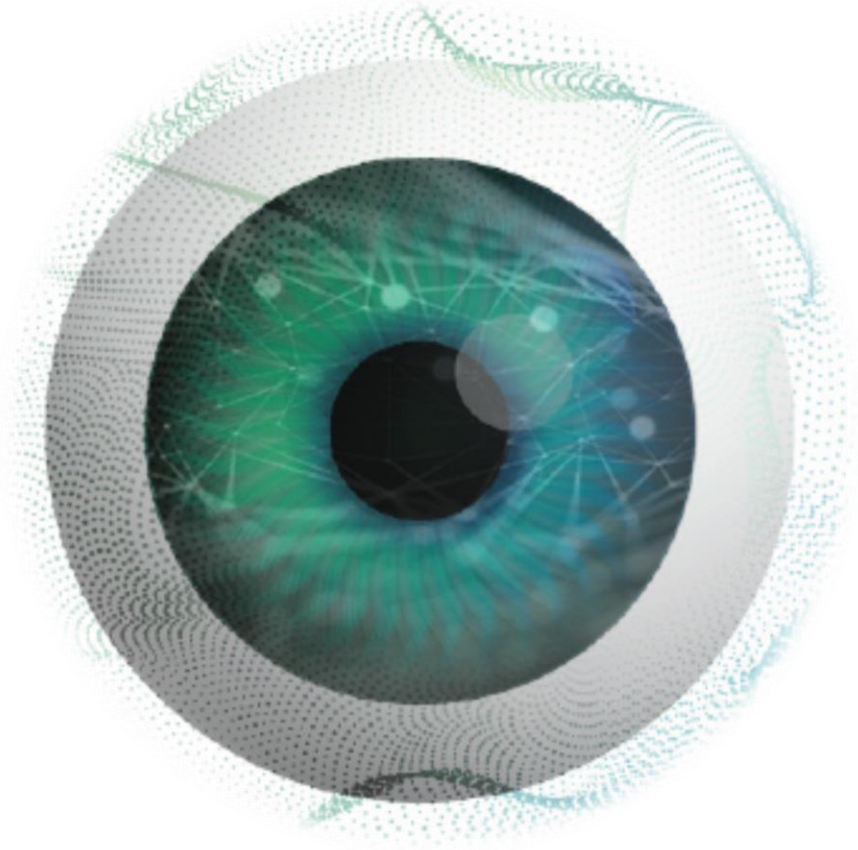
DEEP LEARNING FOR VISUAL RECOGNITION

How to write a good report



Henrik Pedersen, PhD
External lecturer
Department of Computer Science
Aarhus University
hpe@cs.au.dk

Remember

- Use this arXiv template
- Report should be structured like a research paper:
  - Abstract
  - Introduction (Includes motivation and objectives of your project. Be realistic and be specific!)
  - Related work (Summarize at least 2-3 key references. What makes your approach different?)
  - Methods (What did you do and how? Describe network architecture, data set, experiments, etc.)
  - Results (Objectively summarize your results and compare to related work or state-of-the-art)
  - Discussion (How well did you do? What worked? What didn't work?)
- Final report between 6-8 pages.

Topics

- Your task is to summarize what was learned and present it.
- Address the topics listed below:
  - **Context** (Why?): Define the environment in which the problem exists and set up the motivation for the research question.
  - **Problem** (Question): Concisely describe the problem as a question that you went out and answered.
  - **Solution** (Answer): Concisely describe the solution as an answer to the question you posed. Be specific.
  - **Findings**: List the discoveries you made along the way that interest the audience. They may be discoveries in the data, methods that did or did not work or the model performance benefits you achieved along your journey.
  - **Limitations**: Consider where the model does not work or questions that the model does not answer. Do not shy away from these questions, defining where the model excels is more trusted if you can define where it does not excel.
  - **Conclusions** (Why+Question+Answer): Revisit the why, research question and the answer you discovered in a tight little package that is easy to remember and repeat for yourself and others.

