Visualizing and Interpreting Neural Networks

Clémence Réda & Xiaoqi Xu

January 16^{th}, 2019

Plan

Introduction

Motivation

Convolutional Neural Networks (CNNs)

- Powerful tools for tasks in Computer Vision [Simonyan and Zisserman, 2014] or NLP [Zhang and LeCun, 2015]
- Theory is poorly understood, and CNNs can be *stupidly* easily fooled [Nguyen et al., 2015]

Objectives

1. **Look at what the internal layers of convolutional neural networks encode by visualizing the feature maps**
2. **Try to infer how CNNs work**

Method

We use the DeconvNet model suggested by [Zeiler and Fergus, 2014]:

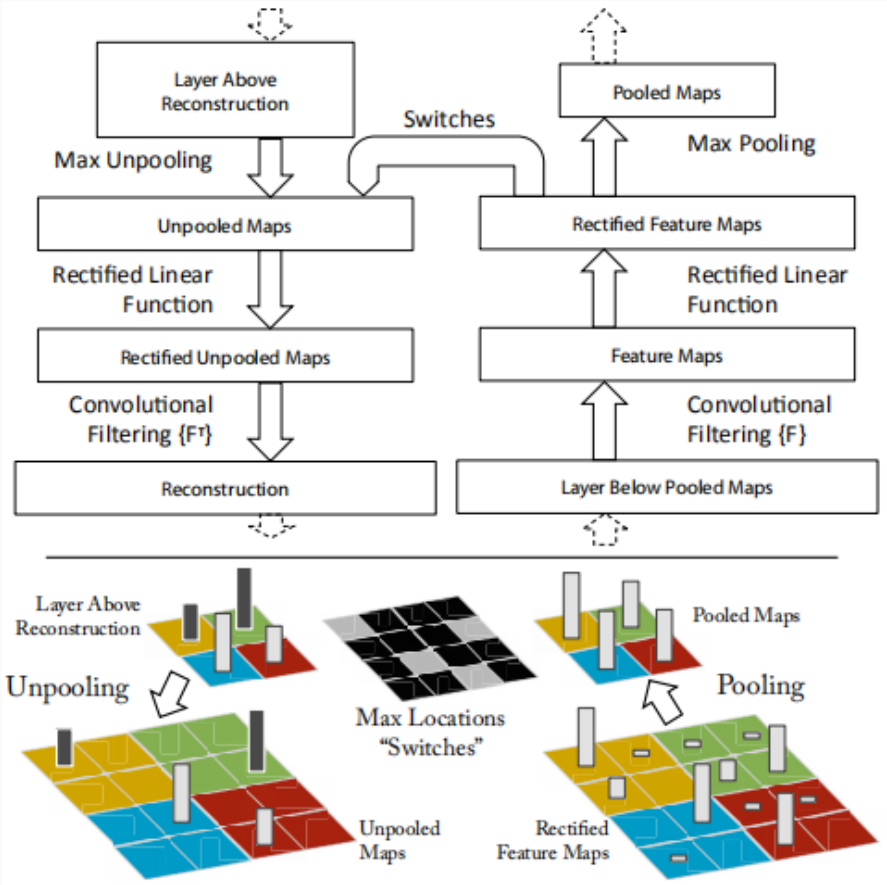
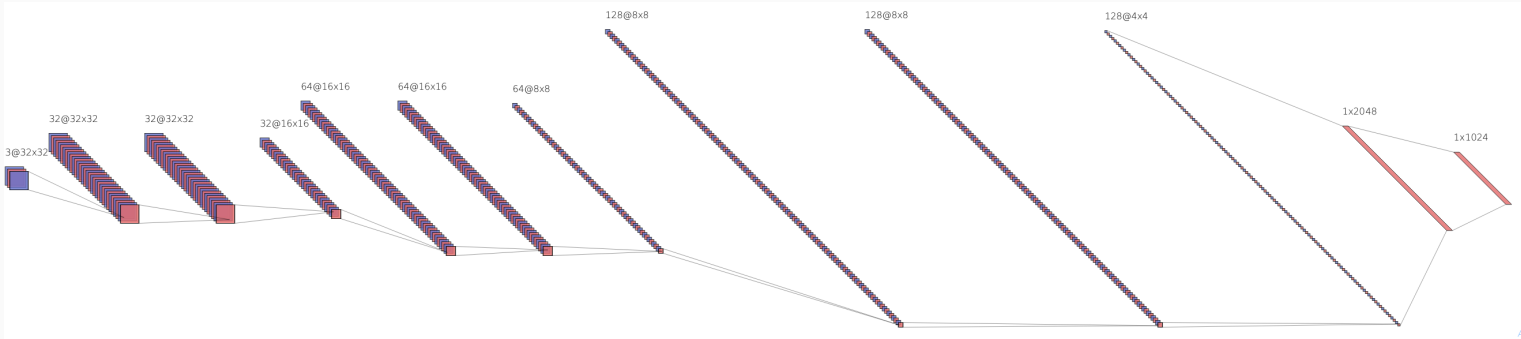


