CNNs.

G1. Visual summary of CNN models and data flow. Based on the survey results, showing the structure of CNNs is the most desired feature for a visual learning tool (Fig. 3). Therefore, to give users an overview of the structure of CNNs, we aim to create a visual summary of a CNN model by visualizing all layer outputs and connections in one view. This could help users to visually track how input image data are transformed to final class predictions through a series of layer operations (**C1**). (Sect. 6.1)

G2. Interactive interface for mathematical formulas. Since CNNs employ various complex mathematical functions to achieve high classification performance, it is important for users to understand each mathematical operation in detail (**C2**). In response, we would like to design an interactive interface for each mathematical formula, enabling users to examine and better understand the inner-workings of layers. (Sect. 6.3)

G3. Fluid transition between different levels of abstraction. To help users connect low-level layer mathematical mechanisms to high-level model structure (**C3**), we would like to design a focus + context display of different views, and provide smooth transitions between them. By easily navigating through different levels of CNN model abstraction, users can get a holistic picture of how CNN works. (Sect. 6.4)

G4. Clear communication and engagement. Our goal is to design and develop an interactive system that is easy to understand and engaging to use so that it can help people to more easily learn about CNNs (**C4**). We aim to accompany our visualizations with explanations to help users to interpret the graphical representation of the CNN model (Sect. 6.5), and we wish to actively engage learners through visualization customizations. (Sect. 6.6)

G5. Web-based implementation. To develop an interactive visual learning tool that is accessible for users without installation and coding (**C5**), we would like to use modern web browsers as the platform to explain the inner-workings of a CNN model, where users can access directly on their laptops or tablets. We also open-source our code to support future research and development of deep learning educational tools. (Sect. 6.7)

6 VISUALIZATION INTERFACE OF CNN EXPLAINER

CNN EXPLAINER's interface is built on our prior prototype [57]. We visualize the forward propagation, i.e., transforming an input image into a class prediction, of a trained model (Fig. 4). Users can explore a CNN at different levels of abstraction through the tightly integrated *Overview* (Sect. 6.1), *Elastic Explanation View* (Sect. 6.2), and the *Interactive Formula View* (Sect. 6.3). Our tool allows users to smoothly transition between these views (Sect. 6.4), provides text annotations and a tutorial article to help users interpret the visualizations (Sect. 6.5), and engages them to test hypotheses through visualization customizations (Sect. 6.6). The system is targeted towards beginners and describes all mathematical operations necessary for a CNN to classify an image.

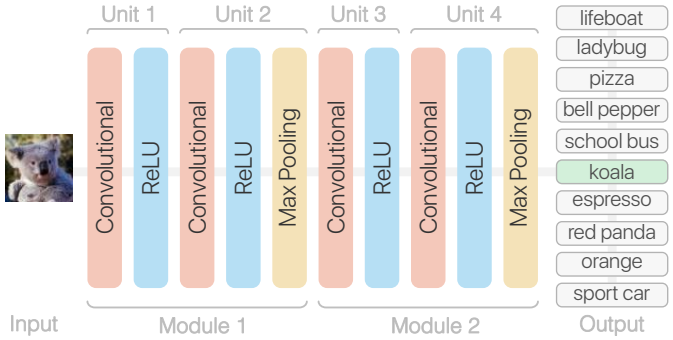


Fig. 4. Illustration of *Tiny VGG* model used in CNN EXPLAINER: this model uses the same, but fewer, convolutional layers as the original VGGNet model [49]. We trained it to classify 10 classes of images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

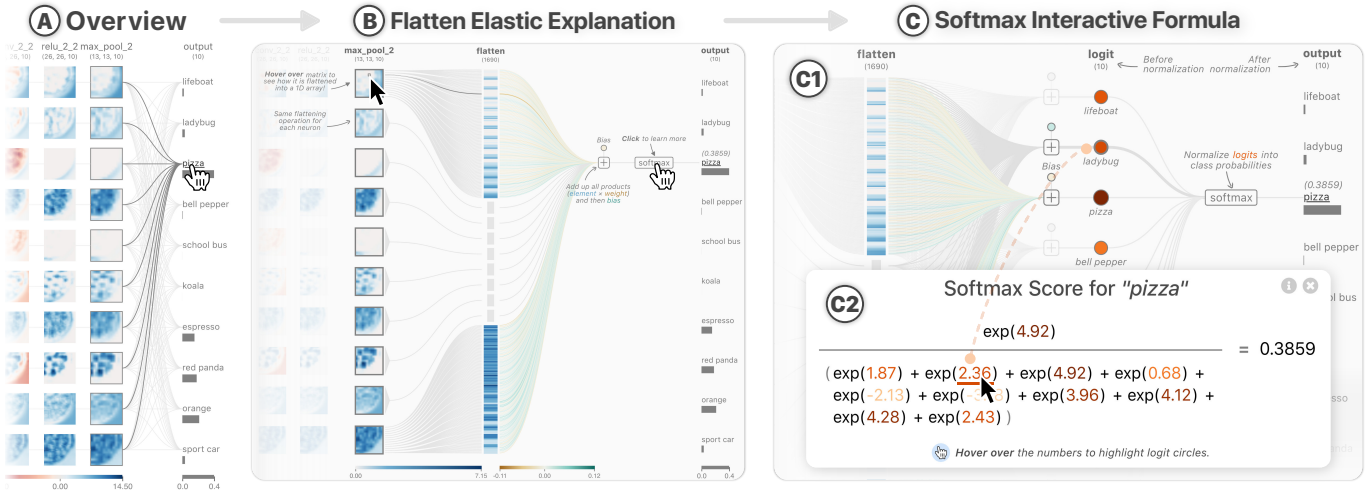
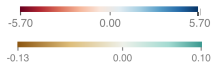


Fig. 5. CNN EXPLAINER helps users learn about the connection between the `output` layer and its previous layer via three tightly integrated views. Users can smoothly transition between these views to gain a more holistic understanding of the `output` layer's lifeboat prediction computation. **(A)** The *Overview* summarizes neurons and their connections. **(B)** The *Flatten Elastic Explanation View* visualizes the often-overlooked flatten layer, helping users more easily understand how a high-dimensional `max_pool_2` layer is connected to the 1-dimensional `output` layer. **(C)** The *Softmax Interactive Formula View* further explains how the softmax function that precedes the `output` layer normalizes the penultimate computation results (i.e., logits) into class probabilities through linking the **(C1)** numbers from the formula to **(C2)** their visual representations within the model structure.

