VATUN: Visual Analytics for Testing and Understanding Convolutional Neural Networks

Cheonbok Park[1]*, Soyoung Yang[2]*, Inyoup Na[3]*, Sunghyo Chung[4], Sungbok Shin[5], Bum Chul Kwon[6], Deokgun Park[7], and Jaegul Choo[2]

[1] NAVER Corp, [2] KAIST, [3] Skelter Labs, [4] Kakao Corp, [5] University of Maryland, [6] IBM Research, [7] University of Texas at Arlington,
* Equal Contribution

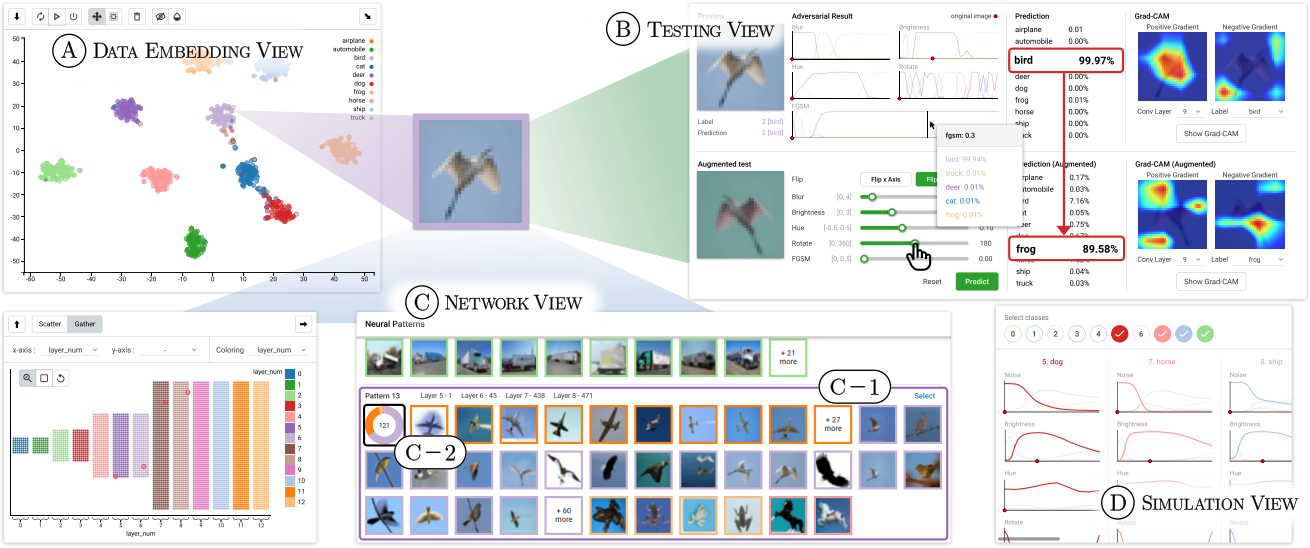

Figure 1: *Overview of VATUN. (A) Data embedding view displays the t-SNE results of the dataset and allows the users to directly access their interesting instance(s). A selected image from this view is shown in the testing view. (B) Testing view reveals the prediction of the trained model by getting different image conditions from the users and highlights evidential parts of the model via Grad-CAM. If the user changes the hue and rotation of the selected image, the predicted class changes from 'Bird' to 'Frog' and the Grad-CAM shows the model fails to capture the body of the bird. (C) Network view shows the network using unit visualization and presents relevant neural patterns based on either an selected image(s) in (B) or filter(s) in (C). (C-1) Neural pattern reveals that the model learned pattern 13 as a feature of 'Sky' or blue background. (C-2) Pie graph allows users to know that this feature is shared with three classes, especially for the 'Airplane' and 'Bird' ones. (D) Simulation view shows how sensitive the model is under different conditions (e.g., rotation, noise, etc.) overall classes.*