Figure 1: Idea behind DeconvNet ([Zeiler and Fergus, 2014])

Our Work

Construction and training of CNNs

We have trained our own CNNs on CIFAR-10.



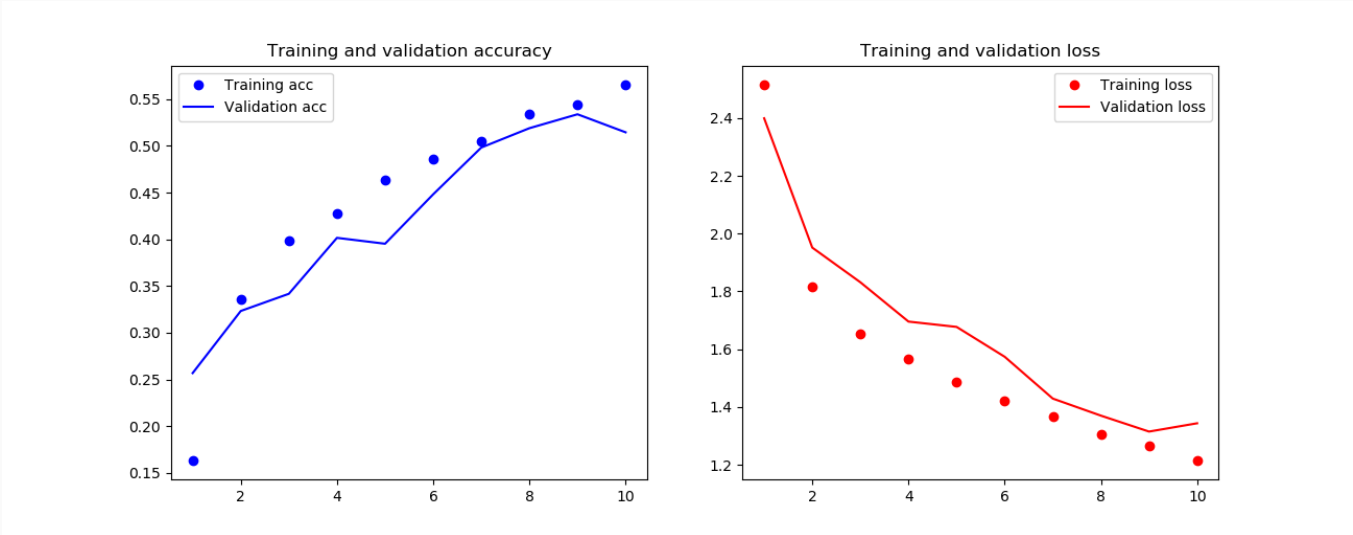


Figure 2: Architecture of [Plotka, 2017] and training loss/accuracy

#epochs=10, lr=10^{-4}, Adam optimizer, batch size = 128

Evaluation of CNNs

	Training	Validation
Accuracy	0.566	0.515
Loss	1.213	1.344

Table 1: Final training accuracy and loss on CIFAR-10 [Krizhevsky and Hinton, 2009] for [Plotka, 2017], #epochs=10, lr=10^{-4}, Adam optimizer, batch size = 128

	Training	Validation
Accuracy	0.818	0.758
Loss	0.517	0.707

Table 2: Final training accuracy and loss on CIFAR-10 [Krizhevsky and Hinton, 2009] for [Bonaccorso, 2017], #epochs=250, lr=10^{-4}, Adam optimizer, batch size = 128

Evaluation of CNNs

State-of-the-art classification accuracy on CIFAR-10:
96.53% [Graham, 2014]

These are not good results.
But model fine-tuning is not the primary goal of this project.