Color scales are used throughout the visualization to show the impact of weight, bias, and activation map values. Consistently in the interface, a **red** to **blue** color scale is used to visualize neuron activation maps as heatmaps, and a **yellow** to **green** color scale represents weights and biases. A persistent color scale legend is present across all views, so the user always has context for the displayed colors. We chose these distinct, diverging color scales with white representing zero, so that a user can easily differentiate positive and negative values. We group layers in the *Tiny VGG* model, our CNN architecture, into four units and two modules (Fig. 4). Each unit starts with one convolutional layer. Both modules are identical and contain the same sequence of operations and hyperparameters. To analyze neuron activations throughout the network with varying contexts, users can alter the range of the heatmap color scale (Sect. 6.6).



6.1 Overview

The *Overview* (Fig. 1A, Fig. 5A) is the opening view of CNN EXPLAINER. This view represents the high-level structure of a CNN: neurons grouped into layers with distinct, sequential operations. It shows neuron activation maps for all layers represented as heatmaps with a diverging **red** to **blue** color scale. Neurons in consecutive layers are connected with edges, which connect each neuron to its inputs; to see these edges, users simply can hover over any activation map. In the model, neurons in convolutional layers and the `output` layer are fully connected to the previous layer, while all other neurons are only connected to one neuron in the previous layer.

6.2 Elastic Explanation View

The *Elastic Explanation Views* visualize the computations that leads to an intermediate result without overwhelming users with low-level mathematical operations. CNN EXPLAINER enters two elastic views after a user clicks a convolutional or an output neuron from the *Overview*. After the transition, far-away heatmaps and edges fade out to help users focus on the selected layers while providing CNN structural context in the background (Fig. 1A).

Explaining the Convolutional Layer (Fig. 1B). The *Convolutional Elastic Explanation View* applies a convolution on each input node of the selected neuron, visualized by a kernel sliding across the input neurons, which yields an intermediate result for each input neuron. This sliding kernel forms the output heatmap during the animation, which imitates the internal process during a convolution operation. While the sliding kernel animation is in progress, the edges in this view are represented as flowing-dashed lines; upon the animations completion, the edges transition to solid lines.

Explaining the Flatten Layer (Fig. 5B). The *Flatten Elastic Explanation View* visualizes the operation of transforming an n-dimensional tensor into a 1-dimensional tensor by traversing pixels in row-major order. This flattening operation is often necessary in a CNN prior to classification so that the fully-connected `output` layer can make classification decisions. The view represents each neuron in the `flatten` layer as a short line whose color is the same as its source pixel in the previous layer. Then, edges connect these neurons with their source components and intermediate results. These edges are colored based on the model's weight value. Users can hover over any component of this connection to highlight the associated edges as well as the `flatten` layer's neuron and the pixel value from the previous layer.

6.3 Interactive Formula View

The *Interactive Formula View* consists of four variations designed for convolutional layers, ReLU activation layers, pooling layers, and the softmax activation function. After users have built up a mental model of the CNN model structure from the previous *Overview* and *Elastic Explanation Views*, these four views demonstrate the detailed mathematics occurring in each layer.

Explaining Convolution, ReLU Activation, and Pooling (Fig. 6A, B, C)) Each view animates the window-sliding operation on the input matrix and output matrix over an interval, so that the user can understand how each element in the input is connected to the output, and vice versa. In addition, the user can interact with the these matrices by hovering over the heatmaps to control the position of the sliding window. For example, in the *Convolutional Interactive Formula View* (Sect. 6.3A), as the user controls the window (kernel) position in either the input or the output matrix, this view visualizes the dot-product formula with input numbers and kernel weights directly extracted from the current kernel. This synchronization between the input, the output and the mathematical function enables the user to better understand how the kernel convolves a matrix in convolutional layers.

Explaining the Softmax Activation (Fig. 6D). This view outlines the operations necessary to calculate the classification score. It is accessible from the *Flatten Elastic Explanation View* to explain how the results (logits) from the previous view lead to the final classification. The view consists of logit values encoded as circles and colored with a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