- **Introduction**
- **Related work**
- **Methods**
- **Results**
- **Discussion**

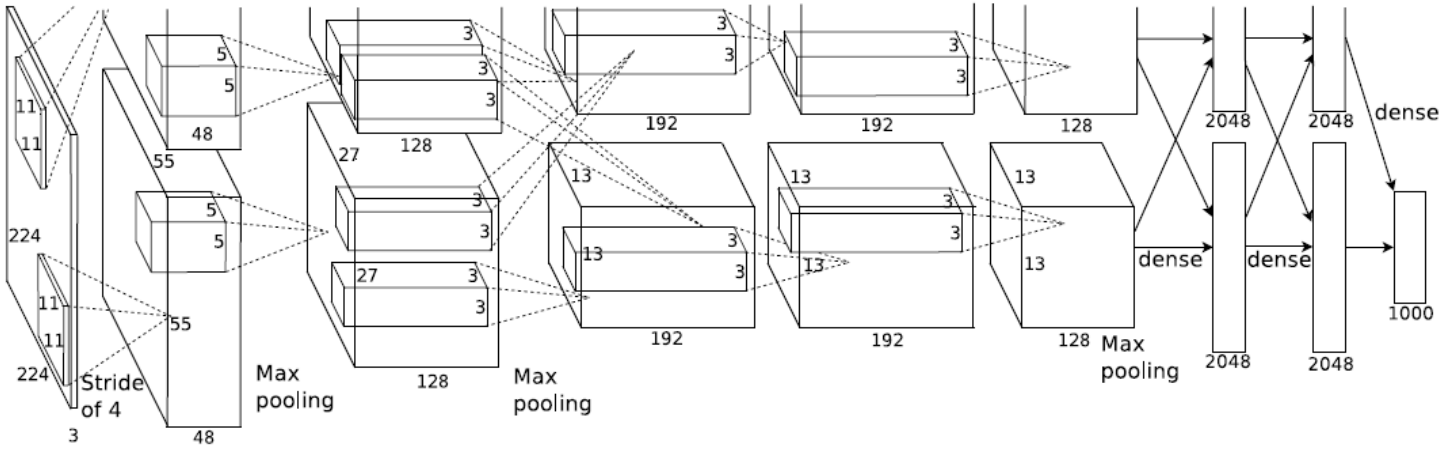
Example – AlexNet paper

ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca



Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Example – AlexNet paper

- **Abstract**
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

The abstract just summarizes the **major findings**, because this is a seminal paper. It is more common to state the research question and summarize the proposed solution and the findings.

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Context: New possibilities in **Big Data** sets, like ImageNet, **but …**

1 Introduction

Current approaches to object recognition make essential use of machine learning methods. To improve their performance, we can collect larger datasets, learn more powerful models, and use better techniques for preventing overfitting. Until recently, datasets of labeled images were relatively small — on the order of tens of thousands of images (e.g., NORB [16], Caltech-101/256 [8, 9], and CIFAR-10/100 [12]). Simple recognition tasks can be solved quite well with datasets of this size, especially if they are augmented with label-preserving transformations. For example, the current-best error rate on the MNIST digit-recognition task (<0.3%) approaches human performance [4]. But objects in realistic settings exhibit considerable variability, so to learn to recognize them it is necessary to use much larger training sets. And indeed, the shortcomings of small image datasets have been widely recognized (e.g., Pinto et al. [21]), but it has only recently become possible to collect labeled datasets with millions of images. The new larger datasets include LabelMe [23], which consists of hundreds of thousands of fully-segmented images, and ImageNet [6], which consists of over 15 million labeled high-resolution images in over 22,000 categories.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Context: New possibilities in **Big Data** sets, like ImageNet, **but …**

Problem: We need bigger models with large learning capacity

Solution: Convolutional Neural Networks (CNNs)