Visualizing outputs of layers

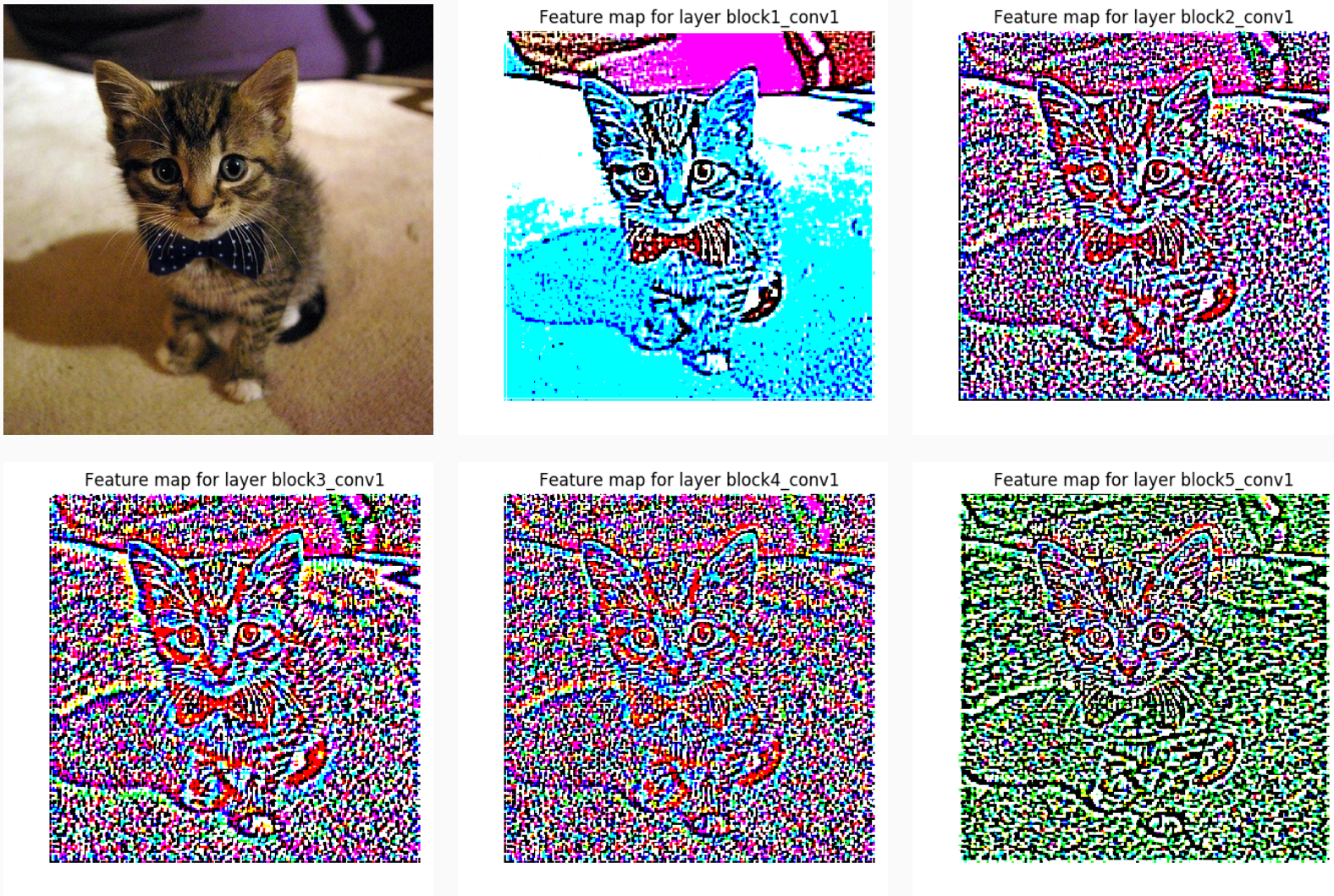


Figure 3: Input and feature map images of VGG for some convolutional layers: output of layer in initial model is deconvoluted.