Abstract

Convolutional neural networks (CNNs) are popularly used in a wide range of applications, such as computer vision, natural language processing, and human-computer interaction. However, testing and understanding a trained model is difficult and very time-consuming. This is because their inner mechanisms are often considered as a 'black-box' due to difficulty in understanding the causal relationships between processes and results. To help the testing and understanding of such models, we present a user-interactive visual analytics system, VATUN, to analyze a CNN-based image classification model. Users can accomplish the following four tasks in our integrated system: (1) detect data instances in which the model confuses classification, (2) compare outcomes of the model by manipulating the conditions of the image, (3) understand reasons for the prediction of the model by highlighting highly influential parts from the image, and (4) analyze the overall what-if scenarios when augmenting the instances for each class. Moreover, by combining multiple techniques, our system lets users analyze behavior of the model from various perspectives. We conduct a user study of an image classification scenario with three domain experts. Our study will contribute to reducing the time cost for testing and understanding the CNN-based models in several industrial areas.

CCS Concepts

• ***Human-centered computing*** → *Visualization toolkits;*

DOI: 10.2312/evs.20211047

1. Introduction

Recent advances in deep learning techniques have accomplished major breakthroughs in various machine learning and artificial intelligence tasks, including computer vision [KSH12], speech recognition [HDY*12], and natural language processing [BCB15, CW08]. As a result, deep neural networks (DNNs) are widely used in various industries, such as automobile and healthcare systems. Inter alia, convolutional neural networks (CNNs) are utilized in a variety of tasks, ranging from handwritten digit recognition to self-driving cars and automated diagnostic systems. As CNNs become critical to our daily lives, the trained model should show consistent performances in diverse circumstances and reveal its reasoning process. For this, machine learning engineers experiment with the model and check whether the model is consistent and explainable. However, as the architecture of the model is complex and the test dataset is limited, understanding and testing the model becomes more and more challenging.

To alleviate engineers' efforts in the testing and understanding of such CNN-based models and the limited test data, we present **VATUN**, a visual analytic for testing and understanding convolutional neural networks. Following user-centered design practice, we interviewed machine learning researchers working in industries about their experiences of testing the model for deployment. Then, we built a system reflecting the feedback they provided. VATUN consists of four main visual components: (1) a data embedding view that emphasizes the ambiguous data instances which the model has wrongly classified (Fig. 1 (A)), (2) a testing view that provides user-driven what-if exploration (Fig. 1 (B)), (3) a network view that visualizes the semantic features learned by the model and helps to understand the decision-making processes of the model (Fig. 1 (C)), and (4) a simulation view that summarizes the what-if scenarios of how the prediction scores for each class change with the image augmentation scenarios (Fig. 1 (D)). To supplement the limits of every single view, the components should be interconnected with each other. If a user explores an image and fails to find the reason for its misclassification, she or he can verify the rationale by utilizing neural patterns in the network view. Hence, VATUN, which integrates multi-aspect analyses to test and understand the model, helps users to investigate the model systematically on the levels of both data instances and model architecture.

2. Related Work

To design a visual analytic tool for testing and understanding CNN-based deep neural network models, we studied a wide range of research based on two perspectives: understanding CNN models and testing DNN models. Since the network includes a nonlinear structure and becomes more complex than before, understanding DNNs requires significant efforts by engineers. To alleviate the labor, an instance-based approach visualizes the CNNs via the data instances, showing how the data instance activates the model and explaining the model via the data instances [SGPR18, RFFT16, PHG*18, SGB*19, LSC*18, MCZ*17, SSC*16, KCK*19, PNJ*19, YLL*20, CBN*20]. In another direction, a network-based approach [LSL*17, WSW*18, LLS*18] visualizes the overall network as a graph structure and aggregates the internal values in the nodes and edges of the model.

