KD Explorer

Bruno Coelho , Guan-de Wu
New York University Tandon School of Engineering
{bgc5612, gw2145 }@nyu.edu

Abstract

In this study, we present KD-Explorer, a visual analytics system for understanding and diagnosing knowledge distillation. Gaining insight about knowledge distillation models and finding possible vulnerabilities is a non-trivial task because the student model relies directly on the teacher model, and it is can be hard to decide which one causes the vulnerability. To address this problem, we propose a multi-view visual interface to analyze the behaviors of the student and teacher model based on superclass-level metrics, class hierarchy, and feature maps. By comparing the behaviors and learned features of the two models, the user can discover and locate potential vulnerabilities. In our experiments, we implement our knowledge distillation models on the CIFAR-100 dataset with two ResNet models. Our experiments verify the effectiveness of knowledge distillation by showing that the student model achieves similar performance to the teacher model. Our main contributions are: A pipeline for training and analyzing knowledge distillation models; A visual analytic system that allows the user to compare the teacher and student models.

1 Introduction

Knowledge distillation is a hot topic in machine learning as it can enable a model to achieve excellent performance with limited memory and time consumption. However, understanding and diagonalizing the student model is a non-trivial task due to the following two challenges.

Firstly, diagnosing the black box of a deep neural network is challenging, involving a wide range of aspects beyond just accuracy. To verify the effectiveness of a model, researchers often need to check the model's behavior in different classes and visualize the learned features. Though many approaches have been proposed for these specific issues, few have attempted to combine them and build an out-of-box tool.

Then, even though we could diagnose the model and find its errors via the above approaches, it is not easy to locate the cause of the errors. The behavior of the student model depends on both the teacher model and the distillation process. Therefore, to reason about errors found, the user must investigate both the student and teacher models.

To demonstrate the usability of the system, we implement our knowledge distillation models on the CIFAR-100 dataset. We select this dataset because it categorizes classes into different superclasses, enabling the analysis of superclass-level metrics and confusion matrices. Then, we implement the teacher model by ResNet-52 and the student model ResNet-20. ResNet models achieve state-of-the-art performance [Gou *et al.*, 2021] with affordable hardware requirements, which is the reason why we choose them.

To summarize, our contributions are:

- Building a pipeline that covers both training and post processing for analyzing knowledge distillation models;
- Implementing a visual analytic system to compare the teacher and student models.

We structure this paper as follows. Beginning with section 2, we cover relevant literature on visualizing and understanding knowledge distillation. In section 3, we describe our methodology and details about the models used and how they were trained, while Section 4 dives into the system design of our visualization. Finally, we discuss limitations and conclude the paper in section 5.

2 Related Work

2.1 Visual Analytics for Neural Networks

While deep neural networks play a significant role in computer vision (CV), natural language processing (NLP), the difficulty of interpreting and debugging complex networks hinders the deployment of the models. To deploy a model in real scenarios, the researchers need to evaluate the model behaviors and verify the model effectiveness based on a wide range of aspects, e.g., the model parameters and experiment results. In response to it, both machine learning and visualization communities have attempted to use visual analytics to aid neural network development by involving human analysis in the process.[Sacha *et al.*, 2018] They often apply visual analytics by visualizing the learning process, model parameters, experiment results.

Tracking and understanding the training process is essential in model development as it provides rich insights for the model assessment and refinement. Dylan Cashman et al.

present RNNBow [Cashman *et al.*, 2017], a web-based interactive system to understand the training process of RNN models by visualizing the gradient contributions from one element of a sequence to previous elements of a sequence. Nicola Pezzotti et al. propose DeepEye [Pezzotti *et al.*, 2018] to identify stable layers during the training process by a novel perplex histogram and activation visualization.