To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task means that this problem cannot be specified even by a dataset as large as ImageNet, so our model should also have lots of prior knowledge to compensate for all the data we don't have. Convolutional neural networks (CNNs) constitute one such class of models [16, 11, 13, 18, 15, 22, 26]. Their capacity can be controlled by varying their depth and breadth, and they also make strong and mostly correct assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies). Thus, compared to standard feedforward neural networks with similarly-sized layers, CNNs have much fewer connections and parameters and so they are easier to train, while their theoretically-best performance is likely to be only slightly worse.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Context: New possibilities in **Big Data** sets, like ImageNet, **but …**

Problem: We need bigger models with large learning capacity

Solution: Convolutional Neural Networks (CNNs)

Getting more specific: We also need GPUs…

Despite the attractive qualities of CNNs, and despite the relative efficiency of their local architecture, they have still been prohibitively expensive to apply in large scale to high-resolution images. Luckily, current GPUs, paired with a highly-optimized implementation of 2D convolution, are powerful enough to facilitate the training of interestingly-large CNNs, and recent datasets such as ImageNet contain enough labeled examples to train such models without severe overfitting.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Context: New possibilities in **Big Data** sets, like ImageNet, **but …**

Problem: We need bigger models with large learning capacity

Solution: Convolutional Neural Networks (CNNs)

Getting more specific: We also need GPUs…

Contributions of this paper: Summarize solution and findings.

> The specific contributions of this paper are as follows: we trained one of the largest convolutional neural networks to date on the subsets of ImageNet used in the ILSVRC-2010 and ILSVRC-2012 competitions [2] and achieved by far the best results ever reported on these datasets. We wrote a highly-optimized GPU implementation of 2D convolution and all the other operations inherent in training convolutional neural networks, which we make available publicly[1]. Our network contains a number of new and unusual features which improve its performance and reduce its training time, which are detailed in Section 3. The size of our network made overfitting a significant problem, even with 1.2 million labeled training examples, so we used several effective techniques for preventing overfitting, which are described in Section 4. Our final network contains five convolutional and three fully-connected layers, and this depth seems to be important: we found that removing any convolutional layer (each of which contains no more than 1% of the model's parameters) resulted in inferior performance.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Context: New possibilities in **Big Data** sets, like ImageNet, **but …**

Problem: We need bigger models with large learning capacity

Solution: Convolutional Neural Networks (CNNs)

Getting more specific: We also need GPUs…

Contributions of this paper: Summarize solution and findings.

How about the **Related Work** section?

- It is common to have a Related Work section after the Introduction.
- The purpose is to put your solution/contributions into more detailed context by explaining what other people did before you, and what makes your solution different.
- Note: The AlexNet paper has no dedicated section for related work, because it is a seminal paper. Some related work is described in the Introduction.

Example – AlexNet paper

- Abstract
- **Introduction**
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Related work: Here is a list of some important papers on CNNs:

- **Image Classification**: [Krizhevsky et al.], [Russakovsky et al.], [Szegedy et al.], [Simonyan et al.], [He et al.], [Huang et al.], [Hu et al.] [Zoph et al.], [He et al. 2], [Loffe et al.],
- **Object detection**: [Girshick et al.], [Ren et al.], [He et al.]
- **Image segmentation**: [Long et al.] [Noh et al.] [Chen et al.]
- **Video classification**: [Karpathy et al.], [Simonyan and Zisserman] [Tran et al.] [Carreira et al.] [Wang et al.]
- **Scene classification**: [Zhou et al.]
- **Face recognition**: [Taigman et al.] [Schroff et al.] [Parkhi et al.]
- **Depth estimation**: [Eigen et al.]
- **Image-to-sentence generation**: [Karpathy and Fei-Fei], [Donahue et al.], [Vinyals et al.] [Xu et al.] [Johnson et al.]
- **Visualization and optimization**: [Szegedy et al.], [Nguyen et al.], [Zeiler and Fergus], [Goodfellow et al.], [Schaul et al.]
- **Survey:** https://arxiv.org/ftp/arxiv/papers/1803/1803.01164.pdf

https://github.com/terryum/awesome-deep-learning-papers

https://github.com/floodsung/Deep-Learning-Papers-Reading-Roadmap

Example – AlexNet paper

- Abstract
- Introduction
- **The Dataset**
- **The Architecture**
  - **ReLU Nonlinearity**
  - **Training on Multiple GPUs**
  - **Local Response Normalization**
  - **Overlapping Pooling**
  - **Overall Architecture**
- **Reducing overfitting**
  - **Data augmentation**
  - **Dropout**
- **Details of learning**
- Results
  - Qualitative evaluation
- Discussion
- References

In your report, all of this goes into the **Methods** section.