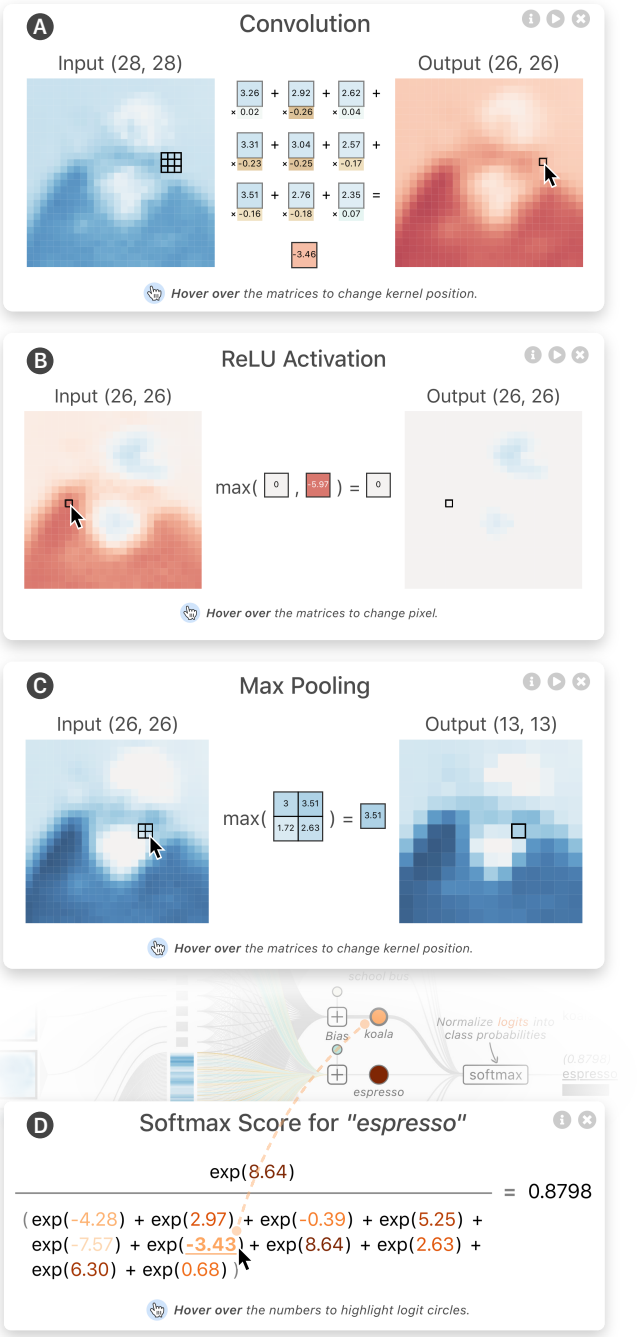


Fig. 6. The *Interactive Formula Views* explain the underlying mathematical operations of a CNN. **(A)** shows the element-wise dot-product occurring in a convolutional neuron, **(B)** visualizes the activation function ReLU, and **(C)** illustrates how max pooling works. Users can hover over heatmaps to display an operation's input-to-output mapping. **(D)** interactively explains the softmax function, helping users connect numbers from the formula to their visual representations. Users can click the info button to scroll to the corresponding section in the tutorial article, and the play button to start the window sliding animation in (A)-(C).

light orange to dark orange color scale, which provides users with a visual cue of the importance of every class. This view also includes a corresponding equation, which explains how the classification score is computed. When users enter this view, pairs of each logit circle and its corresponding value in the equation appear sequentially with animations. As a user hovers over a logit circle, its value will be highlighted in the equation along with the logit circle itself, so the user can understand how each logit contributes to the softmax function. Hovering over numbers in the equation will also highlight the appropriate logit circles. Interacting with logit circles and the mathematical equation in combination allows a user to discern the impact that every logit has on the classification score in the `output` layer.

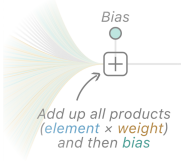
6.4 Transitions Between Views

The *Overview* is the starting state of CNN EXPLAINER and shows the model architecture. From this high-level view, the user can begin inspecting layers, connectivity, classifications, and tracing activations of neurons through the model. When a user is interested in more detail, they can click on neuron activation maps in the visualization. Neurons in a layer that have simple one-to-one connections to a neuron in the previous layer do not require an auxiliary *Elastic Explanation View*, so upon clicking one of these neurons, a user will be able to enter the *Interactive Formula View* to understand the low-level operation that a tensor undergoes at that layer. If a neuron has more complex connectivity, then the user will enter an *Elastic Explanation View* first. In this view, CNN EXPLAINER uses visualizations and annotations before displaying mathematics. Through further interaction, a user can hover and click on parts of the *Elastic Explanation View* to uncover the mathematical operations as well as examine the values of weights and biases. The low-level *Interactive Formula Views* are only shown after transitioning from the previous two views, so that users can learn about the underlying mathemtical operations after hainvg a mental model of the complex and layered CNN model structure.

6.5 Visualizations with Explanations

CNN EXPLAINER is accompanied by an interactive tutorial article beneath the interface that explains CNN layer functions, hyperparameters, and outlines CNN EXPLAINER's interactive features. Learners can read freely, or jump to specific sections by clicking layer names or the info buttons (Fig. 6) from the main visualization. The article provides beginner users detailed information regarding CNNs that can supplement their exploration of the visualization.

Additionally, text annotations are placed throughout the visualization (e.g., explaining the flatten layer operation in the right image), which further guide users and explain concepts that are not easily discernible from the visualization alone. These annotations help users map the underlying algorithm to its visual encoding.



6.6 Customizable Visualizations

The *Control Panel* located across the top of the visualization (Fig. 1) allows the user to alter the CNN input image and edit the overall representation of the network. The *Hyperparameter Widget* (Fig. 7) enables the user to experiment with differnt convolution hyperparameters.

Change input image. Users can choose between (1) preloaded input images for each output class, or (2) upload their own custom image. Preloaded images allow a user to easily access data from the classes the model was originally trained on. User can also freely upload any image for classification into the ten classes the network was trained on. CNN EXPLAINER resizes a user's image while preserving the aspect ratio to fit one dimension of the model input size, and then crop the central region if the other dimensions does not match. The fourth of six AV tool engagement levels is allowing users to change the AV tool's input [42]. Supporting custom image upload engages users, by allowing them to analyze the network's classification decisions and interactively testing their own hypotheses on diverse image inputs.

Show network details. A user can toggle the "Show detail" button, which displays additional network specifications in the *Overview*. When toggled on, the *Overview* will reveal layer dimensions and show color scale legends. Additionally, a user can vary the activation map color scale range. The CNN architecture presented by CNN EXPLAINER is grouped into four units and two modules (Fig. 4). By modifying the drop-down menu in the *Control*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

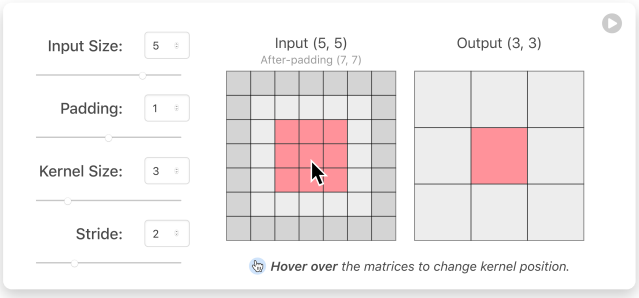


Fig. 7. The *Hyperparameter Widget*, a component of the accompanying interactive article, allows users to adjust hyperparameters and observe in real time how the kernel's sliding pattern changes in convolutional layers.