Deep neural networks are highly parameterized and counter-intuitive. Machine learning communities have proposed various visualization approaches based on guided backpropagation[Selvaraju *et al.*, 2020], attention map[Lu *et al.*, 2012], and regularization [Nguyen *et al.*, 2015] to address it. However, those works only present static results without interaction which is later introduced by some visual analytic works. For example, Mengchen Liu et al. present a visual analytics toolkit called CNNVis to reveal the neurons' learned features, activations, and interactions. [Liu *et al.*, 2016] Minsuk Kahng et al. propose ACTIVIS, a visual analytics system allowing users to explore industry-scale deep neural networks by activation visualization and flexible interaction[Kahng *et al.*, 2017].

When the evaluation metrics, e.g., accuracy, can give the user an impression on experiment results, finer-grained quantitative measures can characterize the neural networks and better understand the model. For example, Bilal Alsallakh et al. reveal and analyze this hierarchy of similar classes by group-level metrics, e.g., precision, recall, and F measure[Bilal *et al.*, 2017].

To investigate knowledge distillation models, we combine the visualization of parameters and results to evaluate the model behaviors. Since we are focused on CNN, we employ the conventional feature visualizations of CNN. Then we compare the teacher and student models based on similar group-level metrics used in [Bilal *et al.*, 2017]. Compared with the above approaches, our system focuses on comparing the teacher model and the student model in knowledge distillation, which has not been studied in previous literature to the best of our knowledge.

3 Methodology

3.1 Knowledge Distillation

We follow the general knowledge distillation process proposed by Hinton and others [Hinton *et al.*, 2015] which introduces temperature scaling to knowledge distillation. More formally, for a specific example in our dataset, we would like our student model to learn the probability of class i, p_i, given the teacher's probabilities for all classes, $z_1, ..., z_{100}$, and temperature parameter T given by Equation 1.

$$p_i = \frac{exp\left(\frac{z_i}{T}\right)}{\sum_j \exp\left(\frac{z_j}{T}\right)} \tag{1}$$

The teacher model uses a traditional Cross-Entropy (CE) loss of the highest probability predicted class and the correct class. The student model uses a weighted sum between the CE loss and the Mean Squared Error (MSE) of it's prediction and the teacher's prediction.

For our experiments, we fix the temperature at 20 and use equal weights (0.5) for both CE and MSE losses.

3.2 Models and dataset

We follow the same general architecture for both the student and teacher model, using a ResNet [He *et al.*, 2015] 20 and 56 respectively.

The models are trained on the popular CIFAR 100 [Krizhevsky, 2009] dataset, consisting of 60,000 32x32 images. Each class has 500 training images and 100 testing images. During the visualization only, we use the fact that our 100 classes can be grouped into 20 superclasses to better analyze the impact of the hierarchical knowledge learnt by our student model.

3.3 Training

For both models, Stochastic Gradient Descent (SGD) with Nesterov momentum 0.9, a learning rate of 0.1 and weight decay 5×10^{-4} was used during training. The teacher model also uses a Cosine Annealing learning rate, while the student model doesn't due to the use of temperature scaling. Both models started with general weights obtained from a pretrained Model Zoo [Cheng *et al.*, 2020] due to time constraints and were then fine tuned using the PyTorch [Paszke *et al.*, 2019] framework for general model definition and boilerplate code, and KD-Lib [Shah *et al.*, 2020] for knowledge distillation helper functions. We stick to the formulation explained in Section 3.1 but the code is easily extendable to other training methods, such as Teacher Assistant or Route Constrained Optimization knowledge distillation.

3.4 Feature maps

There are many different visualization techniques for understanding CNN models [Adebayo *et al.*, 2020]. We refer to them broadly as saliency maps or feature maps (used throughout the paper).

In this work, we focus on Guided Backpropagation [Springenberg *et al.*, 2015], a technique that builds upon the earlier work of DeconvNets [Zeiler and Fergus, 2013]. Considering a single example in the dataset which has been forwarded propagated, DeconvNets use vanilla backpropagating, but consider the influence of the inputs instead of the network's parameters. By doing so, we can view the resulting "image" as showing us what parts of our input our network pays attention to.

