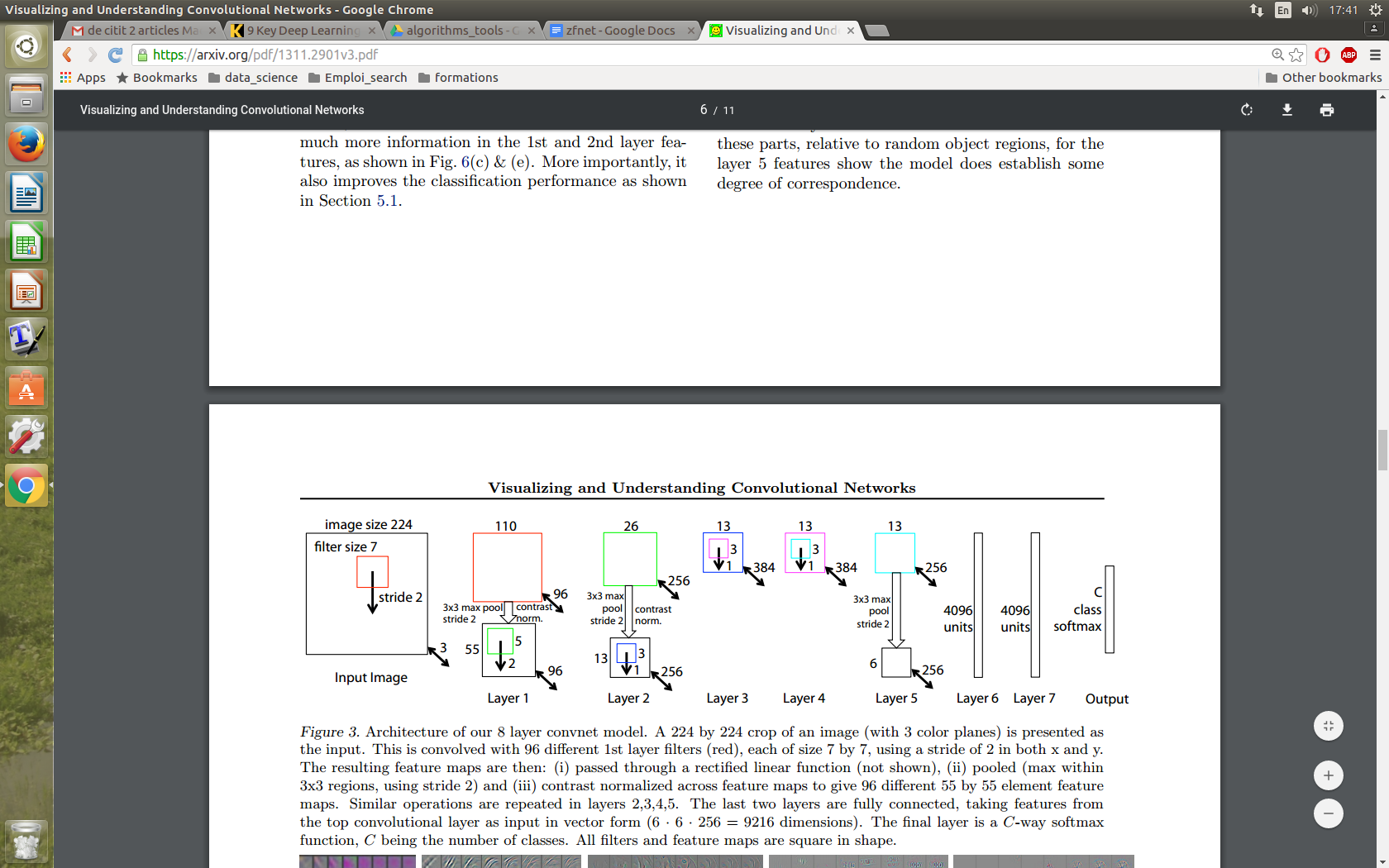
Tools for Machine Learning:

# ZFNet (2013)

With AlexNet stealing the show in 2012, there was a large increase in the number of CNN models submitted to ILSVRC 2013. The winner of the competition that year was a network built by Matthew Zeiler and Rob Fergus from NYU. Named ZF Net, this model achieved an 11.2% error rate. This architecture was more of a fine tuning to the previous AlexNet structure, but still developed some very keys ideas about improving performance. Another reason this was such a great paper is that the authors spent a good amount of time explaining a lot of the intuition behind ConvNets and showing how to visualize the filters and weights correctly.

In this paper titled “Visualizing and Understanding Convolutional Neural Networks”, Zeiler and Fergus begin by discussing the idea that this renewed interest in CNNs is due to the accessibility of large training sets and increased computational power with the usage of GPUs. They also talk about the limited knowledge that researchers had on inner mechanisms of these models, saying that without this insight, the “development of better models is reduced to trial and error”. While we do currently have a better understanding than 3 years ago, this still remains an issue for a lot of researchers! The main contributions of this paper are details of a slightly modified AlexNet model and a very interesting way of visualizing feature maps.



**Main Points**

* Very similar architecture to AlexNet, except for a few minor modifications.
* AlexNet trained on 15 million images, while ZF Net trained on only 1.3 million images.
* Instead of using 11x11 sized filters in the first layer (which is what AlexNet implemented), ZF Net used filters of size 7x7 and a decreased stride value. The reasoning behind this modification is that a smaller filter size in the first conv layer helps retain a lot of original pixel information in the input volume. A filtering of size 11x11 proved to be skipping a lot of relevant information, especially as this is the first conv layer.
* As the network grows, we also see a rise in the number of filters used.
* Used ReLUs for their activation functions, cross-entropy loss for the error function, and trained using batch stochastic gradient descent.
* Trained on a GTX 580 GPU for **twelve days**.
* Developed a visualization technique named Deconvolutional Network, which helps to examine different feature activations and their relation to the input space. Called “deconvnet” because it maps features to pixels (the opposite of what a convolutional layer does).

