DEEP NEURAL NETWORKS FOR NEAR EARTH OBJECT CLASSIFICATION IN WISE DATA

2019 SUMMER RESEARCH PROJECT

Nathan Blair
NASA JPL
nblair@berkeley.edu

Joseph Masiero
NASA JPL
joseph.masiero@jpl.nasa.gov

August 15, 2019

ABSTRACT

The Near-Earth Object Camera (NEOCam) is a new satellite that will conduct a thermal infrared survey of the sky searching for nearby asteroids. The data volume will be orders of magnitude larger than previous surveys such as NEOWISE, and thus new data processing techniques are required. We use data from the NEOWISE survey to construct simulated NEOCam data, and train machine learning models to classify near earth objects in this data. We outline the pros and cons of various model architectures: linear, multi-layer perceptron, kernel methods, random fourier projections, and custom designed deep networks. With our best models, we achieve near-human performance on the classification task.

1 Introduction

Even for trained experts, it is challenging to differentiate between Near Earth Objects (NEOs), like asteroids and comets, and other celestial objects like stars and image artifacts that appear in infrared images of space. This problem is uniquely suited for machine learning methods, as it combines a high data volume with expert supervised data. This research applies the latest deep learning techniques to annotated NEO data.

The NEOWISE mission used a combination of classical computer vision, statistical methods and human Quality Assurance (QA) to identify NEOs from 4-wavelength infrared satellite images of space. The NEOWISE pipeline processes an enormous amount of data, as NEOWISE takes an image every six seconds. However, the new proposed satellite NEOCam is set to produce volumes of data orders of magnitude larger than those of NEOWISE. This means that human QA must be almost entirely eliminated from the data-processing pipeline to avoid the allocation of expensive human resources.

NEOWISE data is particularly suited for training machine learning models because it has been annotated by humans experts and contains a large number of datapoints [1]. NEOWISE data comes in the form of QA sheets. Each QA sheet is classified as either a true detection or a false detection of an object by the WISE satellite. There are, however, a few challenging aspects of NEOWISE data. The first is that it is mixed format data. Each QA sheet contains both visual and numerical data about the proposed real object. Furthermore, the NEOWISE dataset contains missing and variable length data. The number of potential observations of an object per QA sheet varies across the dataset. Finally, it is important that our results generalize to NEOCam data, so we must build a robust architecture that is tailored for retraining as actual NEOCam data begins to come in.

2 Contribution

We evaluate the viability of machine learning models for NEO classification, including linear models, multi-layer perceptrons, kernel methods and custom designed deep networks. We test these models on a NEO classification dataset created by the NEOWISE/NEOCam team. We report the classification accuracy of each of these methods as well as the relative model complexity and over-fitting error. We also discuss visualization techniques for model diagnostics and directions for further research.

3 Approach

3.1 Dataset and Pipeline

Each datapoint in our dataset is called a QA sheet. Each QA sheet contains data attached to k observations of space taken by the WISE satellite. The QA sheet is labeled as a positive example if at least 5 of the k observations contains the same NEO at different points in space. Each observation is made up of a small 2-channel image and a vector of scalars. Furthermore, each QA sheet contains a composite image created by merging the k images and another vector of scalar data that corresponds to the QA sheet as a whole rather than particular observations. Because k varies across the dataset, the model must have a mechanism to deal with variable length data to train on the full dataset. For this reason, we only use the fixed-length portion of the dataset, composite images and some scalars, for certain simpler models. From this point, we will refer to this subset as the fixed-length dataset.

We train on 8 NVIDIA Titan RTX GPUs for high performance.