Panel, a user can adjust the color scale range used by the network to investigate activations with different groupings.

Explore hyperparameter impact. The tutorial article (Sect. 6.5) includes an interactive *Hyperparameter Widget* that allows users to experiment with convolutional hyperparameters (Fig. 7). Users can adjust the input and hyperparameters of the stand-alone visualization to test how different hyperparameters change the sliding convolutional kernel and the output's dimensions. This interactive element emphasizes learning through experimentation by supplementing knowledge gained from reading the article and using the main visualization.

6.7 Web-based, Open-sourced Implementation

CNN EXPLAINER is a web-based, open-sourced visualization tool to teach students the foundations of CNNs. A new user only needs a modern web-broswer to access our tool, no installation required. Additionally, other datasets and linear models can be quickly applied to our visualization system due to our robust implementation.

Model Training. The CNN architecture, Tiny VGG (Fig. 4), presented by CNN EXPLAINER for image classification is inspired by both the popular deep learning architecture, VGGNet [49], and Stanford's CS231n course notes [31]. It is trained on the Tiny ImageNet dataset [1]. The training dataset consists of 200 image classes and contains 100,000 64×64 RGB images, while the validation dataset contains 10,000 images across the 200 image classes. The model is trained using *TensorFlow* [4] on 10 handpicked, everyday classes: lifeboat, ladybug, bell pepper, pizza, school bus, koala, espresso, red panda, orange, and sport car. During the training process, the batch size and learning rate are fine-tuned using a 5-fold-cross-validation scheme. This simple model achieves a 70.8% top-1 accuracy on the validation dataset.

Front-end Visualization. CNN EXPLAINER loads the pre-trained Tiny VGG model and computes forward propagation results in real time in a user's web browser using *TensorFlow.js* [51]. These results are visualized using *D3.js* [6] throughout the multiple interactive views.

7 USAGE SCENARIOS

7.1 Beginner Learning Layer Connectivity

Janis is a virology researcher using CNNs in a current project. Through an online deep learning course she has a general understanding of the goals of applying CNNs, and some basic knowledge of different types of CNN layers, but she needs help filling in some gaps in knowledge. Interested in learning how a 3-dimensional input (RGB image) leads to a 1-dimensional output (vector of class probabilities) in a CNN, Janis begins exploring the architecture from the *Overview* (Fig. 5A).

After clicking the "Show detail" button, Janis notices that the `output` layer is a 1-dimensional tensor of size 10, while `max_pool_2`, the previous layer, is a 3-dimensional (13×13×10) tensor. Confused, she hovers over a neuron in the `output` layer to inspect connections between the final two layers of the architecture: the `max_pool_2` layer has 10 neurons; the `output` layer has 10 neurons each representing a class label, and the `output` layer is fully-connected to the `max_pool_2` layer. She clicks that `output` neuron, which transitions the *Overview* (Fig. 5A) to the *Flatten Elastic Explanation View* (Fig. 5B). She notices that edges between these two layers intersect a 1-dimensional flatten layer and pass through a softmax function. By hovering over pixels from the activation map, Janis understands how the 2-dimensional matrix is "unwrapped" to yield a portion of the 1-dimensional `flatten` layer. As she continues to follow the edge after the `flatten` layer, she clicks the softmax button which leads her to the *Softmax Interactive Formula View* (Fig. 5C). She learns how the outputs of the `flatten` layer are normalized by observing the equation linked with logits through animations. Janis recognizes that her previous coursework has not taught these "hidden" operations prior to the `output` layer, which flatten and normalize the output of the `max_pool_2` layer. Instead of searching through lecture videos and textbooks, CNN EXPLAINER enables Janis to learn these often-overlooked operations through a hierarchy of interactive views in a stand-alone website. She now feels more equipped to apply CNNs to her virology research.

7.2 Teaching Through Interactive Experimentation

A university professor, Damian, is currently teaching a computer vision class which covers CNNs. Damian begins his lecture with standard slides. After describing the theory of convolutions, he opens CNN EXPLAINER to demonstrate the convolution operation working inside a full CNN for image classification. With CNN EXPLAINER projected to the class, Damian transitions from the *Overview* (Fig. 1A) to the *Convolutional Elastic Explanation View* (Fig. 1B). Damian encourages the class to interpret the sliding window animation (Fig. 2B) as it generates several intermediate results. He then asks the class to predict kernel weights in a specific neuron. To test student's hypotheses, Damian enters the *Convolutional Interactive Formula View* (Fig. 1C), to display the convolution operation with the true kernel weights. In this view, he can hover over the input and output matrices to answer questions from the class, and display computations behind the operation.

Recalled from theory, a student asks a question regarding the impact of altering the stride hyperparameter on the animated sliding window in convolutional layers. To illustrate the impact of alternative hyperparameters, Damian scrolls down to the "Convolutional Layer" section of the complementary article, where he experiments by adjusting stride and other hyperparameters with the *Hyperparameter Widget* (Fig. 7) in front of the class. CNN EXPLAINER is the first software that allows Damian to explain convolutional operations and hyperparameters with real image inputs, and quickly answer students' questions in class. Previously, Damian had to draw illustrations with simple matrix inputs on slides or a chalkboard. Finally, to reinforce the concepts and encourage individual experimentation, Damian provides the class with a URL to the web-based CNN EXPLAINER for students to return to in the future.

8 OBSERVATIONAL STUDY

We conducted an observational study to investigate how CNN EXPLAINER's target users (e.g., aspiring deep learning students) would use this tool to learn about CNNs, and also to test the tool's usability.