Feel free to add subsections

This section describes:

- Your data set
- The network architecture(s)
- Your experiments – what did you do and how?

Note: Sometimes you will see that researchers describe their experiments in the Results section. This is not something that I would recommend.

Example – AlexNet paper

- Abstract
- Introduction
- **The Dataset**
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

Report the source of your dataset, its type and size, types of labels (if any), split into training and validation set, etc.

If it is a public dataset, it is important to report the benchmarks.

2 The Dataset

ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. The images were collected from the web and labeled by human labelers using Amazon's Mechanical Turk crowd-sourcing tool. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held. ILSVRC uses a subset of ImageNet with roughly 1000 images in each of 1000 categories. In all, there are roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images.

ILSVRC-2010 is the only version of ILSVRC for which the test set labels are available, so this is the version on which we performed most of our experiments. Since we also entered our model in the ILSVRC-2012 competition, in Section 6 we report our results on this version of the dataset as well, for which test set labels are unavailable. On ImageNet, it is customary to report two error rates: top-1 and top-5, where the top-5 error rate is the fraction of test images for which the correct label is not among the five labels considered most probable by the model.

ImageNet consists of variable-resolution images, while our system requires a constant input dimensionality. Therefore, we down-sampled the images to a fixed resolution of 256×256. Given a rectangular image, we first rescaled the image such that the shorter side was of length 256, and then cropped out the central 256×256 patch from the resulting image. We did not pre-process the images in any other way, except for subtracting the mean activity over the training set from each pixel. So we trained our network on the (centered) raw RGB values of the pixels.

Benchmarks:
- top-1
- top-5

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- **The Architecture**
  - **ReLU Nonlinearity**
  - **Training on Multiple GPUs**
  - **Local Response Normalization**
  - **Overlapping Pooling**
  - **Overall Architecture**
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- Discussion
- References

- Describe your network architecture
- Did you “just” use an existing network, or did you make any modifications?

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- **Reducing overfitting**
  - **Data augmentation**
  - **Dropout**
- **Details of learning**
- Results
  - Qualitative evaluation
- Discussion
- References

- It is important to address things like overfitting and training
- How did you train your model?
- Which optimization algorithm/loss function?
- What types of regularization?
- Did you use data augmentation? How?
- Etc.

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- **Results**
  - **Qualitative evaluation**
- Discussion
- References

- Summarize your **findings**
- Report error statistics, such as test set error in %.
  - If you are using a public dataset, it is a good idea to report the benchmark errors associated with that data set.
  - For instance ImageNet (ILSVRC-2010) used top-1 and top-5 test set error rates.
- Exemplify with qualitative results (i.e., example images).
  - Show both good and bad examples (when it worked, and when it didn't work).
- Show results that support your conclusions (Discussion section).
- Describe objectively what you see – do not discuss results here!

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- **Results**
  - **Qualitative evaluation**
- Discussion
- References

- Examples of **reporting error statistics**

Model	Top-1	Top-5
Sparse coding [2]	*47.1%*	*28.2%*
SIFT + FVs [24]	*45.7%*	*25.7%*
CNN	**37.5%**	**17.0%**

Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- **Results**
  - **Qualitative evaluation**
- Discussion
- References