Guided Backpropgation builds on top of that by also considering only the positive inputs and only the positive error signals when forward/backpropagating before a ReLU layer. We use the M3d-Cam [Gotkowski *et al.*, 2020] implementation.

3.5 Model results

General results can be seen in Table 1. We present the averaged accuracy and precision, since in a multiclass setting, micro-recall = micro-F1 score = micro accuracy. The subclass specific results including recall for each subclass can be seen in our tool. These averaged results are consistent with the general range of accuracies reported by Jianping Guo et al. survey on Knowledge Distillation [Gou *et al.*, 2021].

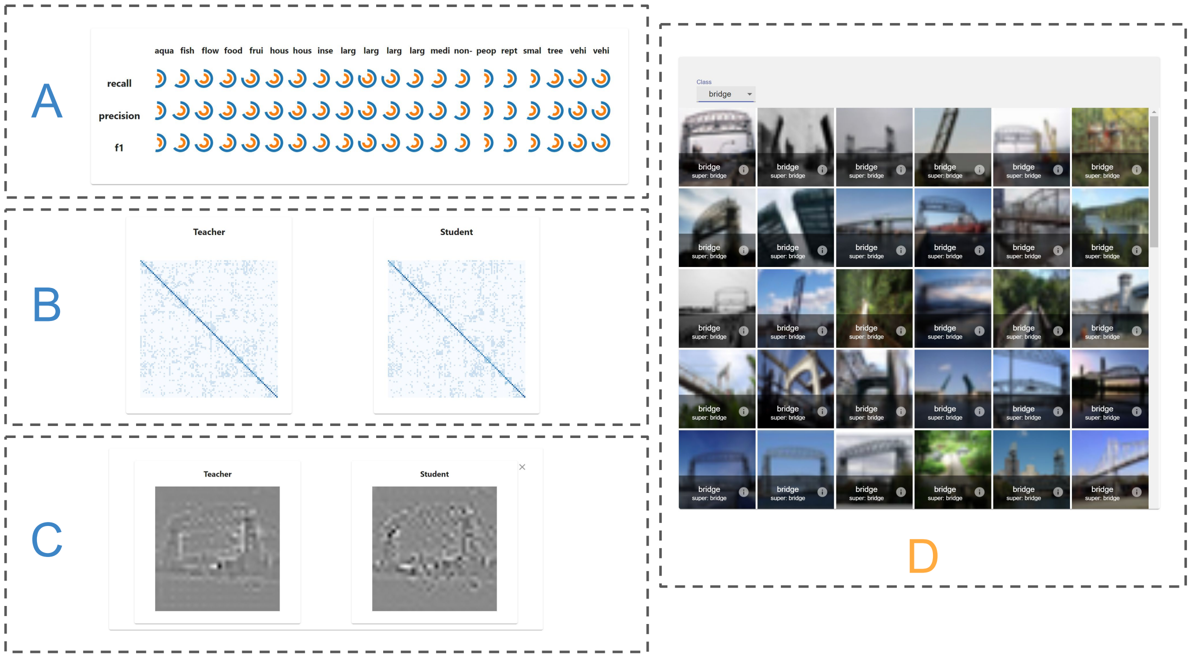
A
aqua fish flow food frui hous hous inse larg larg larg larg medi non- peop rept smal tree vehi vehi
recall
precision
f1
B
Teacher
Student
C
Teacher
Student
D
Class
bridge
bridge
super: bridge

Figure 1: View of our interface. View A is the quantitative evaluation view showing the superclass-level metrics. View B presents the confusion matrices of the teacher and student models. View C is a pop-up view showing the feature maps of the teacher and student. View D refers to the image corpus, allowing the users to query the images in the dataset by their classes

Model	Amount of Parameters	Accuracy	Precision
Teacher	861K	0.674	0.678
Student	278K	0.608	0.630

Table 1: Relevant metrics averaged over all 100 classes. Note that since our classes are equally balanced, both micro and macro averaging give the same results.