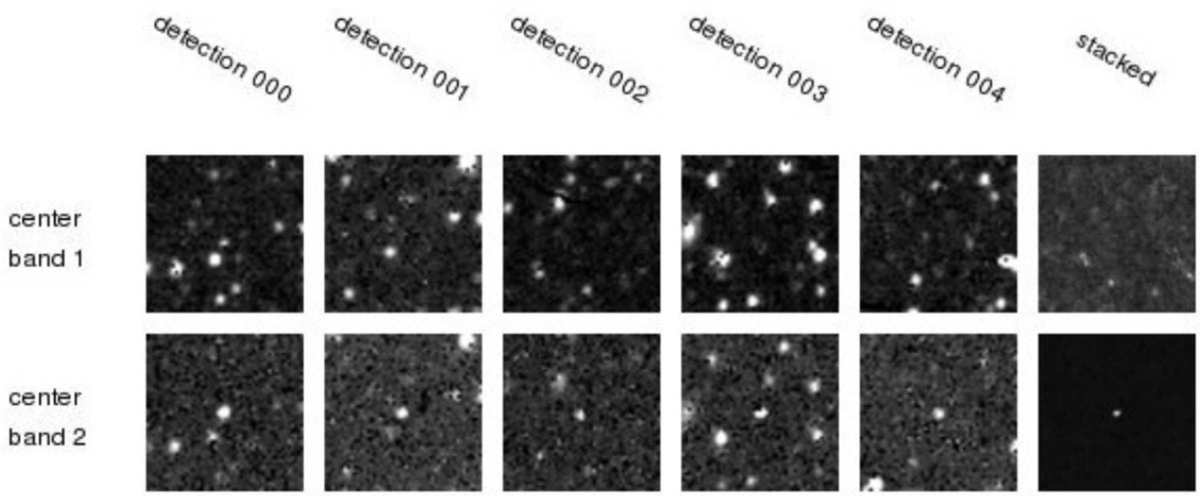


Figure 1: The visual data in a QA sheet with 5 observations

3.2 Linear Models and Variants

We trained several linear models on the fixed-length dataset. We trained our simplest linear models (Ridge) by solving the linear least squares equation with ridge regularization [2]. For added complexity, we used kernel methods. This extension of linear methods allows for generalization of linear methods to non-linear spaces [3]. We tested both polynomial (Quadratic) and radial basis function (RBF) kernels. Finally, we experimented with random fourier feature projections (RFF) before sending the data through a linear model [4].