Visualizing outputs of layers

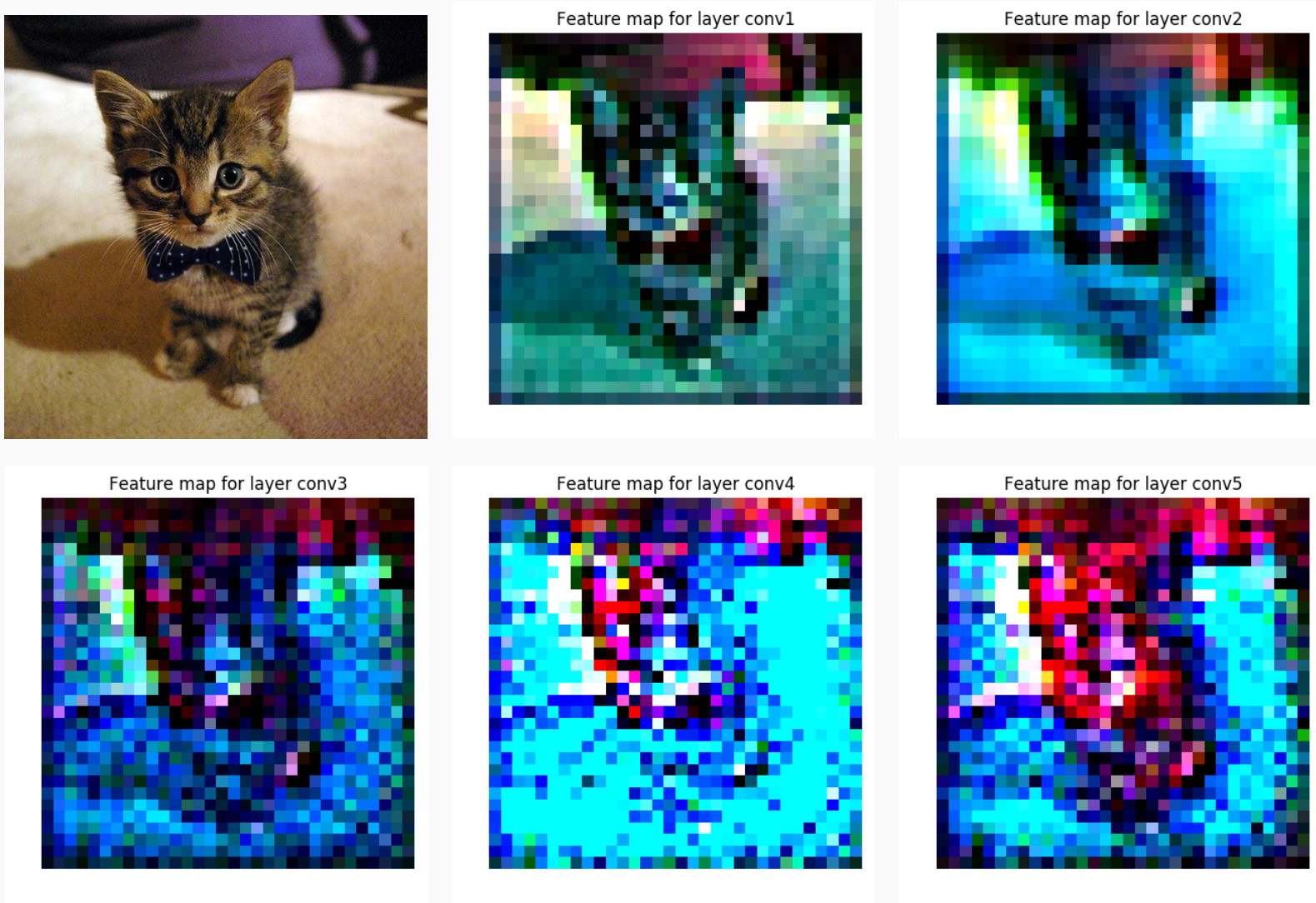


Figure 4: Input and feature map images of [Plotka, 2017] for some convolutional layers: output of layer in initial model is deconvoluted.