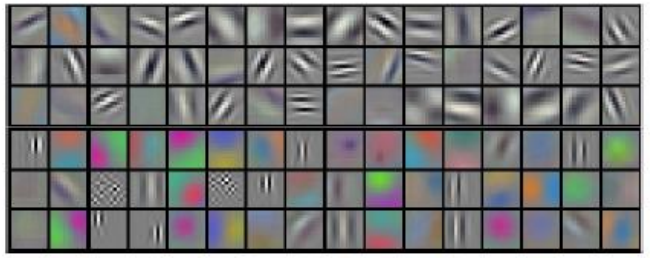


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

- Examples of **qualitative results**

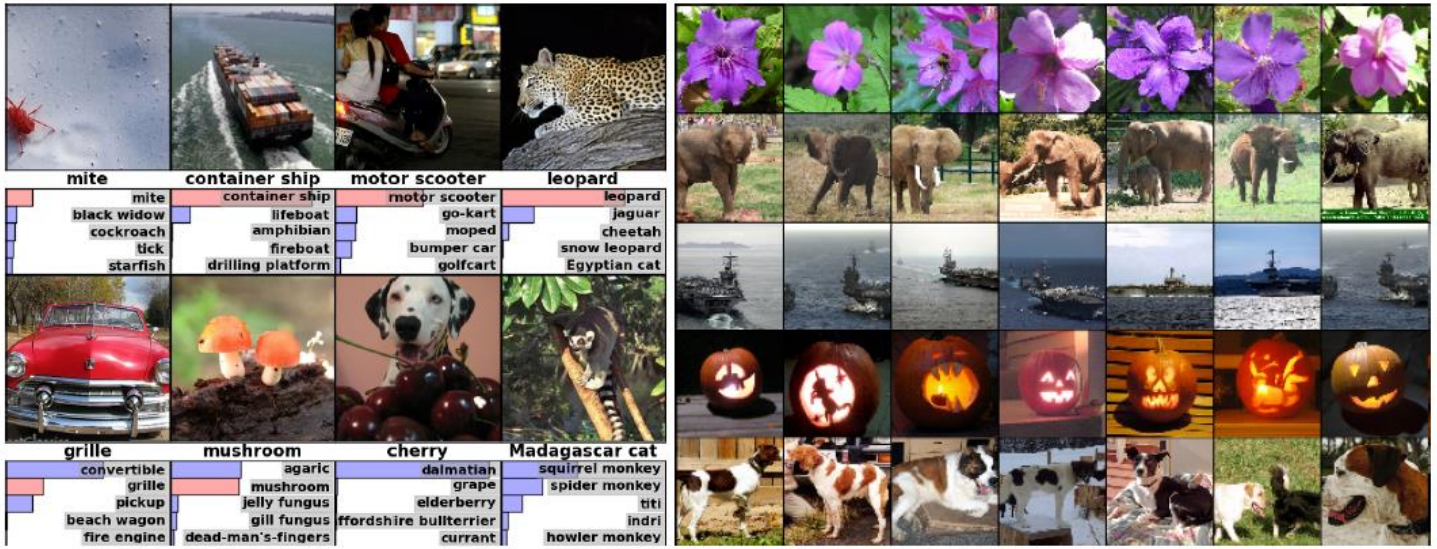


Figure 4: (**Left**) Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). (**Right**) Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

Example – AlexNet paper

- Abstract
- Introduction
- The Dataset
- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling
  - Overall Architecture
- Reducing overfitting
  - Data augmentation
  - Dropout
- Details of learning
- Results
  - Qualitative evaluation
- **Discussion**
- References

- How well did you do?
- What worked?
- What didn't work?
- What are the limitations of your model?

7 Discussion

Our results show that a large, deep convolutional neural network is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. It is notable that our network's performance degrades if a single convolutional layer is removed. For example, removing any of the middle layers results in a loss of about 2% for the top-1 performance of the network. So the depth really is important for achieving our results.

To simplify our experiments, we did not use any unsupervised pre-training even though we expect that it will help, especially if we obtain enough computational power to significantly increase the size of the network without obtaining a corresponding increase in the amount of labeled data. Thus far, our results have improved as we have made our network larger and trained it longer but we still have many orders of magnitude to go in order to match the infero-temporal pathway of the human visual system. Ultimately we would like to use very large and deep convolutional nets on video sequences where the temporal structure provides very helpful information that is missing or far less obvious in static images.

