SSC23-XII-01

Evaluating Rotation-Equivariant Deep Learning Models for On-Orbit Cloud Segmentation

Alex Meredith
Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge MA 02139
ameredit@mit.edu

Faculty Advisor: Kerri Cahoy
Massachusetts Institute of Technology
kcahoy@mit.edu

ABSTRACT

Cloud detection in satellite imagery is key for autonomously taking and downlinking cloud-free images of a target region as well as studying cloud-climate interactions and calibrating microwave radiometers. We propose a C_8-equivariant dense U-Net, a rotation-equivariant deep learning model, trained on visible-spectrum, long-wave infrared (LWIR), and short-wave infrared (SWIR) imagery for on-orbit cloud detection. We train this model on the SPARCS[1] dataset of Landsat 8 images and compare it to three related deep learning models, two rule-based algorithms, and to the literature. Additionally, we compare a C_8-equivariant dense U-Net trained on VIS, LWIR, and SWIR imagery to the same algorithm trained on only VIS and LWIR, on only VIS and SWIR, and on only VIS imagery. We find that augmenting VIS imagery with SWIR imagery is most useful for missions where false positives (non-cloud pixels misidentified as cloud) are extremely costly, and that augmenting with LWIR imagery is most useful for missions where false negatives (cloud pixels misidentified as non-cloud) are extremely costly. We demonstrate also that our C_8-equivariant dense U-Net achieves over 97% accuracy (over 99.5% when evaluated with a 2 pixel buffer at the cloud boundaries) on cloud segmentation on the SPARCS dataset, outperforming existing state-of-the-art algorithms as well as human operators, while remaining computationally lightweight enough to be usable on resource-constrained missions such as CubeSats.

INTRODUCTION

In this paper, we explore the application of transformation-equivariant deep learning to on-orbit cloud segmentation. We develop a C_8-equivariant dense U-Net, a rotation-equivariant deep learning model, and train it on Landsat 8 imagery from the SPARCS[1] dataset. We evaluate our C_8-equivariant dense U-Net in three different ways. First, we compare its performance to that of three other related deep learning models and two rule-based models. Second, we train it on data from four different combinations of bands: visible-spectrum (VIS), long-wave infrared (LWIR), and short-wave infrared (SWIR); VIS and LWIR; VIS and SWIR; and VIS only. Finally, we compare its performance to that of the state-of-the-art SPARCS CNN.[1]

Cloud Segmentation

Cloud detection is a critical capability for weather and climate satellite missions and has historically been performed on the ground using downlinked satellite multispectral data and physics-driven rule-based methods. Examples of physics-driven rule based methods still evaluated on the ground include Fmask, which detects clouds, clear land, clear water, and cloud shadow in Landsat and Sentinel-2 data,[2] the continuity MODIS-VIIRS cloud mask, which detects clouds in MODIS and VIIRS data,[3] the MODIS cloud mask retrieved for the CERES mission, which detects clouds in MODIS data,[4] and the GOES cloud mask, which detects clouds and probable clouds in the GOES advanced baseline imager (ABI) data.[5] All of these methods use physics-derived reflectance and brightness temperature thresholds in different bands of a multispectral data input to determine whether a pixel is cloudy or clear. These methods often rely on determining the surface terrain type based on spectral properties of the inputted data, and have different cloud thresholds depending on the underlying terrain.

Other methods for cloud detection include rule-based methods driven by statistics, like random tree and random forest detection. The `s2cloudless` algo-

rithm is a gradient-boosted tree-based algorithm for Sentinel-2 imagery that improves upon Fmask by almost 10% – likely in part because Fmask is designed to work for Landsat or Sentinel-2, while `s2cloudless` is optimized for Sentinel-2.[6] On EO-1, Bayesian thresholding (BT) and random forest algorithms were tested for cloud detection on visible-spectrum and short-wave infrared data.[7] Rule-based methods have also been applied to visible-spectrum imagery onboard OPS-SAT for cloud detection.[8] OPS-SAT tested luminosity thresholding, a rule-based method similar to the Bayesian thresholding algorithm used aboard EO-1, and k-means segmentation onboard, and additionally tested a random forest algorithm on the OPS-SAT engineering model on the ground.[8] Deep learning is also used for cloud detection. OPS-SAT evaluated a U-Net trained on visible-spectrum Landsat imagery on-orbit for cloud detection.[8] The U-Net outperformed three rule-based methods, achieving a balanced accuracy of 77-89% over three test images.[8] A similar model achieved 91.7% overall accuracy when trained and tested on visible-spectrum and long-wave infrared Landsat images.[8] Improved results can be achieved when multispectral input data is considered – Spatial Procedures for Automated Removal of Cloud and Shadow (SPARCS), a convolutional neural network (CNN) trained on all Landsat 8 bands except the panchromatic band (B8), can discriminate between clear-sky, cloud, cloud shadow, water, and snow/ice pixels with 97.1% accuracy, excluding areas within 2 px of cloud boundaries.[1]

Deep Learning

Deep learning is a type of machine learning that uses multiple hidden layers in an artificial neural network to train a model relating an input to an output. U-Nets and dense U-Nets are two deep learning architectures based on convolutional neural networks (CNNs) which have demonstrated strong performance on image segmentation[9].[10] U-Nets downsample an input to create a high-resolution encoding, then upsample to create an output mask, and include feed-forward connections between the downsampling and upsampling path to better localize features.[9] Dense models replace convolutions in CNNs with "dense blocks", which contain multiple convolutions connected by feed-forward operations, improving gradient and feature propagation throughout a network.[11] Dense U-Nets have successfully outperformed U-Nets on removing artifacts from biomedical images.[10]

Equivariance is a symmetric form of invariance: when an input to a shift-equivariant deep learning model is shifted, the output of the model will be equivalently shifted. Although most operations in CNN-based deep learning models are translationally equivariant, many commonly used strided operations, including max pooling, average pooling, and strided convolutions, are subject to aliasing and do not preserve translational equivariance.[12] Translational equivariance can be achieved by replacing strided operations with densely evaluated operations followed by a strided blur, which effectively low-pass filters a signal before sampling, reducing aliasing and improving translational equivariance.[12]