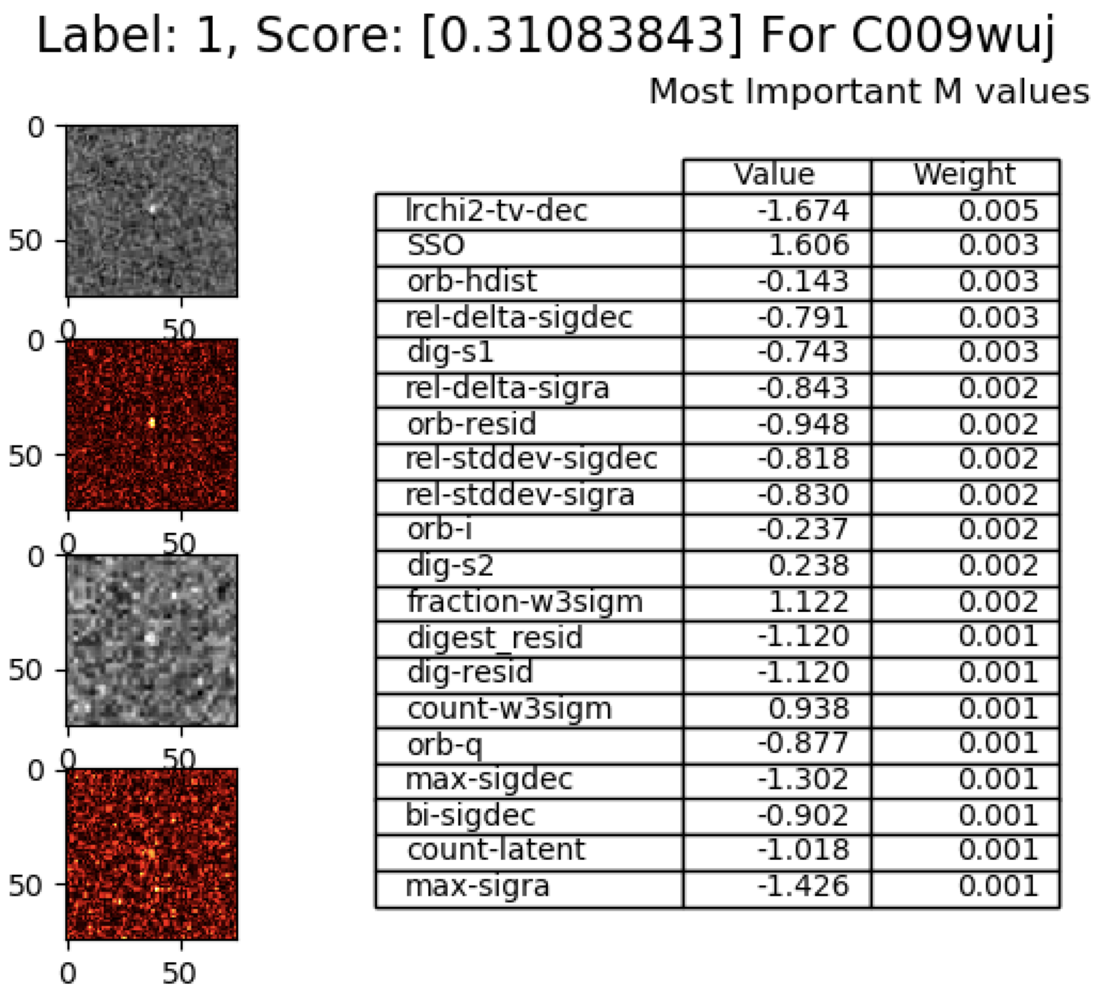
We extended linear models to the full dataset by training a different linear model for different length portions of the dataset (Multiple Linear). When applying this model, however, the number of observations in a QA sheet is not bounded. So, we trained separate models for QA sheets with a popular number of observations and trained the rest of the QA sheets with a model trained on the fixed-length dataset.

3.3 Deep Neural Networks

We have experimented with a variety of deep neural network architectures. At the simplest level, we used a multi-layer perceptron (MLP) with ReLU nonlinearities on the fixed-length dataset [5, 6]. While this model achieves high accuracy, it does not utilize all of the information available in the full dataset. Next, we implemented a model that trains a suite of MLP models (Multiple MLP) for different length portions to the dataset. This is similar to the Multiple Linear model outlined at the end of the previous subsection.

While the Multiple MLP model can train on the full dataset, each of the separate MLP models it is comprised of do not communicate with eachother, so it is highly innefficient. To avoid this problem, we used Siamese networks and max-pooling to construct a model (Basic Siamese NEO Network, Basic SNEON) that is able to condense information from a variable number of observations [7]. Each observation is sent through the same MLP network and condensed into a single number that represents the confidence on the model that that observation is real. Then, the model takes the fifth largest value and considers that confidence along with the fixed-length data to make a final determination about whether or not to accept the QA sheet. We take the fifth largest value because we require at least 5 good observations to accept a QA sheet by NEOWISE convention. We also trained a model that replaces the Siamese MLP network in this model with a Siamese ResNet (SNEON) [8].

3.4 Visualization



	Value	Weight
lrchi2-tv-dec	-1.674	0.005
SSO	1.606	0.003
orb-hdist	-0.143	0.003
rel-delta-sigdec	-0.791	0.003
dig-s1	-0.743	0.003
rel-delta-sigra	-0.843	0.002
orb-resid	-0.948	0.002
rel-stddev-sigdec	-0.818	0.002
rel-stddev-sigra	-0.830	0.002
orb-i	-0.237	0.002
dig-s2	0.238	0.002
fraction-w3sigm	1.122	0.002
digest_resid	-1.120	0.001
dig-resid	-1.120	0.001
count-w3sigm	0.938	0.001
orb-q	-0.877	0.001
max-sigdec	-1.302	0.001
bi-sigdec	-0.902	0.001
count-latent	-1.018	0.001
max-sigra	-1.426	0.001

Figure 2: Saliency visualization for a model trained on the fixed-length dataset. The red images are saliency maps for the corresponding image channel. The table shows the relative importance of some of the fixed-length scalars.