Regarding the aspects of evaluating machine learning systems, suboptimal behaviors against adversarial data are ascribed to the fact that the model is overfitting, underfitting, or biased toward training data [ZBH*17]. Numerous studies attempt to tackle the issue by designing a system to test augmented images with such changes as brightness, occlusion, and rotation [PCYJ17, TPJR18]. Also, to alleviate the bias problem, a number of tools and studies have been developed, including API [Ban18], interface [YSA*18], What-if tool [Goo18], AI Fairness 360 [IBM18], and FairVis [CEH*19], to aid a fair machine learning decision.

However, most of the previous studies are focused on single aspects such as testing or understanding the model. Therefore, our system considers how to collaborate both approaches of testing and understanding the CNNs in a single system.

3. Design Requirements

For more than 12 months, we had regular monthly meetings with two machine learning researchers who have worked in computer vision applications in relevant industries. Based on these interviews, we designed our system and recieved iterative feedback on it. The goal of this study is to create a coordinated view for testing and understanding convolutional network models.

Requirement R1. Test the model via directly handling image instances. It is time-consuming to create or collect the test dataset that the users want to investigate. To alleviate the limitations of the test dataset, our system needs to allow the users to manipulate the image(s) and follow the what-if scenario [SGB*19].

Requirement R2. Visualize evidential parts of image(s). For users to understand the decision process of the model easily, our system should show the parts of the given image that the model concentrates on for classifications.

Requirement R3. Visualize logical process of model. For users to inspect the prediction of the model, our system should disclose the reasoning process by revealing the features the model learned.

Requirement R4. Validate model bias based on multi-aspect approaches. As new information is more difficult to figure out in separate views than when it is combined into a single view, our system aims to integrate all validations steps and make them sync.

4. Design of VATUN

VATUN is a visualization system that allows users to test a CNN-based model easily. VATUN consists of four views: data embedding view, testing view, network view, and simulation view.

4.1. Data Embedding View

In this view, the users can directly access a ambiguously classified image(s) and click the instance(s) to investigate in the testing view (Fig. 1 (B)) and network view (Fig. 1 (C)) **(R4)**. Considering the exploration approach [RFFT16], we aim to help users not only identify the ambiguous data instance(s) directly but also understand how the model learned the relationship between the images. Therefore, the scatterplot presents a dimension-reduced projection of hidden representations in the CNNs just before the last layer. When the

Figure 2: *Testing corner-case data. The explanation view shows that the head part of a dog image is important to classify this image. Concealing the head part leads to inaccurate prediction, 'Bird.'*

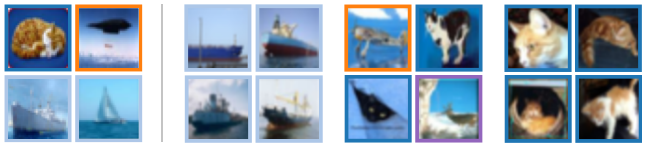

Figure 3: *Comparison of data instances with and without neural pattern recommendation. (a) depicts one-filter strategy, showing the most highly activated images when the users click a filter. (b), (c), and (d) are extracted by our neural pattern recommendation.*

"confidence score" button is clicked on, the dots shows the softmax probability for the predicted class p_c as our confidence score. This confidence score helps users identify ambiguous instances **(R1)**.

4.2. Testing View

To evaluate the model qualitatively, we provide two testing tools and an understanding tool as follows **(R1, R2)**. The image can be taken from the embedding view **(R4)**.

Data augmentation by changing image conditions (R1). Users can select a data instance from the embedding view or upload a test image, as shown in Fig. 1 (B), and apply image augmentations on the selected image. The image can be transformed by adjusting the hue, brightness, rotation, vertical flipping, horizontal flipping, blurring, and removing a part of an image's region. After the augmentation, the user can see how the model alters its prediction score for the augmented image. Furthermore, our system provides an adversarial attack based on a fast gradient sign method (FGSM) [GSS15].