Furthermore, most layers in CNN-based deep learning architectures are not rotationally equivariant. Partial rotational equivariance can be achieved using group equivariant CNNs, which exploit symmetry by generalizing convolution layers to "group convolutions". Although group equivariant CNNs are only partially rotationally equivariant, these models have shown good equivariance to arbitrary rotations in practice.[13] Group convolutions are equivariant to a specific group of transformations, often C_8 (the group of rotations by integer multiples of 45°) or other cyclic groups. Group equivariance generally improves performance and reduces the parameter space of deep learning models. By directly encoding symmetry in the model, the model only has to learn to detect features of interest and does not have to learn symmetry, speeding up training.[14] Group equivariant models are well-suited to domains where images may be arbitrarily rotated about the camera vector, such as satellite imagery.[15]

DATASET, METRICS, & TRAINING

In all three experiments, we train our models on images from the same dataset: a modified version of the Spatial Procedures for Automated Removal of Cloud and Shadow (SPARCS) dataset of 80 Landsat 8 images representing 16 different biomes.[1] We evaluate each model's image segmentation performance qualitatively and quantitatively. For our first experiment, in which we compare the C_8-equivariant dense U-Net to other rule-based and deep learning models, we also evaluate computational resource consumption.

SPARCS Dataset

The SPARCS dataset contains high-accuracy segmentation masks created by human operators, classifying each pixel as cloud, cloud shadow, cloud shadow over water, water, ice/snow, land, or flooded.[1] We post-process the SPARCS dataset to create binary masks where each pixel is classified as cloud or background.

Table 1: Summary of multispectral data available from Landsat 8; bands we use (2-4, 6, 10) are bolded. Adapted from Acharya & Yang.[16]

Band	Name	Wavelength (μm)	Resolution (m)
1	Coastal Aerosol	0.433-0.453	30
2	**Blue**	**0.450-0.515**	**30**
3	**Green**	**0.525-0.600**	**30**
4	**Red**	**0.630-0.680**	**30**
5	Near infrared	0.845-0.885	30
6	**Short-wave infrared (SWIR) 1**	**1.560-1.660**	**30**
7	SWIR 2	2.100-2.300	30
8	Panchromatic	0.500-0.680	15
9	Cirrus	1.360-1.390	30
10	**Long-wave infrared (LWIR) 1**	**10.30-11.30**	**100**
11	LWIR 2	11.50-12.50	100

Landsat 8 has eleven spectral bands (see Table 1), and the SPARCS dataset includes data from Landsat B1-10. We chose to focus on visible-spectrum, long-wave infrared, and short-wave infrared imagery, and so we only use data from Landsat B2-4 (blue, green, and red), B6 (SWIR 1), and B10 (LWIR 1).

Segmentation Metrics

We calculate accuracy, precision, recall, and F_1 score for each model-generated cloud mask. These metrics are given by the following equations, with TP representing the number of true positives, TN representing the number of true negatives, FP representing the number of false positives, and FN representing the number of false negatives:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

$$F_1 \text{ Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

In addition to the classification metrics in Equations 1-4, we plot the receiver operating characteristic (ROC) curve for each model. We also calculate AUC (area under the ROC curve), approximating AUC using midpoint Riemann sums.

Buffered Evaluation

"Buffered evaluation" is a common practice for evaluating segmentation performance when object boundaries are fuzzy or truth masks are inconsistent in quality. When buffered evaluation is used, any pixels within a fixed distance (or buffer) of a boundary between two classes can be classified as either of the classes on the boundary and still be considered correct. Hughes & Kennedy used a 2 pixel buffer on cloud and cloud shadow boundaries when evaluating the original SPARCS CNN to capture the inherent fuzziness of cloud boundaries.[1] Accordingly, we compute and report each classification both without a buffer and with a 2 px buffer at cloud boundaries.

Resource Consumption Metrics

We evaluate the complexity of each model based on the **number of trainable parameters**, the **total number of parameters**, and the **size of the fully trained model**. We also consider the **inference time to classify a single image** and the **peak memory allocated when classifying a single image**, using both a CPU backend (Intel Xeon processor) and GPU backend (Nvidia K80 GPU) via Google Colab.

Implementation and Training

We train our models using the Adam optimizer with the default parameters.[17] We train each model with a learning rate of 0.002 and a batch size of 40 images. The U-Net shows an uncharacteristic spike in test error at 500 epochs, so we train the U-Net and C_8-equivariant U-Net for 505 epochs in order to make fair performance comparisons between models. We train our models using weighted focal loss with $\gamma = 2$ and $\alpha_{\text{cloud}} = 0.8$.[18]
We train our deep learning models on the MIT

Engaging distributed computing cluster.[19] For more information about implementation and training, see Meredith.[20] Our code is freely available at `https://github.com/alexmeredith8299/masters-thesis`.

ALGORITHMS

Luminosity Thresholding

Luminosity thresholding is a simple cloud detection algorithm which classifies each pixel individually. For a multi-band image, one luminosity threshold is found for each band. Each pixel is classified based on its luminosity in each band, resulting in a probabilistic cloud mask with the contribution from each band weighted equally. Figure 1 shows a luminosity thresholding architecture which classifies pixels based on luminosity in the red, green, blue, LWIR, and SWIR bands of an image.

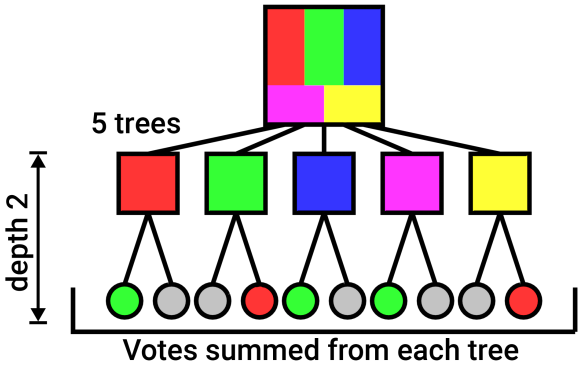


Figure 1: Classification of a single pixel using luminosity thresholding on an image with red, green, blue, LWIR, and SWIR bands.