Visualizing complicated machine learning models allows us to understand how they are making decisions. We used the popular gradient explanation along with the SmoothGrad method to create saliency maps of our models' responses to various input [9]. The gradient explanation displays the gradient of output of a model with respect to the input. Therefore, we can visualize the relative sensitivity of the output to each value in the input. This method has been shown to pass sanity tests that make it a powerful and simple visualization technique [9]. The SmoothGrad method is used to increase the noise robustness of our visualizations [10]. An example output of our visualization technique is shown in Figure 2.

4 Results

We report the best test accuracy for each of the models that we trained. It is noteworthy that these accuracies do not represent theoretical limits for these models. Different hyper-parameters and more training can result in better accuracy than we report, especially for more complex models.

4.1 Human Expert

For the classification task at hand, the average human expert achieves 97-98% accuracy. Because of the way the dataset was created, it has some noise in the labels. We estimate that the labels in the dataset are around 98% accuracy.

4.2 Linear Ridge Regression

Our Ridge model performs surprisingly well using the fixed-length dataset, achieving 94.6% accuracy, with only 3% overfitting. This shows the power of simple models. Often, the simplest solution makes the most sense. These models are by far the easiest to explain and visualize: we simply display the weights of the model.

4.3 Kernel Methods

Our Quadratic kernel model overfits even with high regularization, achieving 90% accuracy with 9% overfitting.

Similarly, our RBF kernel model reached 87% accuracy with 12% overfitting.

While both of these models easily fit the training data, they also memorized the noise in the training set. It seems that these models are not well suited for the classification task at hand.

4.4 Random Fourier Features

We did not see accuracy increases from by projecting our model into a random fourier space before classification, attaining 91.6% test accuracy and 5.7% overfitting. However, this score is possibly boostable by increasing the dimensionality of the projected space. While this method should not be discounted as a possible way to increase model performance, we believe that the RFF method is generally inferior to some of neural network methods which also introduce non-linearities but train more easily.

4.5 Multi-Layer Perceptron

MLP models have a good balance of simplicity and effectiveness. While they can only take in fixed-length data, their ability to be both deep and wide allows them to fit extremely complex functions.

We achieved 95% accuracy with 4.2% over-fitting on our best MLP model. This model had a depth of 7 and a hidden width of 100 neurons. With more depth this type of model may be able to perform even better than we observed. With extra depth, we may need the addition of extra normalization layers such as batch normalization layers and residual blocks to prevent vanishing gradients or overfitting [11, 8].

4.6 Siamese Network

The siamese networks (SNEON, Basic SNEON) are unique amongst the other models we trained in that they can be applied to the full variable-length dataset while still sharing weights. That means that the siamese networks do not have to learn a separate model for different length tracklets, nor do they have to learn separate weights for different observations.

We trained the Basic SNEON network to 95% accuracy with 3.4% overfitting error and the regular SNEON network to 80% accuracy. The regular SNEON network obviously did not train properly, and could certainly be retrained to a higher score. However, we realized that the SNEON architecture with ResNet modules was overly complex for the problem at hand. The Basic SNEON model, like the MLP model, could likely be trained to higher accuracy by increasing the complexity and training time of the model.

4.7 Multiple Linear and Multiple MLP

In general the Multiple Linear and Multiple MLP fits were not successful. They performed worse than fits on a smaller subset of the dataset. For the linear model, this may partially be because we could only run on a small slice of the dataset due to memory constraints. On the MLP model, it is possible that partitioning the dataset onto different models made it hard to train the models with access to only a small portion of the dataset. With these limitations, the Multiple MLP model reached 87% accuracy with 10% overfitting error and the Multiple Linear model reached 88% accuracy with 9% overfitting error. It may be possible to improve these models with modification. For example, by allowing every sub-model have access to the entire fixed-length dataset, or rather access to the output of a model trained on the fixed-length dataset.