4 System Design

4.1 System Overview

We build our system interface based on D3.js [Bostock *et al.*, 2011] and React. To enable quick access to the images in Cifar-100 dataset, we upload them to Google Cloud Storage and allow the user to access them via the interface. Our interface consists of four views, i.e., quantitive evaluation, confusion matrices, image corpus, and feature visualization. The quantitive evaluation view is a group of circular bars showing the superclass-level metrics, i.e., precision, accuracy, and F-measure. Besides the superclass-level metrics, the system discloses the correlation between different superclasses by the confusion matrix view, which presents the possibility that a model predicts a class as another. The views of student and teacher models are placed side by side to support users' comparison. The image corpus contains all the images, and the user can select an image to check its details. The feature map view is a pop-up view showing the feature map of the user-selected image. Figure 1 shows an overview of our tool.

4.2 Quantitive Evaluation

Quantitive evaluation presents the superclass-level metrics by circular bars, i.e., precision, accuracy, and F-measure. Since the three metrics are ranged from 0 to 1 theoretically, we can encode them by a circular bar and map the value to the range of 0 to 2π. The outer and inner circles encode the teacher and student models, respectively, because the teacher model is more likely to have a more significant metric value than the student model.

We compare the design with two alternatives, i.e., group bar chart and divergent bar chart. Our design outperforms them by more concise layout and easier comparison between student and teacher models. Compared with our design, the group bar chart requires more extensive space and can not place all superclasses in a limited area. Though the divergent bar chart allows the users to compare the metric values across different superclasses easily, it is hard to compare the student and teacher models. By comparison, circular bar charts can enable intuitive comparison between student and teacher models with a concise space.

4.3 Confusion Matrices

Besides the accuracy, investigating the patterns of the predictions can provide an insight into the reasons behind the classification errors. Previous studies [Bilal *et al.*, 2017] observed that classification error follows a hierarchical pattern over the classes. For example, a model is more likely to confuse the cat and dog images than the cat and flight images. Thus, conditioning on a similar precision, the model predicting with a more evident hierarchical pattern may better understand the

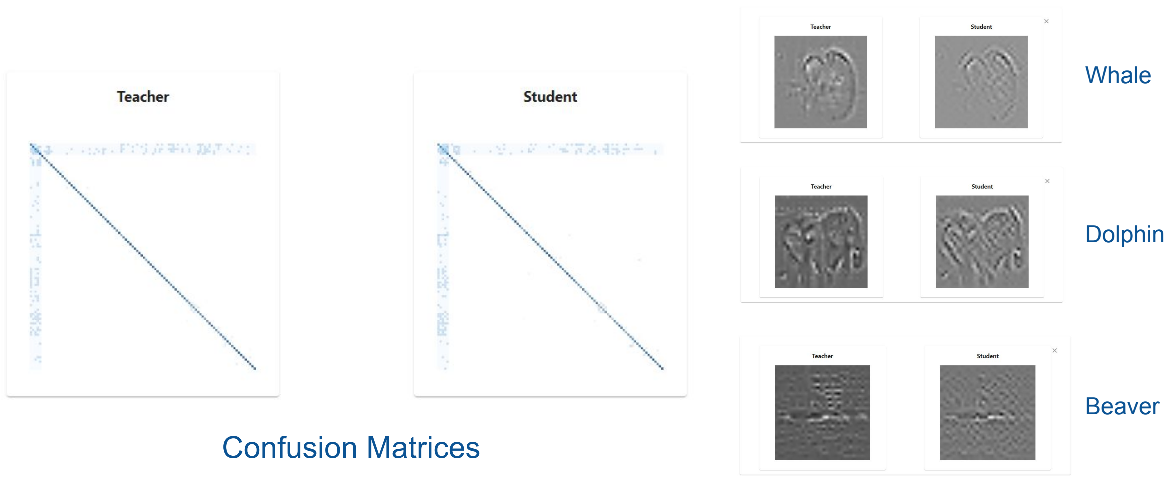

Figure 2: Our use case demonstrates the usability of our system. In this case, the user explores the reasons why the student model fails in classifying aquatic animals. The left figure is the highlighted confusion matrices of aquatic animals when the right three figures refer to the feature maps of the three classes belonging to aquatic animals.