Random Forest

Random forest classification is an ensemble learning method where multiple trees are trained individually on separate random samples of the training data. During training, each tree solves for the splits that will maximize the decrease in "impurity", which is roughly equivalent to the likelihood of eventual misclassification of a data point.[21] Our random forest is trained using Gini impurity.

As in Wagstaff *et al.*,[7] we use a kernel-based random forest classifier, which considers pixel luminosities in a $k \times k$ kernel centered on the pixel of interest when classifying a pixel. As shown in Figure 2, we use a 3×3 kernel around each pixel.

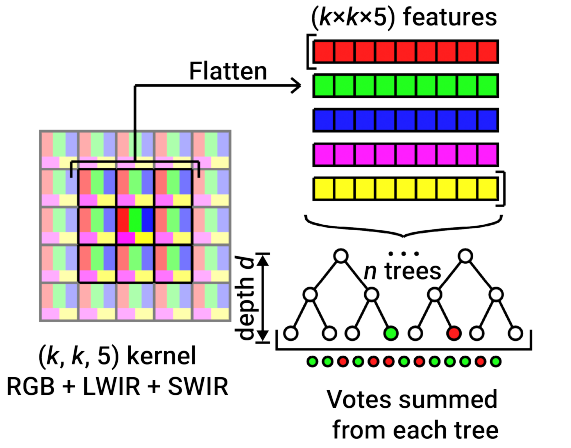


Figure 2: Classification of a single pixel by a kernel-based random forest with n trees of depth d and a $k \times k$ kernel.

U-Net

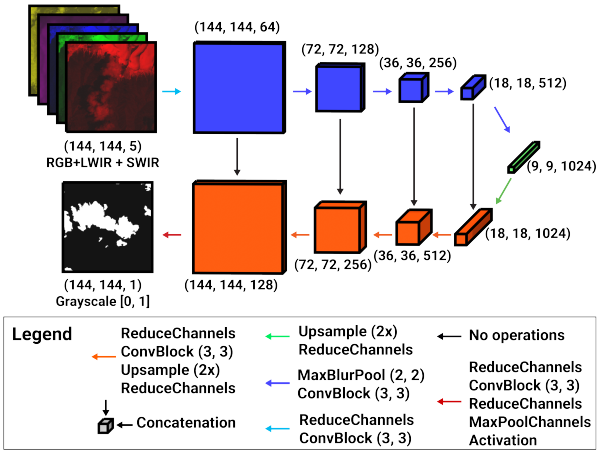


Figure 3: U-Net architecture diagram. See Figure 4 for details of specific operations.

We adapted the original U-Net architecture[9] to work for a 144×144 pixel image rather than a 512×512 pixel image (see Figure 3). We replaced max-pooling operations in the original U-Net with max-blur-pooling, which greatly reduces the aliasing caused by max-pooling and thus improves translational equivariance.[12] We also added batch normalization prior to each rectified linear unit (ReLU) to reduce internal covariate shift and speed up training.[22]

Dense U-Net

Dense deep learning models replace convolution layers with "dense" blocks, which are composed of multiple convolution layers with feed-forward operations

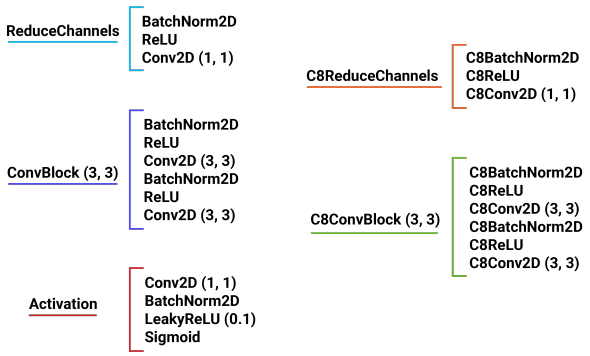


Figure 4: Details of operations used in our deep learning models.

that connect the inputs of each of the convolutions.[11] The four primary design parameters for dense blocks

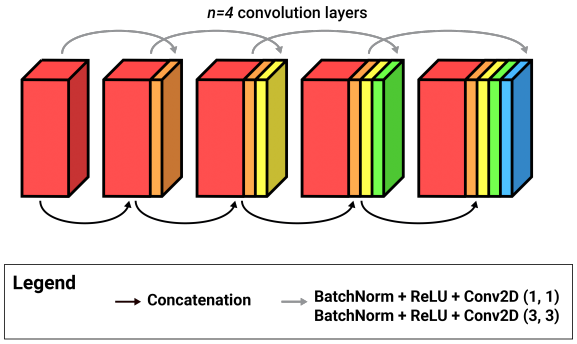


Figure 5: "Dense block" architecture, with $n = 4$ convolution layers and $c_{out} = c_{in}/4$ output channels for the convolutional layers.

are n, the number of convolution layers per dense block, k_1, the size of the kernel used in the first convolution in each layer, k_2, the size of the kernel used in the second convolution in each layer, and c_{out}, the number of output channels of each convolution layer. The output of a dense block with c_{in} input channels has $c_{in} + nc_{out}$ channels. As in the original DenseNet,[11] we use a 1×1 kernel in the first convolution and a 3×3 kernel in the second convolution of each convolution layer. A typical dense block from our dense U-Net is shown in Figure 5, with $n = 4$ convolution layers, $k_1 = 1$, $k_2 = 3$, $c_{out} = c_{in}/4$, where c_{in} is the number of channels in the input to the dense block, and $2c_{in}$ channels in the final dense block output.

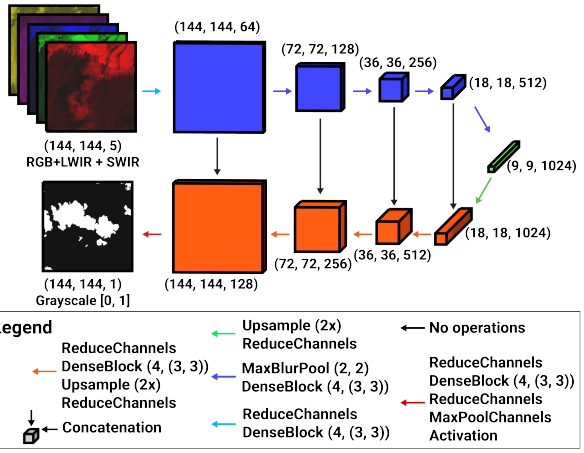


Figure 6: Dense U-Net architecture diagram. See Figures 4 and 5 for details of specific operations.