Reconstruction by maximizing the activation of a certain class

Algorithm 1 [Chollet, 2016]

1: Start from a random noisy image.
2: Define a loss function that seeks to maximize the activation of a certain class output of the network.
3: Use gradient descent to “twist” the image.
4: **return** the image which maximizes the activation of the *softmax* layer output associated with the considered class

Experiment

1. Gather images of one class
   here, *Siamese cat, Siamese* (284 in ImageNet).
2. Save $n = 5$ reconstructed inputs from trained model.
3. Retrain the model with the (unseen) images in the class.
4. Save $n = 5$ reconstructed inputs from model with new weights.

Reconstruction by maximizing the activation of a certain class

Let's see what the machines have learned !

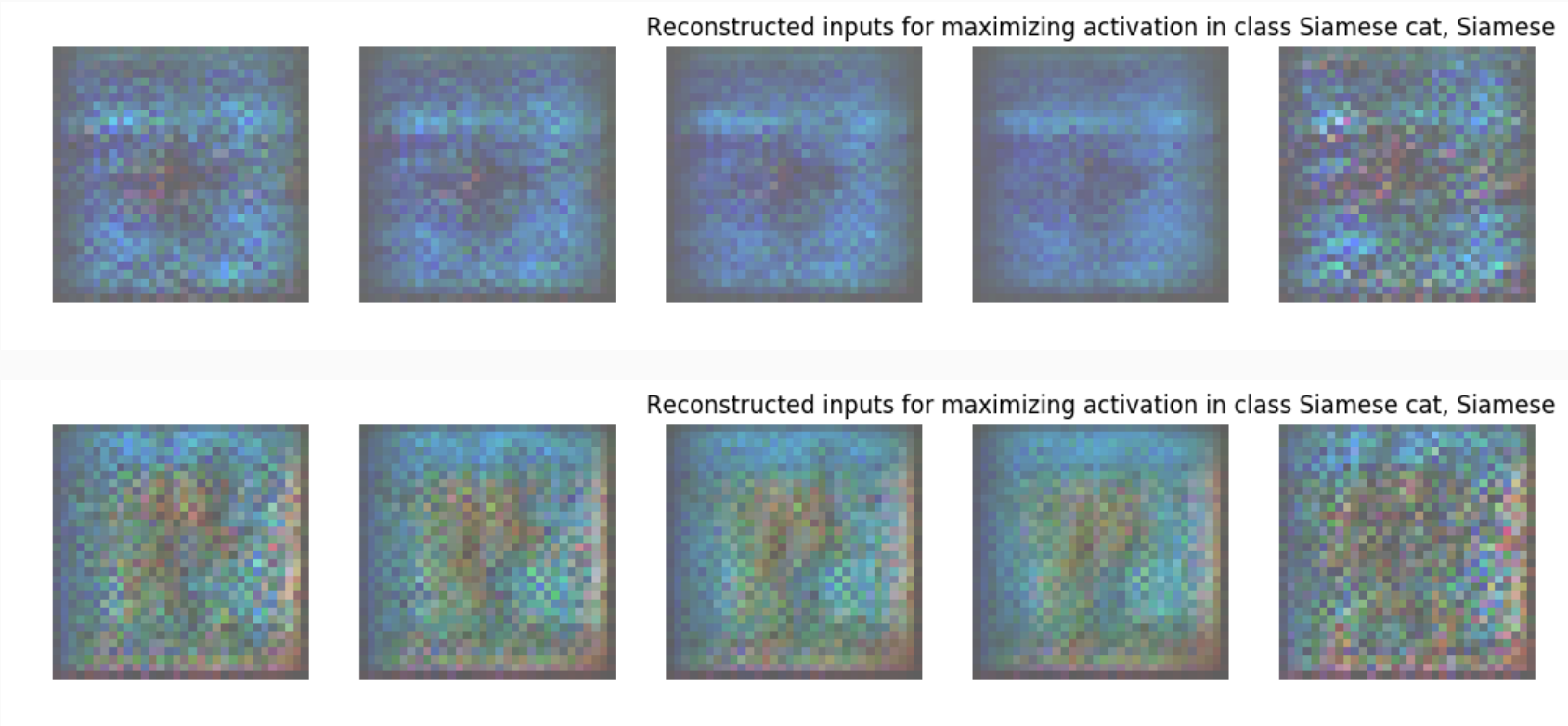


Figure 5: Feature maps before (*top line*) and after training, model [Plotka, 2017], gradient step = 10^{-4}, batch size 32, lr=10^{-3}, Adam optimizer, #epochs=10

Interpretation of Neural Networks

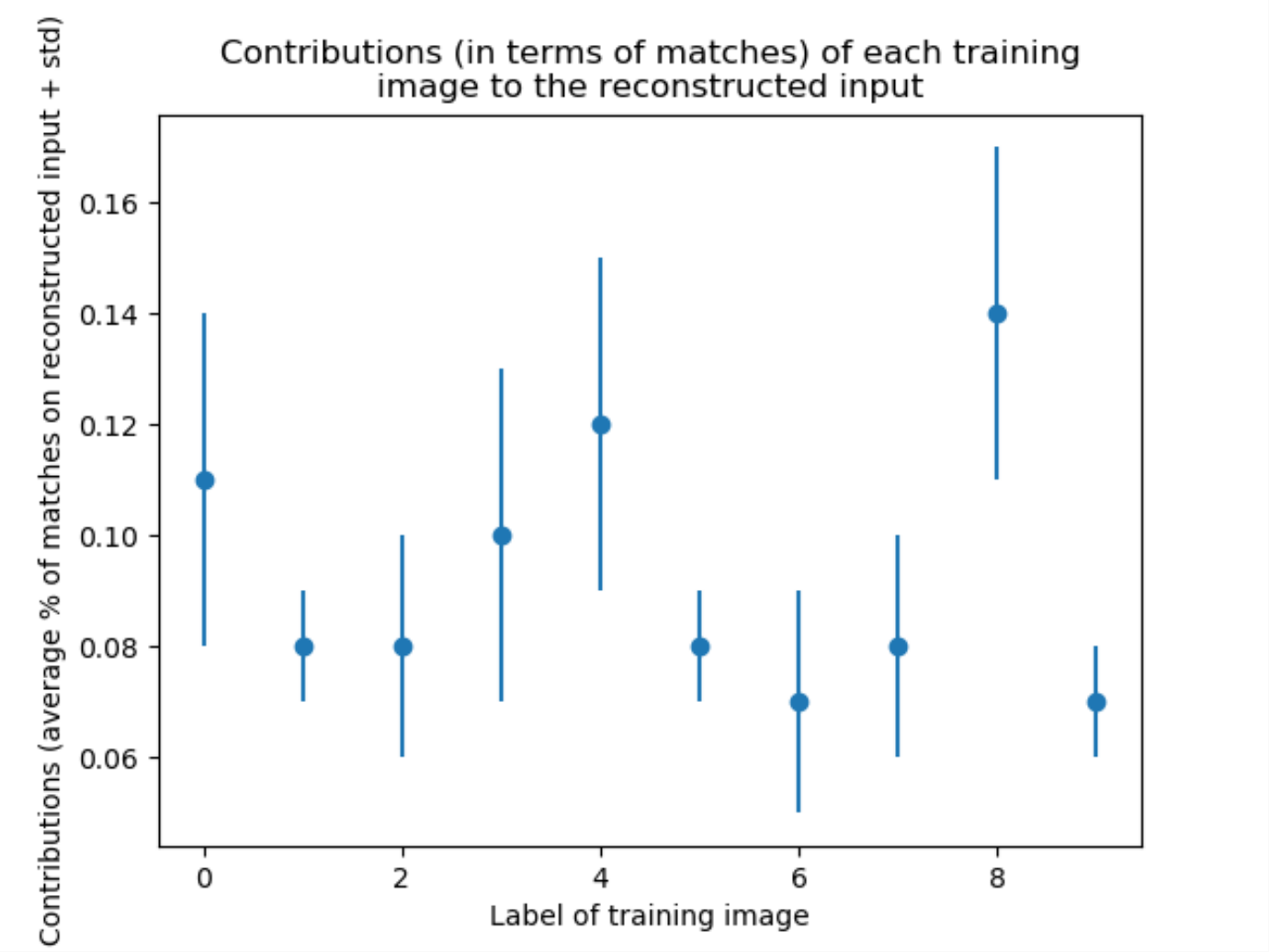
How to quantify the results obtained above?