8.1 Participants

CNN EXPLAINER is designed for deep learning beginners who are interested in learning CNNs. In this study, we aimed to recruit participants who aspire to learn about CNNs and have some knowledge of basic machine learning concepts (e.g., knowing what an image classifier is). We recruited 16 student participants from a large university (4 female, 12 male) through internal mailing lists (e.g., machine learning and computer science Ph.D., M.S., and undergraduate students). Seven participants were Ph.D. students, seven were M.S. students, and the other two were undergraduates. All participants were interested in learning CNNs, and none of them had known CNN EXPLAINER before. Participants self-reported their level of knowledge on non-neural network machine learning techniques, with an average score of 3.26 on a scale of 0 to 5 (0 being "no knowledge" and 5 being "expert"); and an average score of 2.06 on CNNs (on the same scale). No participant self-reported a score of 5 for their knowledge on CNNs, and one participant had a score of 0. To help better organize our discussion, we refer to participants with CNN knowledge score of 0, 1 or 2 as B1-B11, where

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVCG.2020.3030418, IEEE Transactions on Visualization and Computer Graphics

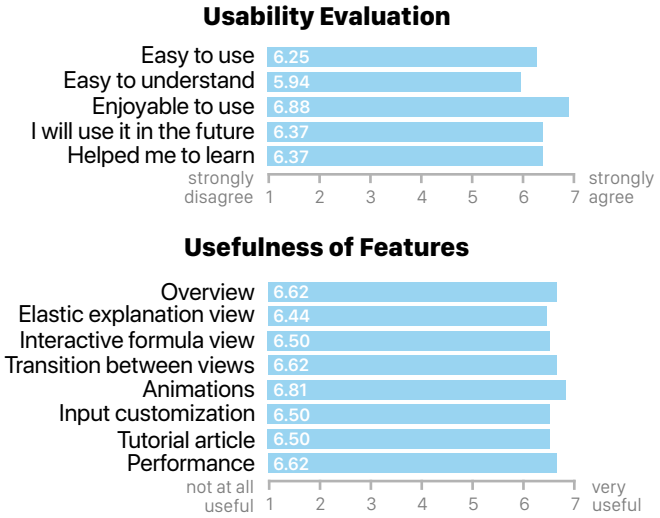


Fig. 8. Average ratings from 16 participants regarding the usability and usefulness of CNN EXPLAINER. **Top:** Participants thought CNN EXPLAINER was easy to use, enjoyable, and helped them learn about CNNs. **Bottom:** All features, especially animations, were rated favorably.

"B" stands for "Beginner"; and those with score of 3 or 4 as K1-K5, where "K" stands for "Knowledgeable."

8.2 Procedure

We conducted this study with participants one-on-one via video-conferencing software. With the permission of all participants, we recorded the participants' audio and computer screen for subsequent analysis. After participants signed consent forms, we provided them a 5-minute overview of CNNs, followed by a 3-minute tutorial of CNN EXPLAINER. Participants then freely explored our tool in their computer's web browser. We also provided a feature checklist, which outlined the main features of our tool and encouraged participants to try as many features as they could. During the study, participants were asked to think aloud and share their computer screen with us; they were encouraged to ask questions when necessary. Each session ended with a usability questionnaire coupled with an exit interview that asked participants about their process of using CNN EXPLAINER, and if this tool could be helpful for them. Each study lasted around 50 minutes, and we compensated each participant with a $10 Amazon Gift card.

8.3 Results and Design Lessons

The exit questionnaire included a series of 7-point Likert-scale questions about the utility and usefulness of different views in CNN EXPLAINER (Fig. 8). All average Likert rating were above 6 except the rating of "easy to understand". From the high ratings and our observations, participants found our tool easy to use and understand, retained a high engagement level during their session, and eventually gained a better understanding of CNN concepts. Our observations also reflect key findings in previous AV research [13, 32]. This section describes design lessons and limitations of our tool distilled from this study.

8.3.1 Transitions between different views

Transitions help users link CNN operations and structures. Several participants (9/16) commented that they liked how our tool transitions between high-level CNN structure views and low-level mathematical explanations. It helps them better understand the interplay between layer computations and the overall CNN data transformation—one of the key challenges for understanding CNN concepts, as we identified from our instructor interviews and our student survey. For example, initially K4 was confused to see the *Convolutional Elastic Explanation View*, but after reading the annotation text, he remarked, "*Oh, I understand what an intermediate layer is now—you run the convolution on the image, then you add all those results to get this.*" After exploring the *Convolutional Interactive Formula View*, he immediately noted, "*Every single aspect of the convolution layer is shown here. [This] is super helpful.*" Similarly, B5 commented, "*Good to see the big picture at once and the transition to different views [...] I like that I can hide details of a unit in a compact way and expand it when [needed].*"

CNN EXPLAINER employs the fisheye view technique for presenting the *Elastic Explanation Views* (Fig. 1B, Fig. 5B): after transitioning from the *Overview* to a specific layer, neighboring layers are still shown while further layers (lower degree-of-interest) have lower opacity. Participants found this transition design helpful for them to learn layer-specific details while having CNN structural context in the background. For instance, K5 said "*I can focus on the current layer but still know the same operation goes on for other layers.*" Our observations from this study suggest that our fluid transition design between different level of abstraction can help users to better connect unfamiliar layer mechanisms to the complex model structure.

8.3.2 Animations for enjoyable learning experience

Another favorite feature of CNN EXPLAINER that participants mentioned was the use of animations, which received the highest rating in the exit questionnaire (Fig. 8). In our tool, animations serve two purposes: to assimilate the relationship between different visual components and to help illustrate the model's underlying operations.

Transition animations help navigating. Layer movement is animated during view transitions. We noticed it helped participants to be aware of different views, and all participants navigated through the views naturally. In addition to assisting with understanding the relationship between distinct views, animation also helped them discover the linking between different visualization elements. For example, B8 quickly found that the logit circle is linked to its corresponding value in the formula, when she saw the circle-number pair appear one-by-one with animation in the *Softmax Interactive Formula View* (Fig. 5C).