different superclasses. CIFAR-100 dataset' superclasses provide an out-of-box class group schema, and we can employ it to investigate the hierarchical patterns of the prediction error. In knowledge distillation, teachers' predictions on incorrect classes are essential because they provide prior student model training. Thus, the prediction's difference on incorrect classes can indicate whether the student model learns the prior knowledge from the teacher. To present the pattern, we create two side-by-side confusion matrices exhibiting the frequency of the prediction class on another for teacher and student models, respectively. We group the classes based on CIFAR-100's superclasses and allow the users to inspect the patterns intra- or inter-superclass interactively. When the user clicks a superclass, the matrices will gray out other irrelevant cells.

4.4 Image Corpus

Image Corpus view presents the images by a grid layout. The user can filter the images based on their classes. When the user clicks a superclass in quantitive evaluation or Confusion Matrix views, the image corpus will also filter the images based on the specified superclass. We store the images in Google Cloud Storage to speed up user access.

4.5 Feature Map

The feature map as defined in Section 3.4 pops up is shown when the user selects an image in Image Corpus. This allows the user to better understand what parts of the input image the model finds important when classifying specific images.

4.6 Use Case

We will describe a use case to demonstrate the usability of our system. When a user first opens the system, the system will show an overview of the student and teacher models by quantitative evaluation and confusion matrices as Figure-1. The user first examines the quantitative evaluation and analyzes the student model's performance compared with the teacher model. The user can extract a series of the superclasses, i.e., aquatic animals, median-size animals, non-insect invertebrates, and vehicles 1, on which the student performs worse than the teacher model significantly. Then, The user first gets interested in aquatic animals because the most significant performance decline is observed in this superclass. The user can click and further investigate the superclass by the confusion matrix view as Figure-2 shows. The confusion matrix view will highlight the selected superclass in both teacher and student models. The user realizes the teacher model frequently labels the aquatic animal samples by other classes, e.g., reptiles and small animals. Thus, the user can guess that the teacher model does not construct a proper prior for the superclass, leading to the student model's failure. Then the user continues to inspect the images and the corresponding feature maps. The user first selects beaver and finds both models do not extract a salient feature map. Then checking other aquatic animals, the user can get similar findings. Thus, the user can conclude that the teacher model can mainly cause the student's failure by failing to understand the features of aquatic animals.

5 Limitations and Conclusion

5.1 Limitations

The CIFAR-100 dataset, while frequently used for analyzing knowledge distillation due to it's hierarchical structure, is not ideal for visualizing attention maps due to using small images as input. Due to the time and resource constraints, no other datasets were considered, which leaves an interesting area of exploration for future work.

By using attention maps visualization as a major part of our tool, a natural question to ask is which is the best technique for understanding the inner working of CNN models. This is still an open research question, specially since there isn't a unique metric to measure the quality of different attention maps and judging them on solely on visual properties can be misleading. In this paper we focused on Guided Backprop due to it's popularity and ease of implementation, but Julius and others [Springenberg *et al.*, 2015] argue that it fails some basic sanity checks when compared to other methods and suggest GradCAM[Selvaraju *et al.*, 2020] as a possible alternative. Further work could allow our tool to directly allow the user to specify a visualization map, therefore also helping in mitigating the bias in each methods.

5.2 Conclusion

This works introduces a general pipeline and visualization tool for understanding and diagnosing knowledge distillation models. We consider challenges that researches in this area face such as evaluating a model's behavior and verifying it's effectiveness based on a wide range of metrics and visualizations.

To deal with these challenges, we design a general processing pipeline that works with any CNN based neural network and is easily able to be adaptable to other needs. Our visualization tool then provides multiple ways to better understand the models considered by integrating raw quantitative metrics, class correlation and feature map visualization into an interactive view. Finally, we consider use cases for our tool and discuss it's features and limitations.