Public computer vision datasets

- Meta Pointer: A large collection organized by CV Datasets.
- Yet another Meta pointer
- ImageNet: a large-scale image dataset for visual recognition organized by WordNet hierarchy
- SUN Database: a benchmark for scene recognition and object detection with annotated scene categories and segmented objects
- Places Database: a scene-centric database with 205 scene categories and 2.5 millions of labelled images
- NYU Depth Dataset v2: a RGB-D dataset of segmented indoor scenes
- Microsoft COCO: a new benchmark for image recognition, segmentation and captioning.
- Flickr100M: 100 million creative commons Flickr images
- Labeled Faces in the Wild: a dataset of 13,000 labeled face photographs
- Human Pose Dataset: a benchmark for articulated human pose estimation
- YouTube Faces DB: a face video dataset for unconstrained face recognition in videos
- UCF101: an action recognition data set of realistic action videos with 101 action categories
- Moments in Time: A dataset of one million 3-second videos

Public computer vision datasets

- https://lionbridge.ai/datasets/20-best-image-datasets-for-computer-vision/
- Labelme: A large dataset created by the MIT Computer Science and Artificial Intelligence Laboratory (CSAIL) containing 187,240 images, 62,197 annotated images, and 658,992 labeled objects.
- Youtube-8M: a large-scale labeled dataset that consists of millions of YouTube video IDs, with annotations of over 3,800+ visual entities.
- Labelled Faces in the Wild: 13,000 labeled images of human faces, for use in developing applications that involve facial recognition.
- Stanford Dogs Dataset: Contains 20,580 images and 120 different dog breed categories, with about 150 images per class.
- Places: Scene-centric database with 205 scene categories and 2.5 million images with a category label.
- CelebFaces: Face dataset with more than 200,000 celebrity images, each with 40 attribute annotations.
- CIFAR-10: A large image dataset of 60,000 32×32 colour images split into 10 classes. The dataset is divided into five training batches and one test batch, each containing 10,000 images.
- Indoor Scene Recognition: A very specific dataset, useful as most scene recognition models are better ‘outside’. Contains 67 Indoor categories, and a total of 15620 images.
- HMDB-51: a large human motion dataset of 51 action classes
- ActivityNet: A large-scale video dataset for human activity understanding

Example – ImageNet

- Let's look at ImageNet: http://www.image-net.org/

- **ImageNet** is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images. Currently over five hundred images per node.
- Paper: https://arxiv.org/abs/1409.0575

LSVRC 2010

- ImageNet Large Scale Visual Recognition Challenge (LSVRC) 2010
- The objective of this competition is **object classification**
- Dataset:
  - Hand-labelled with the presence or absence of 1000 object categories
  - **Training data:** 1.2 million images, **Validation set:** 50,000 images, **Test set:** 150,000 images (without labels).
  - Competition: Submit list of predictions for test set.
- Benchmark/evaluation criteria
  - For each image, **produce a list of 5 object categories** in the descending order of confidence.
  - The idea is to allow an algorithm to identify multiple objects in an image and not be penalized if one of the objects identified was in fact present, but not included in the ground truth.
  - For each image, an algorithm will produce 5 labels l_j, j = 1,...,5.
  - The ground truth labels for the image are g_k, k = 1,...,n with n objects labelled (n=1).
  - The error of the algorithm for that image would be $e = 1/n \Sigma_k \min_j d(l_j,g_k)$.
  - For criteria a) d(x,y) = 0 if x=y and 1 otherwise.
  - For criteria b) d(x,y) = height of the lowest common ancestor of x and y in the category hierarchy.