Algorithm animations contribute to understanding. Animations that simulate the model's inner-workings helped participants learn underlying operations by validating their hypotheses. In the *Convolutional Elastic Explanation View* (Fig. 2B), we animate a small rectangle sliding through one matrix to mimic the CNN's internal sliding window. We noticed many participants had their attention drawn to this animation when they first transitioned into the *Convolutional Elastic Explanation View*. However, they did not report that they understood the convolution operation until interacting with other features, such as reading the annotation text or transitioning to the *Convolutional Interactive Formula View* (Fig. 2C). Some participants went back to watch the animation multiple times and commented that it made sense, for example, K5 said "*Very helpful to see how the image builds as the window slides through,*" but others, such as B9 remarked, "*It is not easy to understand [convolution] using only animation.*" Therefore, we hypothesize that this animation can indirectly help users to learn about the convolution algorithm by validating their newly formed mental models of how specific operation behave. To test this hypothesis, a rigorous controlled experiment would be needed. Related research work on the effect of animation in computer science education also found that algorithm animation does not automatically improve learning, but it may lead learners to make predictions of the algorithm behavior which in turn helps learning [8].

Animations improve learning engagement and enjoyment. We found animations helped to increase participants' engagement level (e.g., spending more time and effort) and made CNN EXPLAINER more enjoyable to use. In the study, many participants repeatedly played and viewed different animations. For example, K2 replayed the window sliding animation multiple times: "*The is very well-animated [...] I always love smooth animations.*" B7 also attributed animations to his enjoyable experience with our tool: "*[The tool is] enjoyable to use [...] I especially like the lovely animation.*"

8.3.3 Engaging learning through visualization customization

CNN EXPLAINER allows users to modify the visualization. For example, users can change the input image or upload their own image for

classification; CNN EXPLAINER visualizes the new prediction with the new activation maps in every layer. Similarly, users can interactively explore how hyperparameters affect the convolution operation (Fig. 7).

Customization enables hypothesis testing. Many participants used visualization customization to test their predictions of model behaviors. For example, through inspecting the input layer in the *Overview*, B4 learned that the input layer comprised multiple different image channels (e.g., red, green, and blue). He changed the input image to a red bell pepper from Tiny Imagenet (shown on the right) and expected to see high values in the input red channel: "*If I click the red image, I would see...*" After the updated visualization showed what he predicted, he said "*Right, it makes sense.*" We found the *Hyperparameter Widget* also allowed participants to test their hypotheses. While reading the description of convolution hyperparameters in the tutorial article, K3 noted "*Wait, then sometimes they won't work*". He then modified the hyperparatmeters in the *Hyperparameter Widget* and noticed some combinations indeed did not yield a valid operation output: "*It won't be able to slide, because the stride and kernel size don't fit the matrix*".

Customization facilitates engagement. Participants were intrigued to modify the visualization, and their engagement sparked further interest in learning CNNs. In the study, B6 spent a large amount of time on testing the CNN's behavior on edge cases by finding "difficult" images online. He searched with keywords "koala", "koala in a car", "bell pepper pizza", and eventually found a bell pepper pizza photo (shown on the right[3]). Our CNN model predicted the image as bell pepper with a probability of 0.71 and ladybug with a probability of 0.2. He commented, "*The model is not robust [...] oh, the ladybug ['s high softmax score] might come from the red dot.*" Another participant B5 uploaded his own photo as a new input image for the CNN model. After seeing his picture being classified as espresso, B5 started to use our tool to explore the reason of such classification by tracking back activation maps. He also asked how do experts interpret CNNs and said he would be interested in learning more about deep learning interpretability. This observation reflects previous findings that customizable visualization makes learning more engaging [13, 42].

8.3.4 Limitations

While we found CNN EXPLAINER provided participants with an engaging and enjoyable learning experience and helped them to more easily learn about CNNs, we also noticed some potential improvements to our current system design from this study.

Beginners need more guidance. We found that participants with less knowledge of CNNs needed more instructions to begin using CNN EXPLAINER. Some participants reported that the visual representation of the CNN and animation initially were not easy to understand, but the tutorial article and text annotations greatly helped them to interpret the visualizations. B8 skimmed through the tutorial article before interacting with the main visualization. She said, "*After going through the article, I think I will be able to use the tool better [...] I think the article is good, for beginner users especially.*" B2 appreciated the ability to jump to a certain section in the article by clicking the layer name in the visualization, and he suggested us to "*include a step-by-step tutorial for first time users [...] There was too much information, and I didn't know where to click at the beginning*". Therefore, we believe adding more text annotation and having a step-by-step tutorial mode could help beginners better understand the relations between CNN operations and their visual representations.

Limited explanation of *why* CNN works. Some participants, especially those less experienced with CNNs, were interested in learning *why* the CNN architecture works in addition to learning *how* a CNN model makes predictions. For example, B7 asked "*Why do we need ReLU?*" when he was learning the formula of the ReLU function. B5 understood what a Max Pooling layer's operation does but was unclear why it contributes to CNN's performance: "*It is counter-intuitive that Max Pooling reduces the [representation] size but makes the model better.*" Similarly, B6 commented on the Max Pooling layer: "*Why not take the minimum value? [...] I know how to compute them [layers], but I don't know why we compute them.*" Even though it is still an open question why CNNs work so well for various applications [17, 59], there are some commonly accepted "intuitions" of how different layers help this model class succeed. We briefly explain them in the tutorial article: for example, ReLU function is used to introduce non-linearty in the model. However, we believe it is worth designing visualizations that help users to learn about these concepts. For example, allowing users to change the ReLU activation function to a linear function, and then visualizing the new model predictions may help users gain understanding of *why* non-linear activation functions are needed in CNNs.

[3]Photo by Jennifer Laughlin, used with permission.

9 DISCUSSION AND FUTURE WORK

Explaining training process and backpropagation. CNN EXPLAINER helps users to learn how a pre-trained CNN model transforms the input image data into a class prediction. As we identified from two preliminary studies and an observational study, students are also interested in learning about the training process and backpropagation of CNNs. We plan to work with instructors and students to design and develop new visualizations to help beginners gain understanding of the training process and backpropagation in detail.

Generalizing to other layer types and neural network models. Our observational study demonstrated that CNN EXPLAINER helps users more easily understand low-level layer operations, high-level model structure, and their connections. We can adapt the *Interactive Formula Views* to explain other layer types (e.g., Leaky ReLU [38]) or a combination of layers (e.g. Residual Block [21]). Similarly, the transition between different levels of abstraction can be generalized to other neural networks, such as long short-term memory networks [22] and Transformer models [55] that require learners to understand the intricate layer operations in the context of a complex network structure.