C_8-*Equivariant U-Net*

The C_8-equivariant U-Net is very similar to the U-Net, but with nearly all operations replaced with C_8-equivariant equivalents. Because the regular representation of C_8 supports pointwise nonlinearities, many operations, including batch normalization and ReLU, can simply be applied to each orientation representation in C_8-equivariant deep learning models while preserving equivariance.[23] C_n-equivariant convolutions (where C_n is the group of $\frac{360°}{n}$ rotations), however, differ fairly significantly from regular convolutions. A C_4-equivariant convolution is shown in Figure 7, along with an "input convolution" that transforms an input image on $\mathbb{R}^2$ to a function on $\mathbb{R}^2 \rtimes C_4$ and an "orientation pooling" layer that takes the orientation-wise maximum to generate a final output map on $\mathbb{R}^2$. See Meredith[20] for a more detailed tutorial on C_n-equivariant convolutions.

The main exceptions to the idea that the C_8-

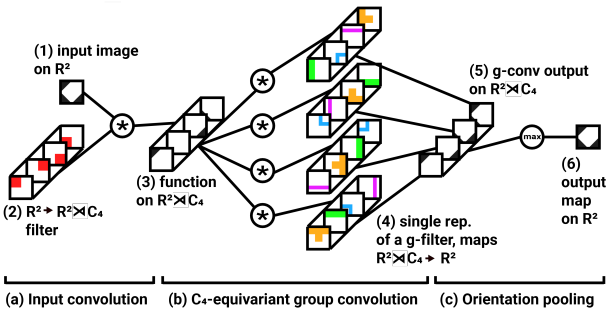


Figure 7: Input convolution, group convolution, and orientation pooling in a C_4-equivariant network.

equivariant U-Net is just a U-Net with C_8-equivariant operations occur in the first "input" and final "output" convolution steps. The input convolution step involves a non-equivariant batch normalization and ReLU followed by a convolution that "lifts" the input from an image on $\mathbb{R}^2$ to a feature map on $\mathbb{R}^2 \rtimes C_8$. The output convolution step involves "orientation pooling", which projects a feature map on $\mathbb{R}^2 \rtimes C_8$ back to $\mathbb{R}^2$ by taking the pixel-wise maximum or average across the orientation dimension, followed by the same activation block used in the U-Net.

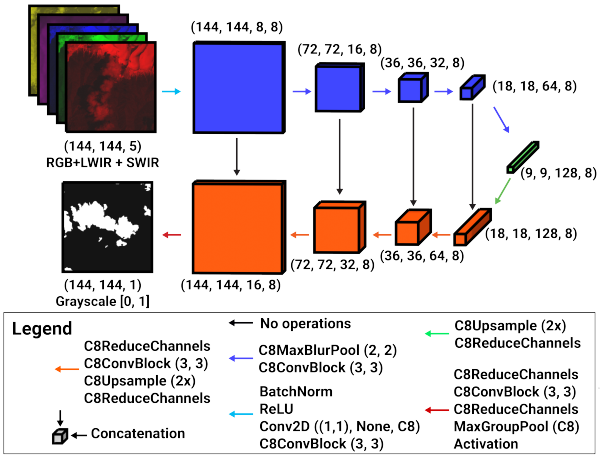


Figure 8: C_8-equivariant U-Net architecture. At each layer, (x, y, c, n) represents the image dimensions $(x \times y)$, number of channels per orientation c, and number of orientations n, with $n = 8$ representing C_8-equivariance. See Figure 4 for details of specific operations.

C_8-Equivariant Dense U-Net

The C_8-equivariant dense U-Net has the same architecture as the C_8-equivariant U-Net (see Figures 8 and 9), except for the use of C_8-equivariant dense blocks in place of C_8-equivariant convolution blocks. The C_8-equivariant dense U-Net is able to leverage the natural rotational symmetry of satellite imagery and benefit from the improved gradient propagation of dense deep learning networks.[24]

ALGORITHM STUDY

Qualitative Results

We apply each model to an "easy" cloud segmentation example with no snow, ice, cold water, or bright land pixels, and to a "hard" image with overlapping snow and cloud (see Figure 10). The luminosity thresholding and random forest algorithms perform much better on the "easy" image than on the "hard" image, but the deep learning algorithms perform well on both images, highlighting the limitations of rule-based algorithms that only consider limited spatial context.

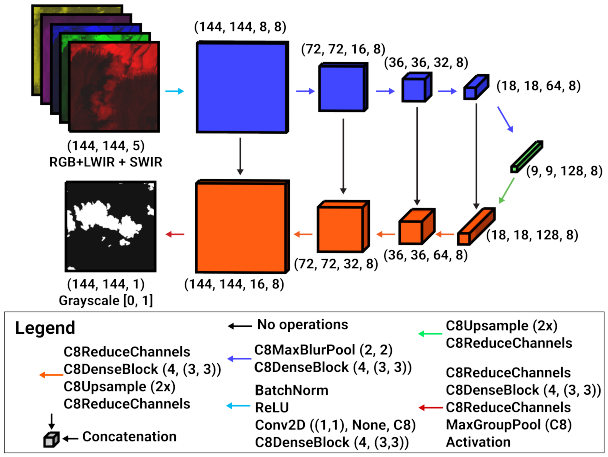


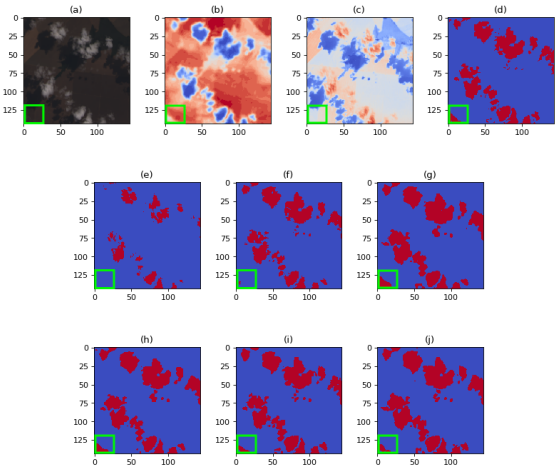
Figure 9: C_8-equivariant dense U-Net architecture. At each layer, the (x, y, c, n) tuple represents the image dimensions $(x \times y)$, the number of channels per orientation c, and the number of orientations n.