LSVRC 2011

- ImageNet Large Scale Visual Recognition Challenge (LSVRC) 2011
- Same as LSVRC 2010 + new task: **specifying the location of objects**.
- Dataset:
  - Same images as LSVRC 2010
- Benchmark/evaluation criteria
  - New task: Classification with localization
  - For each image, produce 5 class labels l_j, j = 1,...,5 and 5 bounding boxes b_j, j = 1,...5, one for each class label.
  - For each ground truth class label g_k, the ground truth bounding boxes are z_{km}, m = 1,...M_k, where M_k is the number of instances of the k'th object in the current image.
  - The error of the algorithm for that image would be $e = 1/n \cdot \sum_k \min_j \min_{Mk}{}^m \max\{d(l_j,g_k),f(b_j, z_{km})\}$
  - where $f(b_j, z_{km}) = 0$ if b_j and z_{km} has over 50% overlap (IoU>0.5), and $f(b_j, z_{km}) = 1$ otherwise.
  - What is IoU: Intersection over Union?

Intersection over Union (IoU)

- IoU measures the overlap between 2 boundaries.
- We use that to measure how much our predicted boundary overlaps with the ground truth (the real object boundary).
- In LSVRC 2011, we predefine an IoU threshold (say 0.5) in classifying whether the prediction is a true positive or a false positive.

Ground truth

Prediction

$$IoU = \frac{\text{area of overlap}}{\text{area of union}}$$

LSVRC 2012

- ImageNet Large Scale Visual Recognition Challenge (LSVRC) 2012
- Same as LSVRC 2012 + new task: **Fine-grained classification**
- Dataset:
  - Same images as LSVRC 2010
- Benchmark/evaluation criteria
  - New task: Fine-grained classification on 100+ dog categories
  - For each of the dog categories predict if a specified dog (indicated by their bounding box) in a test image is of a particular category.
  - The output from your system should be a real-valued confidence that the dog is of a particular category so that a **precision/recall curve** can be drawn.
  - The fine-grained classification task will be judged by the precision/recall curve.
  - The principal quantitative measure used will be the **average precision (AP)** on individual categories and the mean **average precision (mAP)** across all categories.

Precision and Recall

- **Precision** measures how good a model is at predicting the positive class. It is also known as positive prediction rate.
- **Recall** measures how good the model is at predicting the positive class when the actual outcome is positive. It is also known as true positive rate or sensitivity.

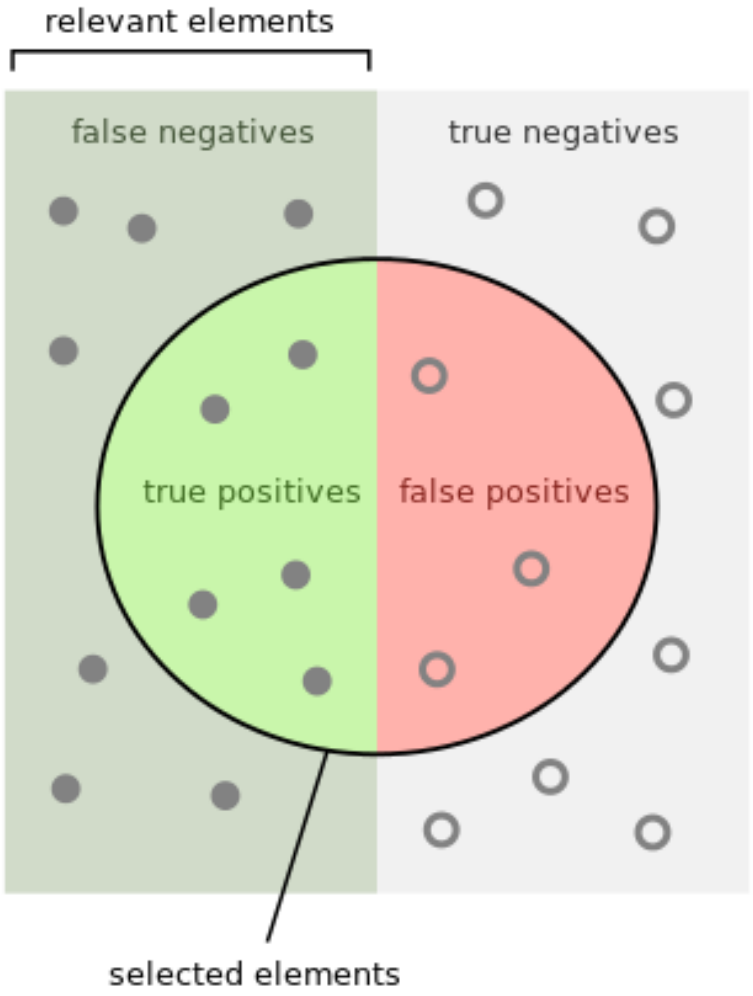
$$\text{Precision} = \frac{\#TP}{\#TP + \#FP}$$