Statistical result on instance level (R1). It is hard to collect and test all possible cases of image transformation. Therefore, the augmented result charts automatically summarize how the prediction changes according changes in the image, such as blurring, brightness, hue, or rotation, or an adversarial attack. Based on the simulation result chart, as shown in Fig. 1 (B), the users can verify that the trained model is vulnerable to rotation but robust to FGSM.

Visually explain the images via Grad-CAM (R2). We use a machine learning technique called Grad-CAM [SDV*16] to provide a visual explanation of how the model classifies the given images. Users can change the conditions of the image and compare the model explanations for the original image and the augmented one via Grad-CAM, where these two steps can be done interactively, which is similar to the what-if scenario [Goo18]. As shown

(a) **(b)**

Figure 4: *Robustness to rotation. In both (a) and (b), the graph on the left are rotation augmentation graph, where the x-axis is range of rotation and the y-axis is confidence score. The images on the right are from train dataset. 'Airplane' class (a) is robust to rotations, while 'Horse' (b) is not.*

in Fig. 2, users can define the conditions of Grad-CAM: an input image, a layer of the model followed by the "Conv Layer" caption, and a class followed by the "Label" caption.

4.3. Network View with Neural Pattern Recommendation

Based on the network view, users can test which nodes are activated and understand the decision-making process of the model by crossing the multiple views **(R3)**. The users can select the input data instance in either the embedding or the testing view **(R4)**.

Network visualization. We visualize all 4,224 filters in the 13 convolution layers in the CNN model by unit visualization [PKE17]. Each dot in the unit visualization represents filters, and its color shows the layer number where the filter belongs to.

Neural pattern recommendation. Extracting features from one filter is a general method, however, it results in inconsistent features that are hard to understand (Fig. 3 (a)). Therefore, our system extracts the features via the sequences of activated nodes across the layers of the model (Fig. 3 (b), (c), (d)). Before the user interaction, our system collects neural patterns in two steps. First, for each image input, we extract indices of the node or filter that has the highest activation score at each convolutional or fully connected layer and we make a sequence of those node indices for each image. Second, we collect sub-sequences that frequently appear in the set of sequences. If a sub-sequence appears in more than four sequences of images, the sub-sequence and corresponding images are defined as a neural pattern.

After computing the neural patterns for every dataset, our system recommends the neural patterns that include the images or filters, which the user selected on another view, as shown in Fig. 1 (C). The neural pattern is a group of image instances that share the consistent patterns between multiple filters, so this view allows the users to understand the reasoning process of the model. In addition, as shown in Fig. 1 (C-2), we summarize the class information of each neural pattern as a pie graph. The pie graph helps users recognize whether the pattern is unique or general across the classes.

4.4. Simulation View

The simulation view (Fig. 1 (D)) shows statistically summarized graphs at the class level, where the lines represent the changes of the models' prediction score along with the augmentation degree **(R1)**. The x-axis of the chart is the magnitude of the image augmentation, the y-axis is the average softmax probability of each class, and all statistics are pre-computed. Based on the simulation

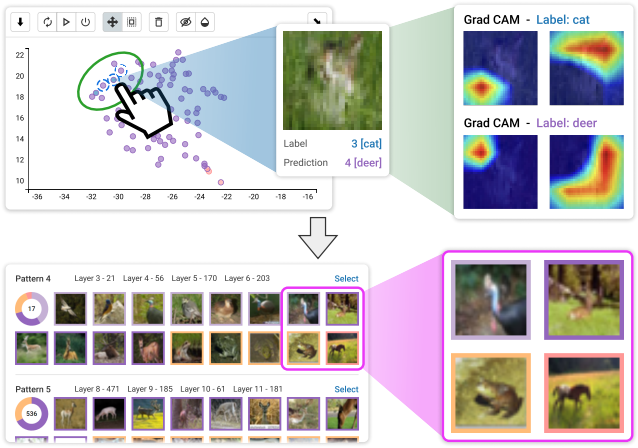

Figure 5: *Example of the complementary relationship between the testing view and the network view. Grad-CAM in testing view does not clearly explain the misclassification. However, the neural pattern recommendation explains that the model is biased to the background of grass or earth for the 'Deer' class.*

charts, the users can understand where the model is robust or weak. For instance, as shown in Fig. 4, the 'Airplane' class includes images with a diverse set of rotations. This allows the 'Airplane' class to maintain the first rank in all augmentation degrees of rotation.