However, the luminosity thresholding and random forest algorithms still miss the optically thin cloud patch boxed in green in the "easy" image. This cloud patch is difficult to see in the SWIR input and invisible in the VIS and LWIR inputs. Still, all four deep learning algorithms capture this optically thin cloud patch to some degree. The two C_8-equivariant models classify the optically thin cloud most accurately, while the other deep learning models overestimate the size of this cloud patch.

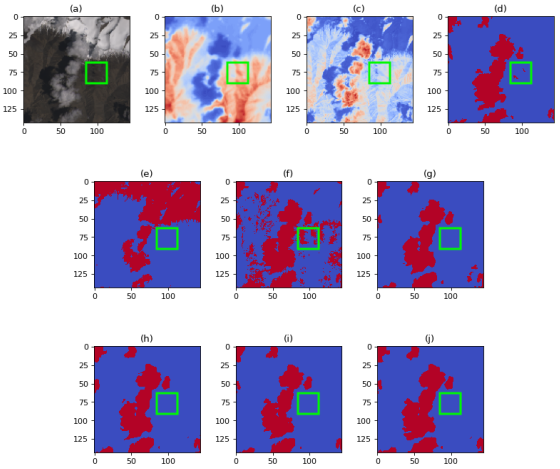
In the "hard" image, no models correctly classify the optically thin cloud patch boxed in green. This patch is not bright in either the VIS or SWIR image input, nor is it cold in the LWIR image input. As such, this cloud patch, which is only perceptible in Landsat B8 and B9, is essentially invisible to the cloud segmentation models.

Quantitative Results

The performance metrics for each of our models, evaluated without a buffer, are given in Table 2, with the best performance on each metric bolded. Figure 11 shows the receiver-operating characteristic (ROC) curve for each of our models, evaluated without a buffer. The curve representing the C_8-equivariant dense U-Net is consistently closest to the $(0, 1)$ point



(i) Evaluation on "easy" image, with optically thin cloud boxed in green.



(ii) Evaluation on "hard" image, with optically thin cloud boxed in green.

Figure 10: Models evaluated on two images. Subfigures include (a) VIS input, (b) LWIR input, (c) SWIR input, (d) "Truth" mask, (e) Luminosity thresholding output, (f) Random forest output, (g) U-Net output, (h) Dense U-Net output, (i) C_8-equivariant U-Net output, (j) C_8-equivariant dense U-Net output.

representing perfect classification, demonstrating its superior performance. The next closest curve represents the dense U-Net, then the C_8-equivariant U-Net, then the U-Net, then the random forest algorithm, and finally the curve furthest from $(0, 1)$ represents the luminosity thresholding algorithm. This order is consistent with the order of the F_1 scores presented in Table 2, except that the ROC curve representing the dense U-Net is closer to $(0, 1)$ than the curve representing the C_8-equivariant U-Net. So, with a cloud threshold other than 0.5, the dense U-Net would likely outperform the C_8-equivariant U-Net.

Table 2: Performance metrics, evaluated with no buffer at cloud boundaries.

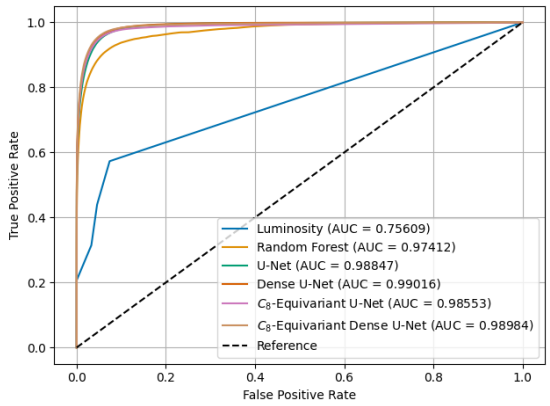
Model	Acc.	Prec.	Recall	$\mathbf{F}_1$
Luminosity Thresholding	89.19%	55.40%	31.45%	0.4013
Random Forest	95.60%	79.16%	83.80%	0.8141
U-Net	96.59%	84.88%	85.62%	0.8525
Dense U-Net	96.77%	84.31%	**88.36%**	0.8629
C_8-Eq. U-Net	96.87%	**87.01%**	85.64%	0.8632
C_8-Eq. Dense U-Net	**97.01%**	86.15%	88.17%	**0.8715**

The performance metrics for all models (with a 2 px buffer) are given in Table 3, with the best performance on each metric bolded. The F_1 scores for the four deep learning algorithms are notably closer with a 2 px buffer at cloud boundaries than with no buffer. So, some of the performance gains made by the C_8-equivariant dense U-Net can be attributed to better discrimination between cloud and non-cloud pixels at or within 2 pixels of cloud boundaries. Still, the C_8-equivariant dense U-Net demonstrates the best cloud segmentation performance even when a 2 px buffer is used, outperforming all other models on every quantitative metric evaluated.

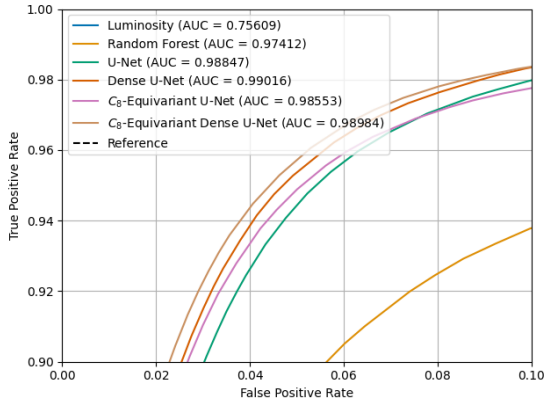
Figure 12 shows the ROC curves for each model when evaluated with a 2 px buffer at the cloud boundaries. The curves representing the deep learning models are clustered much more closely than when no buffer is considered. Still, the AUC for each model reflects the same relative performance seen in Figure 11. Although the C_8-equivariant dense U-Net and the dense U-Net are tied for the highest AUC, the C_8-equivariant dense U-Net outperforms the dense U-Net except for a small region where sensitivity is high and specificity is low, and gets closer to $(0, 1)$ than any other classifier, reflecting its superior performance.