4.8 Aside on the Dataset and Confusion Matrix

Our dataset was not evenly split between positive and negative examples. In fact, 77% of the dataset was negative examples. Because of this, the models we trained had a strong negative bias. Our linear model with 94% accuracy had 96% negative predictive value and 95% true negative rate. On the other hand, the model had 84% positive predictive value and 88% true positive rate. This result was similar across models. For our MLP model: 95% accuracy, 96% negative predictive value, 98% true negative rate, 92% positive predictive value and 80% true positive rate.

In general, an uneven dataset can make machine learning models more difficult to train. However, uneven datasets also often more accurately represent the true distribution of datapoints in the test environment.

5 Related Works

5.1 Astronomical Deep Learning

Machine learning models are beginning to become more popular in astronomy as data volume has increased and data-driven methods have begun to outperform expert-model driven methods. Still, most astronomical surveys use computationally simpler methods than Deep Neural Networks such as Template Matching, K-Means Clustering, Support Vector Machines, Random Forests, and shallow Artifical Neural Networks [12, 13, 14, 15, 16]. DNNs have, however, been used with success for detecting and classifying exoplanets and small galaxies [17, 18, 19]. **The most similar work that we have found to our project is "Deep Learning for Image Sequence Classification of Astronomical Events" by Rodrigo Carrasco-Davis et al.**, which uses a Deep RNN model for classification over a sequence of images [20]. We did not attempt a sequence based model like RNN. We also face a slightly more difficult problem of dealing with multi-modal, multispectral data, bad data, variable length data, and missing detections.

5.2 Multi-Modal, Variable Length and Robust Deep Learning

There is a deep field of research into making machine learning algorithms applicable in the face of unwieldy data. We present a few of these methods which we have taken inspiration from for the purpose of this project.

Ngiam et al. show one example of combining multiple modalities for deep learning tasks by combining audio and visual data which leads to performance increases [21]. In our work, we hope that combining visual and numerical data will allow our model to learn a more accurate and more robust representation of the data.

Much work has been done on classification for variable length sequences. The extremely popular ResNet architecture uses an average pool operation to compress images of variable size after passage through a fully convolutional deep neural network [8]. In similar applications, it is possible and popular to replace this average pool operation with a max or min pool. In a sequence processing setting, like natural language processing or audio processing, recurrent neural networks (RNNs) and long short term memory (LSTM) networks are popular for their ability to extract meaning from moderate length sequences [22, 23]. Recently, it has been shown that attention mechanisms may be able to outperform both RNNs and LSTMs for the same tasks [24]. The Siamese network takes a sequence of data and applies the same network to each element in the sequence [7]. We use this as a basic building block for our SNEON DNN architectures.

It is also important that our learned model is moderately robust to distribution shift in the input data. This will be crucial if the sensors become more noisy over time, if the type of data changes seasonally, or if we decide to use the NEOWISE trained model on NEOCam data. Hendrycks et al. give an overview of recent noise and adversarial robustness work and provide methods and benchmarks for achieving robust models [25]. Much of this work boils down to regularizing models to avoiding over-fitting to noise. It is also common to fine-tune models on new data when it arrives [26]. As new data from NEOCam arrives, we could combine fine tuning and online learning to make a reinforcement-learning like paradigm [27].

6 Conclusion and Future Work

Machine learning models can save thousands of man-hours and make automatic processing pipelines in astronomy much more efficient. However, visualization and explainability are just as important as accuracy; it is essential that these models are more than just a black box. We want to understand when and why our models are failing.

We found that neural network models have slightly more success than linear and other simple models, and also have significantly more capacity for improvement. However, neural network models tend to be more opaque, and more difficult to train. By using saliency maps such as the gradient explanation and keeping close track of the model's confusion matrix on the test set, we are able to train neural networks without having to sacrifice too much explainability.