5. Usage Scenarios

Our usage scenario demonstrates how VATUN can be useful in helping users evaluate CNN-based models. We invited three domain experts, where E1 and E2 are software engineers working at the software (model) testing team at a major company and E3 is a deep learning researcher working on video data with CNNs. First, we introduced the tool for 25 minutes and gave them 10 minutes to become familiar with the model and the system. For the next hour, participants freely used the system to explore the pre-trained model and to assess various cases by testing and understanding the model.

For our usage scenarios, we use the VGG-16 network [HZRS16, HLMW17], which is one of the most widely used CNNs. The VGG-16 model contains 13 convolutional layers. To validate our system, we utilize the CIFAR-10 dataset [KH09], a widely used image classification benchmark. With the CIFAR-10 dataset, the test accuracy of VGG-16 is 94.1%.

5.1. Model Testing

Find model bias via multiple views (R4). E1 noticed that an image in Fig. 5 was wrongly classified as 'Deer' instead of the true value of 'Cat,' and the image dot is located near the center of the 'Deer' class in the data embedding view. "In the testing view, I could not understand why the model classified the cat image into the deer class, but I guess it was due to the grass. So I move up to the network and neural pattern view. Via the neural pattern, I could finally understand the misclassification and verify that my hypothesis was correct." Interestingly, we found that the neural patterns for the images classified in the 'Deer' class have many features related to the grass and the earth.

Figure 6: *Neural patterns from selected image. Rights are neural patterns corresponding to the selected image on the left. These patterns mainly reveals 'Grass' and 'Horse' for each image group.*

Evaluate robustness of the model via testing view (R1, R2). Overall, experts were interested in the cases of failure in the classification task and concentrated on spots where clusters of the same classes overlapped. E3 found an image with the true label 'Ship,' but the model misclassified it as 'Airplane.' "From Grad-CAM in the testing view, I was able to see that the 'Sea' and 'Sky' part of each image contributed to classifying the images as 'Ship' and 'Airplane.' I speculated that when an airplane is with a sea background, the model misinterprets the airplane as a ship." Also, E3 adjusted the brightness of the image and found an image that was correctly classified in the 'Dog' class. E3 said, "The trained model would be reliable if the model properly classified such augmented images."

Neural patterns of selected data instances (R3). Most of the experts tried to analyze the features of the filters, which are heavily influenced the prediction. The experts clicked an image instance and explored the neural patterns corresponding to the selection. In images in Fig. 6, they found that the model captures the two features of the grass and horse's head. Prevalent features were in low-level layers, which diverse classes share, e.g., grass with car or dog. Besides, in high-level layers, the patterns become highly related to the class of the image, e.g., horse head. E2 said, "Owing to the neural pattern recommendation, we can explore features significant for classification and understand the characteristics of filters in CNNs."

6. Conclusion

In this paper, we propose VATUN, a novel visual analytic for testing and understanding convolutional neural networks by integrating visual components from data instances to network approaches. As our system has not been deployed with the empirical dataset, we plan to expand our model with diverse datasets, such as optical character recognition (OCR) and medical images. Especially in safety-critical domains, the evaluation process is significant and requires tremendous time due to the severity of the consequences. We anticipate that VATUN will alleviate the time problem and allow machine learning engineers and non-experts to easily diagnose biases and develop a robust model in the industry.