Resource Consumption

Table 4 shows the resource utilization of all models when classifying a single image. As expected, the



(i) Original view.



(ii) Zoomed-in view.

Figure 11: Receiver-operating characteristic (ROC) curves for all models, evaluated with no buffer at cloud boundaries.

Table 3: Performance metrics for all models, evaluated using a 2 px buffer at cloud boundaries.

Model	Acc.	Prec.	Recall	F_1
Luminosity Thresholding	92.22%	60.20%	43.18%	0.5029
Random Forest	98.68%	92.11%	96.91%	0.9445
U-Net	99.41%	98.46%	96.54%	0.9749
Dense U-Net	99.43%	98.47%	96.89%	0.9768
C_8-Eq. U-Net	**99.54%**	98.78%	97.18%	0.9797
C_8-Eq. Dense U-Net	**99.54%**	**98.82%**	**97.32%**	**0.9806**

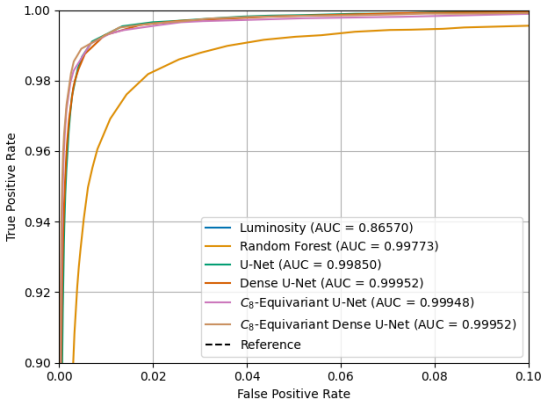


Figure 12: ROC curves for all models, evaluated with a 2 px buffer at cloud boundaries (zoomed-in view).

luminosity thresholding algorithm is fastest with a CPU backend, followed by the random forest algorithm. The dense deep learning models are faster than their non-dense equivalents with a CPU backend, but slower and more memory-intensive with a GPU backend. This is likely because the dense models have fewer parameters but more concatenations, reducing the number of opportunities to combine tensor operations when using a GPU.

Finally, the C_8-equivariant models classify images more slowly than their non-equivariant equivalents, despite using less GPU memory and having fewer parameters overall. These models are 2 to 3 times slower than their non-equivariant equivalents with a GPU backend, and are 1.1-1.3 times slower with a CPU backend. This may be due to overhead related to checking group representations.

Table 5 shows the size and number of parameters for different models. The random forest algorithm requires the most storage by far – this is one typical and major drawback of data-driven methods. Also, the C_8-equivariant dense U-Net has fewer parameters and a smaller total model size than all the other deep learning models.

SPECTRAL BAND STUDY

Qualitative Results

Figure 13 evaluates the C_8-equivariant dense U-Net trained on four different combinations of Landsat 8 bands on an "easy" image with no snow, cold water, or bright non-cloud pixels and on a "hard" image with overlapping snow and cloud. For the "easy" image, all four model-generated masks look very similar to the "truth" mask, except for slight differences

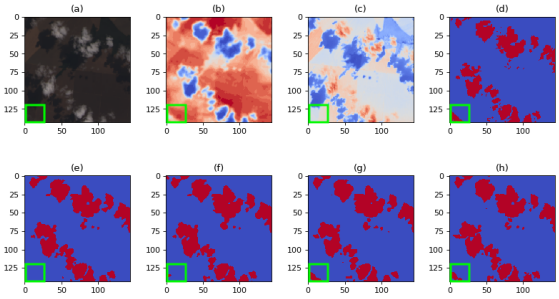
Table 4: Peak memory usage and average inference time for single-image classification.

Model	GPU Mem. (MiB)	CPU Mem. (KiB)	GPU Time (s)	CPU Time (s)
Luminosity Thresholding	–	344.9	–	0.0003
Random Forest	–	1445.8	–	0.1136
U-Net	464.9	127.0	0.0092	0.3698
Dense U-Net	517.4	103.6	0.0150	0.1801
C_8-Eq. U-Net	448.3	101.2	0.0185	0.3926
C_8-Eq. Dense U-Net	503.5	144.8	0.0484	0.2281

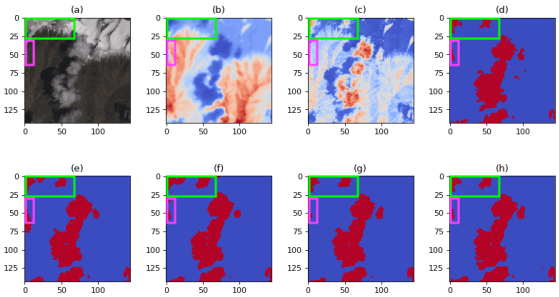
Table 5: Saved model size and complexity.

Model	Model Size	Total Params	Trainable Params
Luminosity Thresholding	4 KB	–	–
Random Forest	1.3 GB	–	–
U-Net	295.9 MB	2.46×10^7	2.46×10^7
Dense U-Net	32.2 MB	2.64×10^6	2.64×10^6
C_8-Eq. U-Net	123.9 MB	2.10×10^6	2.10×10^6
C_8-Eq. Dense U-Net	14.6 MB	2.93×10^5	2.90×10^5

identifying the optically thin clouds (boxed in green). For the "hard" image, the model trained on only VIS imagery and the model trained on VIS and LWIR imagery have difficulty distinguishing snow and cloud in the region boxed in green. The model trained on VIS imagery only also has trouble correctly classifying optically thin cloud boxed in pink, as does the model trained on VIS and SWIR data. The model trained on VIS, LWIR, and SWIR data is able to correctly classify clouds in both the green-boxed region and the pink-boxed region. These results qualitatively demonstrate the challenges distinguishing snow and clouds without SWIR data, and the difficulty identifying optically thin clouds without LWIR or cirrus-band data.