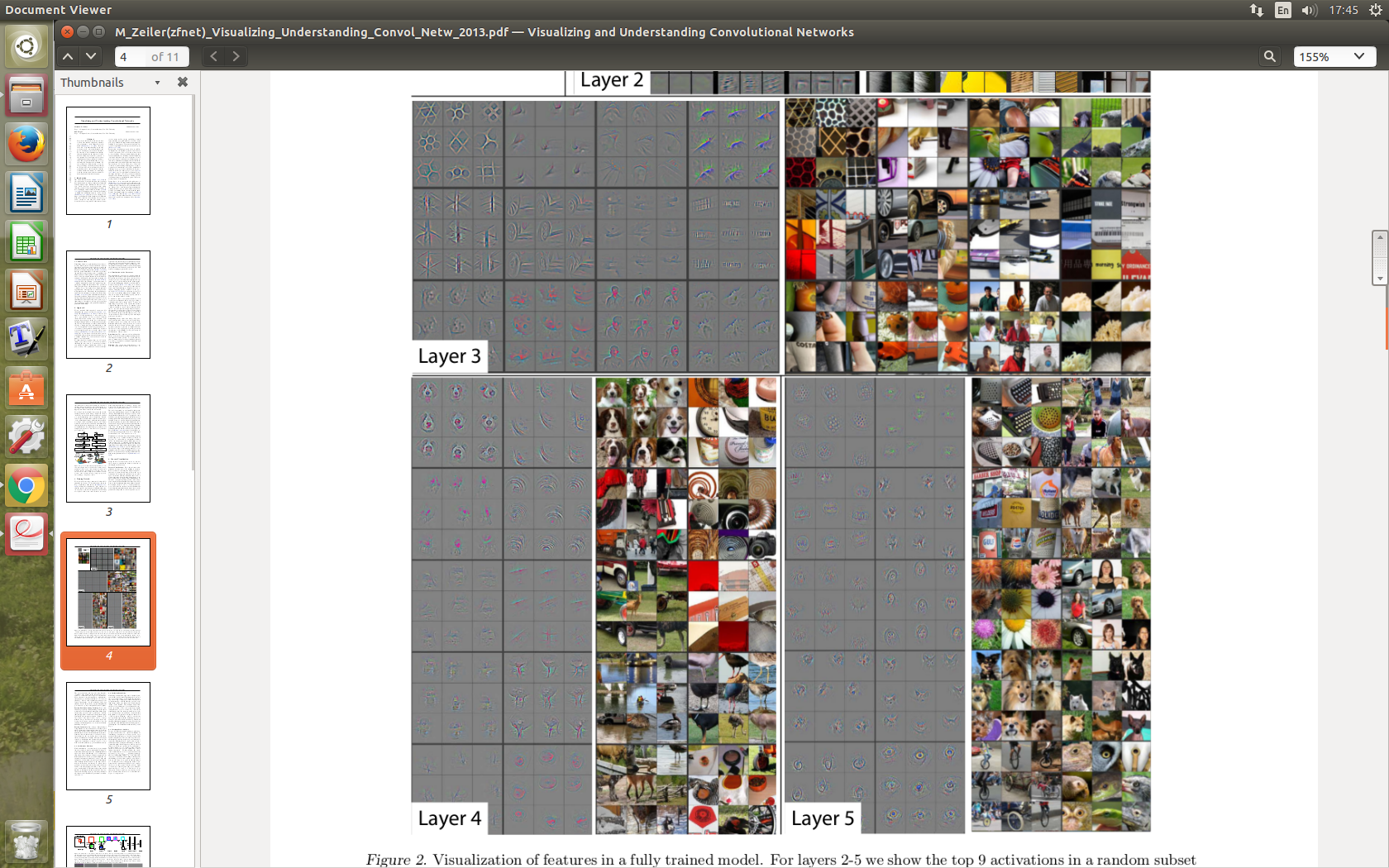
**DeConvNet**

The basic idea behind how this works is that at every layer of the trained CNN, you attach a “deconvnet” which has a path back to the image pixels. An input image is fed into the CNN and activations are computed at each level. This is the forward pass. Now, let’s say we want to examine the activations of a certain feature in the 4th conv layer. We would store the activations of this one feature map, but set all of the other activations in the layer to 0, and then pass this feature map as the input into the deconvnet. This deconvnet has the same filters as the original CNN. This input then goes through a series of unpool (reverse maxpooling), rectify, and filter operations for each preceding layer until the input space is reached.

The reasoning behind this whole process is that we want to examine what type of structures excite a given feature map. Let’s look at the visualizations of the first and second layers.



Like we discussed in [Part 1](https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/), the first layer of your ConvNet is always a low level feature detector that will detect simple edges or colors in this particular case. We can see that with the second layer, we have more circular features that are being detected. Let’s look at layers 3, 4, and 5.



These layers show a lot more of the higher level features such as dogs’ faces or flowers. One thing to note is that as you may remember, after the first conv layer, we normally have a pooling layer that downsamples the image (for example, turns a 32x32x3 volume into a 16x16x3 volume). The effect this has is that the 2nd layer has a broader scope of what it can see in the original image. For more info on deconvnet or the paper in general, check out Zeiler himself [presenting](https://www.youtube.com/watch?v=ghEmQSxT6tw) on the topic.

**Why It’s Important**

ZF Net was not only the winner of the competition in 2013, but also provided great intuition as to the workings on CNNs and illustrated more ways to improve performance. The visualization approach described helps not only to explain the inner workings of CNNs, but also provides insight for improvements to network architectures. The fascinating deconv visualization approach and occlusion experiments make this one of my personal favorite papers.