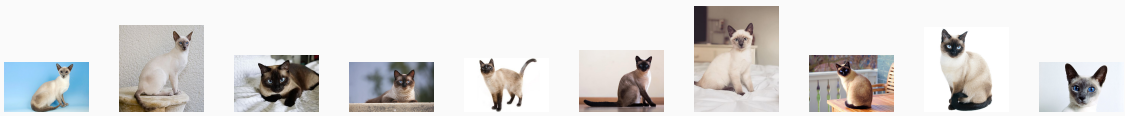
$\rightarrow$ try to describe them in terms of **descriptors**

- SIFT [Lowe, 1999] (*)
- Harris corner descriptor [Harris and Stephens, 1988] (*)
- Bag-of-Words [Sivic and Zisserman, 2009] (**)

(*) **Goal**: estimate the contributions of each training image to the reconstructed input
(**) **Goal**: see if the reconstructed input gets any ”closer” to the original class after retraining

Estimate the contributions of each training image





Reconstructed input ”closer” to the original class?

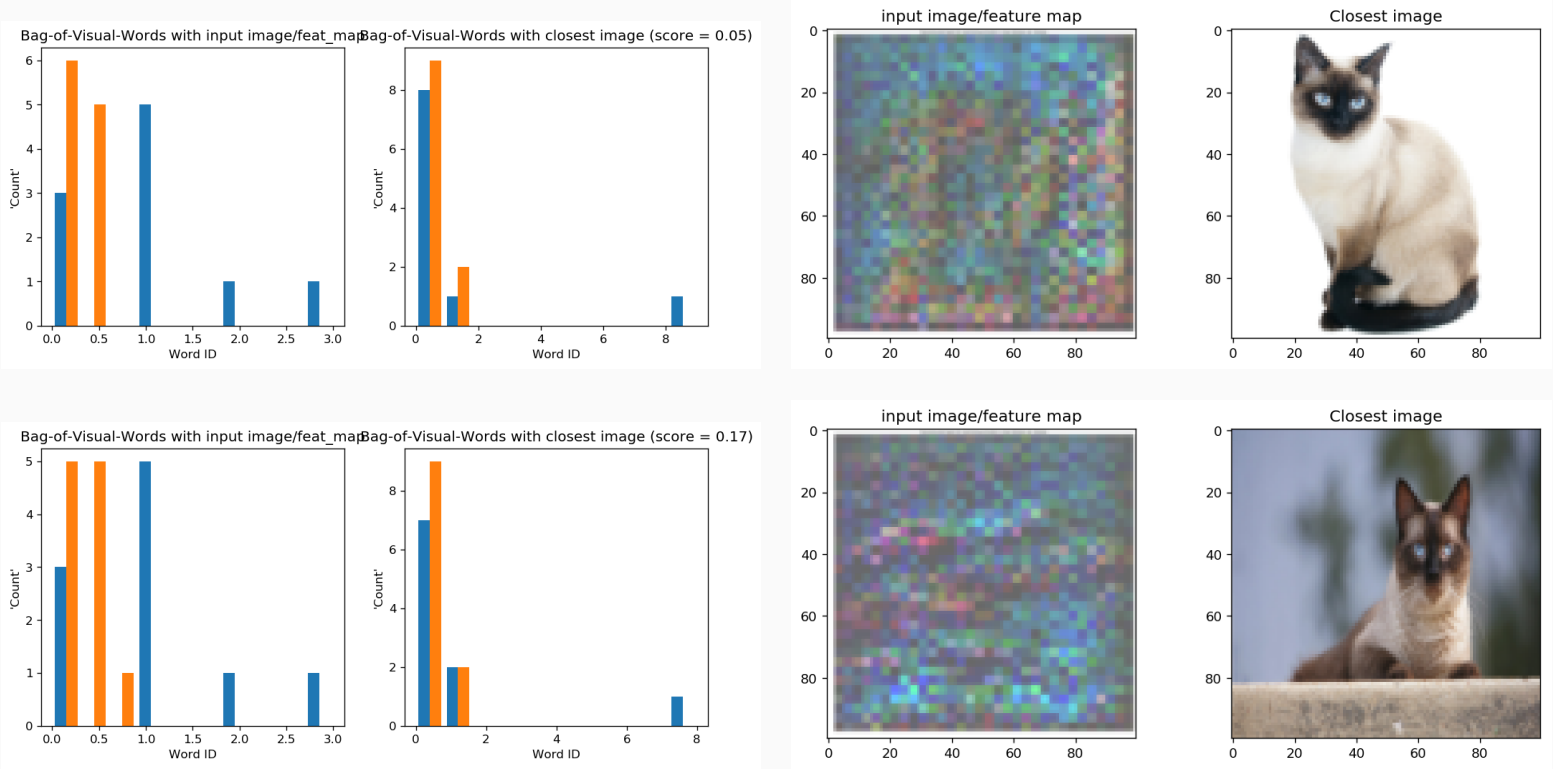


Figure 6: BoW analysis before retraining (*top*) and after retraining applied to the reconstructed inputs against the set of images of class 284 with model [Plotka, 2017].
Left: histograms associated to the reconstructed input and ”closest” image on the right-hand side.

Reconstructed input ”closer” to the original class?

The only non-random part is in the training: new weights may be different from one training session to another.

	Mean maximum score	Median maximum score
Before training	0.050	0.050
After training	0.095	0.090

Table 3: Values corresponding to the experiment described above with model [Plotka, 2017] ($n = 10$ tries).

Thus training for one class allows the NN to reconstruct slightly more resembling inputs of this class.

Conclusion

Conclusion

- We have trained 2 small CNNs using Keras.
- We have visualized the feature maps associated with an image and tried to interpret the messages that each layer encodes.
- We have reconstructed an image of a certain class from a random image by maximizing activation of the class output in the *softmax* function.
- Code is available at