Integrating algorithm visualization best practices. Existing work has studied how to design effective visualizations to help students learn algorithms. CNN EXPLAINER applies two key design principles from AV—visualizations with explanations and customizable visualizations (**G4**). However, there are many other AV design practices that future researchers can integrate in educational deep learning tools, such as giving interactive "pop quizzes" during the visualization process [41] and encouraging users to build their own visualizations [52].

Quantitative evaluation of educational effectiveness. We conducted a qualitative observational study to evaluate the usefulness and usability of CNN EXPLAINER. Further quantitative user studies would help us investigate how visualization tools help users gain understanding of deep learning concepts. We will draw inspiration from recent research [11, 28] to assess users' engagement level and content understanding through analysis of interaction logs.

10 CONCLUSION

As deep learning is increasingly used throughout our everyday life, it is important to help learners take the first step toward understanding this promising yet complex technology. In this work, we present CNN EXPLAINER, an interactive visualization system designed for non-experts to more easily learn about CNNs. Our tool runs in modern web browsers and is open-sourced, broadening the public's education access to modern AI techniques. We discussed design lessons learned from our iterative design process and an observational user study. We hope our work will inspire further research and development of visualization tools that help democratize and lower the barrier to understanding and appropriately applying AI technologies.

ACKNOWLEDGMENTS

We thank Anmol Chhabria, Kaan Sancak, Kantwon Rogers, and the Georgia Tech Visualization Lab for their support and constructive feedback. This work was supported in part by NSF grants IIS-1563816, CNS-1704701, NASA NSTRF, DARPA GARD, gifts from Intel, NVIDIA, Google, Amazon. Use, duplication, or disclosure is subject to the restrictions as stated in Agreement number HR00112030001 between the Government and the Performer.

REFERENCES

[1] Tiny ImageNet Visual Recognition Challenge. https://tiny-imagenet.herokuapp.com, 2015.

[2] Backpropagation Algorithm. https://developers-dot-devsite-v2-prod.appspot.com/machine-learning/crash-course/backprop-scroll, 2018.

[3] TensorSpace.js: Neural Network 3D Visualization Framework. https://tensorspace.org, 2018.

[4] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 265–283. Savannah, GA, USA, Nov. 2016.

[5] A. Bilal, A. Jourabloo, M. Ye, X. Liu, and L. Ren. Do Convolutional Neural Networks Learn Class Hierarchy? *IEEE Transactions on Visualization and Computer Graphics*, 24(1):152–162, Jan. 2018.

[6] M. Bostock, V. Ogievetsky, and J. Heer. D^3 Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, Dec. 2011.

[7] M. H. Brown. *Algorithm Animation*. MIT Press, Cambridge, MA, USA, 1988.

[8] M. D. Byrne, R. Catrambone, and J. T. Stasko. Evaluating animations as student aids in learning computer algorithms. *Computers & Education*, 33(4):253–278, Dec. 1999.

[9] M. Carney, B. Webster, I. Alvarado, K. Phillips, N. Howell, J. Griffith, J. Jongejan, A. Pitaru, and A. Chen. Teachable Machine: Approachable Web-Based Tool for Exploring Machine Learning Classification. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, Honolulu, HI, USA, 2020.

[10] S. Carter and M. Nielsen. Using Artificial Intelligence to Augment Human Intelligence. *Distill*, 2(12), Dec. 2017.

[11] M. Conlen, A. Kale, and J. Heer. Capture & Analysis of Active Reading Behaviors for Interactive Articles on the Web. *Computer Graphics Forum*, 38(3):687–698, June 2019.

[12] N. Das, H. Park, Z. J. Wang, F. Hohman, R. Firstman, E. Rogers, and D. H. Chau. Massif: Interactive interpretation of adversarial attacks on deep learning. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, Honolulu, HI, USA, 2020.

[13] E. Fouh, M. Akbar, and C. A. Shaffer. The Role of Visualization in Computer Science Education. *Computers in the Schools*, 29(1-2):95–117, Jan. 2012.

[14] D. Galles. Data structure visualizations, 2006.

[15] R. Garcia, A. C. Telea, B. Castro da Silva, J. Tørresen, and J. L. Dihl Comba. A task-and-technique centered survey on visual analytics for deep learning model engineering. *Computers & Graphics*, 77:30–49, Dec. 2018.

[16] S. Grissom, M. F. McNally, and T. Naps. Algorithm visualization in CS education: Comparing levels of student engagement. In *Proceedings of the 2003 ACM Symposium on Software Visualization - SoftVis '03*, pp. 87–94. San Diego, CA, USA, 2003.

[17] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, May 2018.

[18] P. J. Guo. Online python tutor: Embeddable web-based program visualization for cs education. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education - SIGCSE '13*, pp. 579–584. ACM Press, Denver, CO, USA, 2013.

[19] S. Hansen, N. Narayanan, and M. Hegarty. Designing Educationally Effective Algorithm Visualizations. *Journal of Visual Languages & Computing*, 13(3):291–317, June 2002.

[20] A. W. Harley. An Interactive Node-Link Visualization of Convolutional Neural Networks. In *Advances in Visual Computing*, vol. 9474, pp. 867–877. Springer International Publishing, 2015.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, Dec. 2015.

[22] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, Nov. 1997.

[23] F. Hohman, M. Kahng, R. Pienta, and D. H. Chau. Visual Analytics in Deep Learning: An Interrogative Survey for the Next Frontiers. *IEEE Transactions on Visualization and Computer Graphics*, 25(8):2674–2693, Aug. 2019.

[24] F. Hohman, H. Park, C. Robinson, and D. H. Chau. Summit: Scaling Deep Learning Interpretability by Visualizing Activation and Attribution Summarizations. *IEEE Transactions on Visualization and Computer Graphics*, 26(1):1096–1106, Jan. 2020.