7. Acknowledgment

This work was partially supported by Institute of Information & Communications Technology Planning Evaluation (IITP) grant funded by the Korean government (MSIT) (No.2019-0- 00075, Artificial Intelligence Graduate School Program(KAIST)), and the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. NRF-2019R1A2C4070420).

References

[Ban18] BANTILAN N.: Themis-ml: A fairness-aware machine learning interface for end-to-end discrimination discovery and mitigation. *Journal of Technology in Human Services 36*, 1 (2018), 15–30. 2

[BCB15] BAHDANAU D., CHO K., BENGIO Y.: Neural machine translation by jointly learning to align and translate. In *Proc. the International Conference on Learning Representations (ICLR)* (2015). 2

[CBN*20] CHAN G. Y.-Y., BERTINI E., NONATO L. G., BARR B., SILVA C. T.: Melody: Generating and visualizing machine learning model summary to understand data and classifiers together. *arXiv preprint arXiv:2007.10614* (2020). 2

[CEH*19] CABRERA Á. A., EPPERSON W., HOHMAN F., KAHNG M., MORGENSTERN J., CHAU D. H.: FAIRVIS: visual analytics for discovering intersectional bias in machine learning. In *Proc. IEEE Conference on Visual Analytics Science and Technology (VAST)* (2019), pp. 46–56. doi:10.1109/VAST47406.2019.8986948. 2

[CW08] COLLOBERT R., WESTON J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In *Proc. the International Conference on Machine Learning (ICML)* (2008), vol. 307, pp. 160–167. doi:10.1145/1390156.1390177. 2

[Goo18] GOOGLE: What if tool. URL: https://pair-code.github.io/what-if-tool. 2, 3

[GSS15] GOODFELLOW I. J., SHLENS J., SZEGEDY C.: Explaining and harnessing adversarial examples. In *Proc. the International Conference on Learning Representations (ICLR)* (2015). 3

[HDY*12] HINTON G., DENG L., YU D., DAHL G. E., MOHAMED A., JAITLY N., SENIOR A., VANHOUCKE V., NGUYEN P., SAINATH T. N., KINGSBURY B.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine 29*, 6 (2012), 82–97. doi:10.1109/MSP.2012.2205597. 2

[HLMW17] HUANG G., LIU Z., MAATEN L. V. D., WEINBERGER K. Q.: Densely connected convolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 2261–2269. doi:10.1109/CVPR.2017.243. 4

[HZRS16] HE K., ZHANG X., REN S., SUN J.: Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778. doi:10.1109/CVPR.2016.90. 4

[IBM18] IBM: Fairness 360. URL: https://www.ibm.com/blogs/research/2018/09/ai-fairness-360/. 2

[KCK*19] KWON B. C., CHOI M., KIM J. T., CHOI E., KIM Y. B., KWON S., SUN J., CHOO J.: Retainvis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 299–309. doi:10.1109/TVCG.2018.2865027. 2

[KH09] KRIZHEVSKY A., HINTON G.: Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep 1* (2009). 4

[KSH12] KRIZHEVSKY A., SUTSKEVER I., HINTON G.: Imagenet classification with deep convolutional neural networks. doi:10.1145/3065386. 2

[LLS*18] LIU M., LIU S., SU H., CAO K., ZHU J.: Analyzing the noise robustness of deep neural networks. In *Proc. IEEE Conference on Visual Analytics Science and Technology (VAST)* (2018), pp. 60–71. doi:10.1109/VAST.2018.8802509. 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. doi:10.1109/TVCG.2017.2744938. 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. doi:10.1109/TVCG.2016.2598831. 2

[MCZ*17] MING Y., CAO S., ZHANG R., LI Z., CHEN Y., SONG Y., QU H.: Understanding hidden memories of recurrent neural networks. In *Proc. IEEE Conference on Visual Analytics Science and Technology (VAST)* (2017), pp. 13–24. doi:10.1109/VAST.2017.8585721. 2