References

[Adebayo *et al.*, 2020] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity Checks for Saliency Maps. *arXiv:1810.03292 [cs, stat]*, November 2020. arXiv: 1810.03292.

[Bilal *et al.*, 2017] Alsallakh Bilal, Amin Jourabloo, Mao Ye, Xiaoming Liu, and Liu Ren. Do convolutional neural networks learn class hierarchy? *IEEE transactions on visualization and computer graphics*, 24(1):152–162, 2017.

[Bostock *et al.*, 2011] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[Cashman *et al.*, 2017] Dylan Cashman, Genevieve Patterson, Abigail Mosca, and Remco Chang. Rnnbow: Visualizing learning via backpropagation gradients in recurrent neural networks. In *Workshop on Visual Analytics for Deep Learning (VADL)*, volume 4, 2017.

[Cheng *et al.*, 2020] Xu Cheng, Zhefan Rao, Yilan Chen, and Quanshi Zhang. Explaining Knowledge Distillation by Quantifying the Knowledge. *arXiv:2003.03622 [cs, stat]*, March 2020. arXiv: 2003.03622.

[Gotkowski *et al.*, 2020] Karol Gotkowski, Camila Gonzalez, Andreas Bucher, and Anirban Mukhopadhyay. M3d-cam: A pytorch library to generate 3d data attention maps for medical deep learning, 2020.

[Gou *et al.*, 2021] Jianping Gou, Baosheng Yu, Stephen John Maybank, and Dacheng Tao. Knowledge Distillation: A Survey. *International Journal of Computer Vision*, March 2021. arXiv: 2006.05525.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. arXiv: 1512.03385.

[Hinton *et al.*, 2015] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531 [cs, stat]*, March 2015. arXiv: 1503.02531.

[Kahng *et al.*, 2017] Minsuk Kahng, Pierre Y Andrews, Aditya Kalro, and Duen Horng Chau. A cti v is: Visual exploration of industry-scale deep neural network models. *IEEE transactions on visualization and computer graphics*, 24(1):88–97, 2017.

[Krizhevsky, 2009] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[Liu *et al.*, 2016] Mengchen Liu, Jiaxin Shi, Zhen Li, Chongxuan Li, Jun Zhu, and Shixia Liu. Towards better analysis of deep convolutional neural networks. *IEEE transactions on visualization and computer graphics*, 23(1):91–100, 2016.

[Lu *et al.*, 2012] Yao Lu, Wei Zhang, Cheng Jin, and Xiangyang Xue. Learning attention map from images. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1067–1074. IEEE, 2012.

[Nguyen *et al.*, 2015] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 427–436, 2015.

[Paszke *et al.*, 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[Pezzotti *et al.*, 2018] Nicola Pezzotti, Thomas Höllt, Jan Van Gemert, Boudewijn P.F. Lelieveldt, Elmar Eisemann, and Anna Vilanova. Deepeyes: Progressive visual analytics for designing deep neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):98–108, 2018.

[Sacha *et al.*, 2018] Dominik Sacha, Matthias Kraus, Daniel A Keim, and Min Chen. Vis4ml: An ontology for visual analytics assisted machine learning. *IEEE transactions on visualization and computer graphics*, 25(1):385–395, 2018.

[Selvaraju *et al.*, 2020] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization. *International Journal of Computer Vision*, 128(2):336–359, February 2020. arXiv: 1610.02391.

[Shah *et al.*, 2020] Het Shah, Avishree Khare, Neelay Shah, and Khizir Siddiqui. Kd-lib: A pytorch library for knowledge distillation, pruning and quantization, 2020.

[Springenberg *et al.*, 2015] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for Simplicity: The All Convolutional Net. *arXiv:1412.6806 [cs]*, April 2015. arXiv: 1412.6806.

[Zeiler and Fergus, 2013] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. *arXiv:1311.2901 [cs]*, November 2013. arXiv: 1311.2901.