[25] C. Hundhausen and S. Douglas. Using visualizations to learn algorithms: Should students construct their own, or view an expert's? In *Proceeding 2000 IEEE International Symposium on Visual Languages*, pp. 21–28. IEEE Comput. Soc, Seattle, WA, USA, 2000.

[26] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko. A Meta-Study of Algorithm Visualization Effectiveness. *Journal of Visual Languages & Computing*, 13(3):259–290, June 2002.

[27] M. Kahng, P. Y. Andrews, A. Kalro, and D. H. Chau. ActiVis: Visual Exploration of Industry-Scale Deep Neural Network Models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):88–97, Jan. 2018.

[28] M. Kahng and D. H. Chau. How Does Visualization Help People Learn Deep Learning? Evaluation of GAN Lab. In *IEEE VIS 2019 Workshop on EValuation of Interactive VisuAl Machine Learning Systems*, Oct. 2019.

[29] M. Kahng, N. Thorat, D. H. Chau, F. B. Viegas, and M. Wattenberg. GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):310–320, Jan. 2019.

[30] A. Karpathy. ConvNetJS MNIST demo, 2016.

[31] A. Karpathy. CS231n Convolutional Neural Networks for Visual Recognition, 2016.

[32] C. Kehoe, J. Stasko, and A. Taylor. Rethinking the evaluation of algorithm animations as learning aids: An observational study. *International Journal of Human-Computer Studies*, 54(2):265–284, Feb. 2001.

[33] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, May 2015.

[34] M. Liu, S. Liu, H. Su, K. Cao, and J. Zhu. Analyzing the Noise Robustness of Deep Neural Networks. In *2018 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pp. 60–71. IEEE, Berlin, Germany, Oct. 2018.

[35] M. Liu, J. Shi, K. Cao, J. Zhu, and S. Liu. Analyzing the Training Processes of Deep Generative Models. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):77–87, Jan. 2018.

[36] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu. Towards Better Analysis of Deep Convolutional Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, Jan. 2017.

[37] S. Liu, D. Maljovec, B. Wang, P.-T. Bremer, and V. Pascucci. Visualizing High-Dimensional Data: Advances in the Past Decade. *IEEE Transactions on Visualization and Computer Graphics*, 23(3):1249–1268, Mar. 2017.

[38] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[39] A. Madsen. Visualizing memorization in RNNs. *Distill*, 4(3):10.23915/distill.00016, Mar. 2019.

[40] R. E. Mayer and R. B. Anderson. Animations need narrations: An experimental test of a dual-coding hypothesis. *Journal of Educational Psychology*, 83(4):484–490, 1991.

[41] T. L. Naps, J. R. Eagan, and L. L. Norton. JHAVÉ—an environment to actively engage students in Web-based algorithm visualizations. *ACM SIGCSE Bulletin*, 32(1):109–113, Mar. 2000.

[42] T. L. Naps, G. Rößling, V. Almstrum, W. Dann, R. Fleischer, C. Hundhausen, A. Korhonen, L. Malmi, M. McNally, S. Rodger, and J. Á. Velázquez-Iturbide. Exploring the Role of Visualization and Engagement in Computer Science Education. *SIGCSE Bull.*, 35(2):131–152, June 2002.

[43] A. P. Norton and Y. Qi. Adversarial-Playground: A visualization suite showing how adversarial examples fool deep learning. In *2017 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pp. 1–4. IEEE, Phoenix, AZ, USA, Oct. 2017.

[44] C. Olah. Neural Networks, Manifolds, and Topology, June 2014.

[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pp. 8024–8035. 2019.

[46] 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.

[47] D. Schweitzer and W. Brown. Interactive visualization for the active learning classroom. *ACM SIGCSE Bulletin*, 39(1):208, Mar. 2007.

[48] C. A. Shaffer, M. L. Cooper, A. J. D. Alon, M. Akbar, M. Stewart, S. Ponce, and S. H. Edwards. Algorithm Visualization: The State of the Field. *ACM Transactions on Computing Education*, 10(3):1–22, Aug. 2010.

[49] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv:1409.1556 [cs]*, Apr. 2015.

[50] D. Smilkov, S. Carter, D. Sculley, F. B. Viégas, and M. Wattenberg. Direct-Manipulation Visualization of Deep Networks. *arXiv:1708.03788*, Aug. 2017.

[51] D. Smilkov, N. Thorat, Y. Assogba, A. Yuan, N. Kreeger, P. Yu, K. Zhang, S. Cai, E. Nielsen, D. Soergel, S. Bileschi, M. Terry, C. Nicholson, S. N. Gupta, S. Sirajuddin, D. Sculley, R. Monga, G. Corrado, F. B. Viégas, and M. Wattenberg. TensorFlow.js: Machine Learning for the Web and Beyond. *arXiv:1901.05350 [cs]*, Feb. 2019.

[52] J. T. Stasko. Using student-built algorithm animations as learning aids. *ACM SIGCSE Bulletin*, 29(1):25–29, Mar. 1997.

[53] E. Stevens, L. Antiga, and T. Viehmann. *Deep Learning with PyTorch*. O'Reilly Media, 2019.

[54] H. Strobelt, S. Gehrmann, H. Pfister, and A. M. Rush. LSTMVis: A Tool for Visual Analysis of Hidden State Dynamics in Recurrent Neural Networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):667–676, Jan. 2018.

[55] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 6000–6010, 2017.

[56] J. Wang, L. Gou, H. Yang, and H.-W. Shen. GANViz: A Visual Analytics Approach to Understand the Adversarial Game. *IEEE Transactions on Visualization and Computer Graphics*, 24(6):1905–1917, June 2018.

[57] Z. J. Wang, R. Turko, O. Shaikh, H. Park, N. Das, F. Hohman, M. Kahng, and D. H. Chau. CNN 101: Interactive visual learning for convolutional neural networks. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20. ACM, Honolulu, HI, USA, 2020.

[58] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding Neural Networks Through Deep Visualization. In *ICML Deep Learning Workshop*, 2015.

[59] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations (ICLR), Toulon, France, Conference Track Proceedings*, 2017.