  `https://github.com/kuredatan/nn-visu|`

Questions?

References i


Bonaccorso, G. (2017).
Cifar-10 image classification with keras convnet.
https://gist.github.com/giuseppebonaccorso/e77e505fc7b61983f7b42dc1250f31c8.


Chollet, F. (2016).
How convolutional neural networks see the world.
https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html.


Graham, B. (2014).
Fractional max-pooling.
arXiv preprint arXiv:1412.6071.

References ii


Harris, C. and Stephens, M. (1988).
A combined corner and edge detector.
In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer.


Krizhevsky, A. and Hinton, G. (2009).
Learning multiple layers of features from tiny images.
Technical report, Citeseer.


Lowe, D. G. (1999).
Object recognition from local scale-invariant features.
In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee.

References iii


Nguyen, A., Yosinski, J., and Clune, J. (2015).
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.


Plotka, S. (2017).
Cifar-10 classification using keras (tutorial).
`https://blog.plon.io/tutorials/cifar-10-classification-using-keras-tutorial/`.


Simonyan, K. and Zisserman, A. (2014).
Very deep convolutional networks for large-scale image recognition.
arXiv preprint arXiv:1409.1556.

References iv


Sivic, J. and Zisserman, A. (2009).
Efficient visual search of videos cast as text retrieval.
IEEE transactions on pattern analysis and machine intelligence, 31(4):591–606.


Zeiler, M. D. and Fergus, R. (2014).
Visualizing and understanding convolutional networks.
In *European conference on computer vision*, pages 818–833. Springer.


Zhang, X. and LeCun, Y. (2015).
Text understanding from scratch.
arXiv preprint arXiv:1502.01710.