In the future it will be important to clean our dataset and fine-tune our models to increase the performance before deploying the best model on a real system.

References

[1] R. M. Cutri, E. L. Wright, T. Conrow, J. Bauer, D. Benford, H. Brandenburg, J. Dailey, P. R. M. Eisenhardt, T. Evans, S. Fajardo-Acosta, J. Fowler, C. Gelino, C. Grillmair, M. Harbut, D. Hoffman, T. Jarrett, J. D. Kirkpatrick, D. Leisawitz, W. Liu, A. Mainzer, K. Marsh, F. Masci, H. McCallon, D. Padgett, M. E. Ressler, D. Royer, M. F. Skrutskie, S. A. Stanford, P. L. Wyatt, D. Tholen, C. W. Tsai, S. Wachter, S. L. Wheelock, L. Yan, R. Alles, R. Beck, T. Grav, J. Masiero, B. McCollum, P. McGehee, M. Papin, and M. Wittman. Explanatory Supplement to the WISE All-Sky Data Release Products. Technical report, March 2012.

[2] A. E. Hoerl and R. W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

[3] Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification. Technical report, Department of Computer Science, National Taiwan University, 2003.

[4] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, pages 1177–1184, USA, 2007. Curran Associates Inc.

[5] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Netw.*, 2(5):359–366, July 1989.

[6] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines vinod nair. volume 27, pages 807–814, 06 2010.

[7] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. 2015.

[8] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 689–696, USA, 2011. Omnipress.

[9] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *CoRR*, abs/1810.03292, 2018.

[10] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda B. Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *CoRR*, abs/1706.03825, 2017.

[11] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[12] Dalya Baron. Machine learning in astronomy: a practical overview, 2019. cite arxiv:1904.07248Comment: 37 pages, 18 figures.

[13] E. Bertin. Classification of astronomical images with a neural network. *Science with Astronomical Near-Infrared Sky Surveys*, page 49–51, 1994.

[14] Emmanuel Bertin. Mining pixels: The extraction and classification of astronomical sources. *ESO ASTROPHYSICS SYMPOSIA Mining the Sky*, page 353–371, 2001.

[15] E. Bertin and S. Arnouts. SExtractor: Software for source extraction. , 117:393–404, June 1996.

[16] N. Drory. Yet another object detection application (yoda). *Astronomy Astrophysics*, 397(1):371–379, Nov 2002.

[17] M. Huertas-Company, J. R. Primack, A. Dekel, D. C. Koo, S. Lapiner, D. Ceverino, R. C. Simons, G. F. Snyder, M. Bernardi, Z. Chen, and et al. Deep learning identifies high-z galaxies in a central blue nugget phase in a characteristic mass range. *The Astrophysical Journal*, 858(2):114, 2018.

[18] Kyle A. Pearson, Leon Palafox, and Caitlin A. Griffith. Searching for exoplanets using artificial intelligence. *Monthly Notices of the Royal Astronomical Society*, 474(1):478–491, 2017.

[19] Christopher J. Shallue and Andrew Vanderburg. Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(2):94, 2018.

[20] Rodrigo Carrasco, Guillermo Cabrera-Vives, Francisco Förster, Pablo Estevez, Pablo Huijse, Pavlos Protopapas, Ignacio Reyes, Jorge Martínez, and Cristóbal Donoso. Deep learning for image sequence classification of astronomical events, 12 2018.

[21] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y. Ng. Multimodal deep learning. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 689–696, USA, 2011. Omnipress.

[22] Zachary Chase Lipton. A critical review of recurrent neural networks for sequence learning. *CoRR*, abs/1506.00019, 2015.

[23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[25] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *CoRR*, abs/1903.12261, 2019.

[26] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *CoRR*, abs/1411.1792, 2014.

[27] Doyen Sahoo, Quang Pham, Jing Lu, and Steven C. H. Hoi. Online deep learning: Learning deep neural networks on the fly. *CoRR*, abs/1711.03705, 2017.