(i) Evaluation on "easy" image, with optically thin cloud boxed in green.



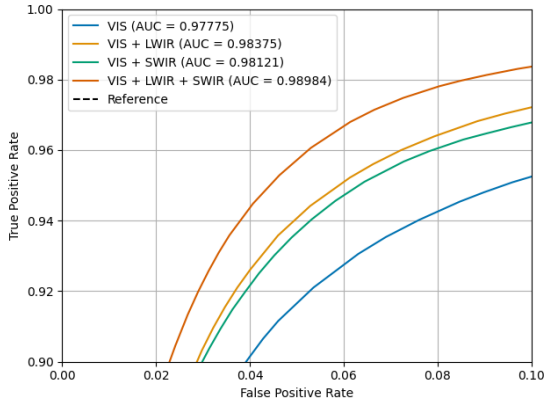
(ii) Evaluation on "hard" image, with cloud patch over snow boxed in green and optically thin cloud boxed in pink.

Figure 13: C_8-equivariant dense U-Net evaluated on two images. Subfigures include (a) VIS input, (b) LWIR input, (c) SWIR input, (d) "Truth" mask, (e) VIS-trained model output, (f) VIS+LWIR-trained model output, (g) VIS+SWIR-trained model output, (h) VIS+LWIR+SWIR-trained model output.

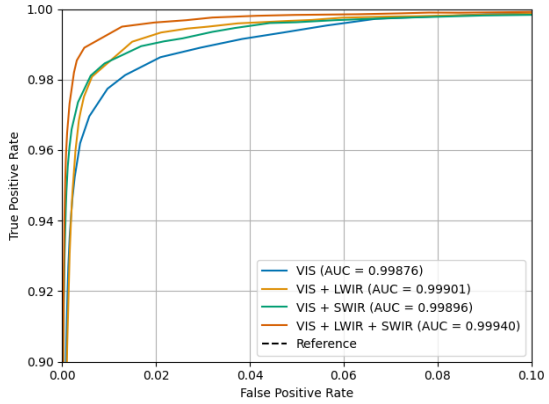
Quantitative Results

Table 6 summarizes the results of the C_8-equivariant dense U-Net trained on four different combinations of bands. Figure 14 plots the receiver-operating characteristic (ROC) curve for the C_8-equivariant dense U-Net trained on four different combinations of bands. The ROC curve for the model trained on VIS, LWIR, and SWIR bands consistently has the highest true positive rate, and the ROC curve for the model trained on VIS data has the lowest true positive rate, with the false positive rate held constant. Without a buffer, the model trained on VIS

and LWIR data consistently has a slightly higher true positive rate than the model trained on VIS SWIR data, with the false positive rate held constant. However, with a 2 px buffer at the cloud boundaries, the curves representing the model trained on VIS and LWIR data and the model trained on VIS and SWIR data cross. The model trained on VIS and SWIR data has a higher true positive rate when the false positive rate is low (specificity is high), and the model trained on VIS and LWIR data has a higher true positive rate when sensitivity is more important than specificity.



(i) With no buffer at cloud boundaries.



(ii) With 2 px buffer at cloud boundaries.

Figure 14: ROC curves for the C_8-equivariant dense U-Net trained on different bands.

COMPARISON TO LITERATURE

The SPARCS dataset was originally created for the SPARCS CNN.[1] The SPARCS CNN, unlike our models, relies on a ten-band input and predicts several different terrain classes. Additionally, our cloud detection models classify 144 x 144 pixel patches of input images and the SPARCS CNN classifies 256 x 256 pixel patches.[1] Nevertheless, we use the same dataset, so we can compare the different models. Table 7 summarizes our results alongside the results found by Hughes and Kennedy.[1] Only metrics calculated with a 2 px buffer at cloud boundaries are available for the SPARCS CNN, so we present all metrics using a 2 px buffer at cloud boundaries.

Table 6: Performance metrics for the C_8-equivariant dense U-Net trained on different bands.

(i) With no buffer at cloud boundaries.

Model	Acc.	Prec.	Recall	F_1
VIS	96.53%	86.86%	82.29%	0.8451
VIS + LWIR	96.76%	86.65%	84.91%	0.8577
VIS + SWIR	96.89%	**90.10%**	82.04%	0.8588
VIS + LWIR + SWIR	**97.01%**	86.15%	**88.17%**	**0.8715**

(ii) With 2 px buffer at cloud boundaries.

Model	Acc.	Prec.	Recall	F_1
VIS	99.11%	98.70%	93.54%	0.9605
VIS + LWIR	99.29%	97.78%	95.98%	0.9687
VIS + SWIR	99.21%	**99.50%**	93.39%	0.9635
VIS + LWIR + SWIR	**99.54%**	98.82%	**97.32%**	**0.9806**

As shown in Table 7, the SPARCS CNN has a higher F_1 score than our luminosity thresholding algorithm, random forest algorithm, and C_8-equivariant dense U-Net trained on VIS data, but a lower F_1 score than all of our other models. This may be because SPARCS CNN is designed to distinguish clear-sky, water, snow/ice, cloud, and cloud shadow classes, while our models only perform cloud segmentation. However, the SPARCS CNN has significantly more parameters than most of our models, and should have enough parameters for multi-class segmentation.

It is more likely that most of our deep learning models outperform the SPARCS CNN because of differences in architecture. The SPARCS CNN uses max pooling, which leads to aliasing and inhibit translation equivariance[12].[1] Our models use blur convolutions before max pooling and thus are less susceptible to aliasing.[12] Additionally, our C_8-equivariant and dense models make further performance gains by leveraging rotational equivariance and the improved gradient flow through dense networks[24].[23]

Table 7: Performance metrics calculated for the SPARCS CNN[1] and for our models, using a 2 px buffer at the cloud boundaries.