[PCYJ17] PEI K., CAO Y., YANG J., JANA S.: Deepxplore: Automated whitebox testing of deep learning systems. In *Proc. the Symposium on Operating Systems Principles (SOSP)* (2017), pp. 1–18. doi:10.1145/3132747.3132785. 2

[PHG*18] PEZZOTTI N., HÖLLT T., GEMERT J. V., LELIEVELDT B. P. F., 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. doi:10.1109/TVCG.2017.2744358. 2

[PKE17] PARK D. G., KIM S., ELMQVIST N.: Gatherplots: Generalized scatterplots for nominal data. *CoRR abs/1708.08033* (2017). arXiv:1708.08033. 3

[PNJ*19] PARK C., NA I., JO Y., SHIN S., YOO J., KWON B. C., ZHAO J., NOH H., LEE Y., CHOO J.: Sanvis: Visual analytics for understanding self-attention networks. In *Proc. IEEE Visualization Conference (VIS)* (2019), pp. 146–150. doi:10.1109/VISUAL.2019.8933677. 2

[RFFT16] RAUBER P., FADEL S., FALCÃO A., TELEA A.: Visualizing the hidden activity of artificial neural networks. *IEEE Transactions on Visualization and Computer Graphics 23* (2016), 1–1. doi:10.1109/TVCG.2016.2598838. 2

[SDV*16] SELVARAJU R. R., DAS A., VEDANTAM R., COGSWELL M., PARIKH D., BATRA D.: Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR abs/1610.02391* (2016). arXiv:1610.02391. 3

[SGB*19] STROBELT H., GEHRMANN S., BEHRISCH M., PERER A., PFISTER H., RUSH A. M.: Seq2seq-vis: A visual debugging tool for sequence-to-sequence models. *IEEE Transactions on Visualization and Computer Graphics 25*, 1 (2019), 353–363. doi:10.1109/TVCG.2018.2865044. 2

[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. doi:10.1109/TVCG.2017.2744158. 2

[SSC*16] SUNGHYO C., SANGHO S., CHEONBOK P., KYEONGPIL K., JAEGUL C., BUM CHUL K.: Revacnn: Steering convolutional neural network via real-time visual analytics. *FILM at NIPS - Future of Interactive Learning Machines Workshop* (2016). 2

[TPJR18] TIAN Y., PEI K., JANA S., RAY B.: Deeptest: automated testing of deep-neural-network-driven autonomous cars. In *Proc. the International Conference on Software Engineering (ICSE)* (2018), pp. 303–314. doi:10.1145/3180155.3180220. 2

[WSW*18] WONGSUPHASAWAT K., SMILKOV D., WEXLER J., WILSON J., MANÉ D., FRITZ D., KRISHNAN D., VIÉGAS F. B., WATTENBERG M.: Visualizing dataflow graphs of deep learning models in tensorflow. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (2018), 1–12. doi:10.1109/TVCG.2017.2744878. 2

[YLL*20] YANG W., LI Z., LIU M., LU Y., CAO K., MACIEJEWSKI R., LIU S.: Diagnosing concept drift with visual analytics. In *Proc. IEEE Conference on Visual Analytics Science and Technology (VAST)* (2020), pp. 12–23. doi:10.1109/VAST50239.2020.00007. 2

[YSA*18] YANG K., STOYANOVICH J., ASUDEH A., HOWE B., JAGADISH H. V., MIKLAU G.: A nutritional label for rankings. In *Proc. the International Conference on Management of Data (SIGMOD)* (2018), pp. 1773–1776. doi:10.1145/3183713.3193568. 2

[ZBH*17] ZHANG C., BENGIO S., HARDT M., RECHT B., VINYALS O.: Understanding deep learning requires rethinking generalization. In *Proc. the International Conference on Learning Representations (ICLR)* (2017). 2