Model	Prec.	Rec.	$\mathbf{F}_1$	Trainable Params
SPARCS CNN[1]	95.73%	96.42%	0.9607	2.05×10^7
Luminosity Thresholding	60.20%	43.18%	0.5029	–
Random Forest	92.11%	96.91%	0.9445	–
U-Net	98.46%	96.54%	0.9749	2.46×10^7
Dense U-Net	98.47%	96.89%	0.9768	2.64×10^6
C_8-Eq. U-Net	98.78%	97.18%	0.9797	2.10×10^6
C_8-Eq. Dense U-Net	98.82%	97.32%	0.9806	2.90×10^5
C_8-Eq. Dense U-Net (VIS-only)	98.70%	93.54%	0.9605	2.90×10^5
C_8-Eq. Dense U-Net (VIS + LWIR)	97.78%	95.98%	0.9687	2.90×10^5
C_8-Eq. Dense U-Net (VIS + SWIR)	99.50%	93.39%	0.9635	2.90×10^5

CONCLUSION

We present four different machine learning algorithms, including two C_8-equivariant machine learning models, and two rule-based algorithms, all for identifying clouds in satellite imagery. We also evaluate the performance of the C_8-equivariant dense U-Net on cloud segmentation when trained on visible-spectrum (VIS) data only, VIS and LWIR data, VIS and SWIR data, and VIS, LWIR and SWIR data.
Of all six models, the C_8-equivariant dense U-Net produces the most accurate segmentation maps, achieving an F_1 score of 0.9806 and accuracy of 99.54% on the SPARCS dataset when evaluated with a 2 px buffer at the cloud boundaries, outperforming human operators (96% self-consistency) as well as the state-of-the-art SPARCS CNN.[1] The C_8-equivariant dense U-Net also has only around 290,000 trainable parameters – the fewest of our deep learning models. The C_8-equivariant dense U-Net is a strong fit for deployment on resource-constrained platforms. It takes only 0.2281 seconds to classify an image using a CPU backend, and requires under 15 MB of memory, making it a good fit for missions without onboard GPUs. When trained on visible-spectrum data only it has an F_1 score of 0.9605, when trained on VIS and LWIR data it achieves an F_1 score of 0.9687, and when trained on VIS and SWIR data it achieves an F_1 score of 0.9635, demonstrating suitability for missions without multispectral instruments.

FUTURE WORK & APPLICATIONS

Our models were trained on a high-performance computing cluster and evaluated on a laptop; these models are not yet ready to be deployed to real-time operating systems (RTOS) and flight hardware. In preparation for deployment, we plan to translate our models to TorchScript, which can be run directly from C++ on embedded hardware. Also, before deployment, we plan to create a new dataset with images collected directly from target regions and to retrain the models on the mission-specific dataset. This will help optimize model performance and is especially important for identifying clouds over water; because our training data is from Landsat, all training images were taken over or near land.
A future application of rotation-equivariant machine learning is learning on time-series data in order to track the movement of clouds over time. Another application is learning cloud morphology. If these two applications are combined, the ability to identify the movement and morphology of clouds would be useful for near-term weather prediction.

ACKNOWLEDGMENTS

This work was supported by the NSF Graduate Research Fellowship Program under Grant No. 2141064. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] M Joseph Hughes and Robert Kennedy. High-quality cloud masking of landsat 8 imagery using convolutional neural networks. *Remote Sensing*, 11(21):2591, 2019.

[2] Zhe Zhu, Shixiong Wang, and Curtis E Woodcock. Improvement and expansion of the fmask algorithm: Cloud, cloud shadow, and snow detection for landsats 4–7, 8, and sentinel 2 images. *Remote sensing of Environment*, 159:269–277, 2015.

[3] Richard A Frey, Steven A Ackerman, Robert E Holz, Steven Dutcher, and Zach Griffith. The continuity modis-viirs cloud mask. *Remote Sensing*, 12(20):3334, 2020.

[4] Qing Z Trepte, Patrick Minnis, Szedung Sun-Mack, Christopher R Yost, Yan Chen, Zhonghai Jin, Gang Hong, Fu-Lung Chang, William L Smith, Kristopher M Bedka, et al. Global cloud detection for ceres edition 4 using terra and aqua modis data. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):9410–9449, 2019.

[5] Andrew Heidinger and William C. Straka III. Abi cloud mask version 3.0, 2012.

[6] Anze Zupanc. Improving cloud detection with machine learning, 2017.

[7] Kiri L Wagstaff, Alphan Altinok, Steve A Chien, Umaa Rebbapragada, Steve R Schaffer, David R Thompson, and Daniel Q Tran. Cloud filtering and novelty detection using onboard machine learning for the eo-1 spacecraft. In *Proc. IJCAI Workshop AI in the Oceans and Space*, 2017.

[8] Shreeyam Kacker, Alex Meredith, Kerri Cahoy, and Georges Labreche. Machine learning image processing algorithms onboard ops-sat. In *2022 Small Satellite Conference*, 2022.

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[10] Steven Guan, Amir A Khan, Siddhartha Sikdar, and Parag V Chitnis. Fully dense unet for 2-d sparse photoacoustic tomography artifact removal. *IEEE journal of biomedical and health informatics*, 24(2):568–576, 2019.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[12] Richard Zhang. Making convolutional networks shift-invariant again. In *International conference on machine learning*, pages 7324–7334. PMLR, 2019.

[13] Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 849–858, 2018.

[14] Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

[15] Simon Graham, David Epstein, and Nasir Rajpoot. Dense steerable filter cnns for exploiting rotational symmetry in histology images. *IEEE Transactions on Medical Imaging*, 39(12):4124–4136, 2020.

[16] Tri Dev Acharya and Intae Yang. Exploring landsat 8. *International Journal of IT, Engineering and Applied Sciences Research (IJIEASR)*, 4(4):4–10, 2015.

[17] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[18] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

[19] Engaging cluster documentation, 2023.

[20] Alex Meredith. Applying rotation-equivariant deep learning to cloud and road segmentation in satellite and aerial imagery. Master's thesis, Massachusetts Institute of Technology, 2023.

[21] Gilles Louppe. Understanding random forests: From theory to practice. *arXiv preprint arXiv:1407.7502*, 2014.

[22] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.

[23] Maurice Weiler and Gabriele Cesa. General e (2)-equivariant steerable cnns. *Advances in Neural Information Processing Systems*, 32, 2019.

[24] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *Advances in neural information processing systems*, 31, 2018.