$$\text{Recall} = \frac{\#TP}{\#TP + \#FN}$$

TP: True positive
TN: True negative (not used here)
FP: False positive
FN: False negative

- Related:
  - True negative rate or specificity: #TN / (#TN+#FP)
  - Accuracy: (#TP + #TN) / (#TP + #TN + #FP + #FN)

Precision and Recall



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

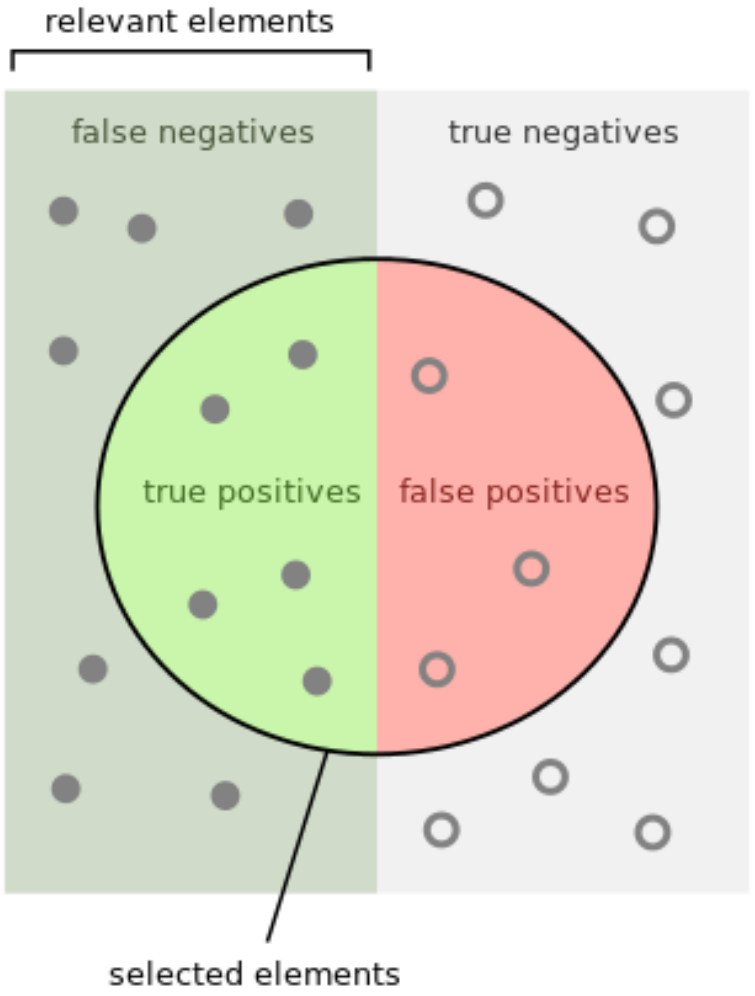
How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

In classification, a **precision of 1.0** for a class C means that every item labelled as belonging to class C does indeed belong to class C (but says nothing about the number of items from class C that were not labelled correctly) whereas a **recall of 1.0** means that every item from class C was labelled as belonging to class C (but says nothing about how many items from other classes were incorrectly also labelled as belonging to class C).

https://en.wikipedia.org/wiki/Precision_and_recall

Precision and Recall



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{selected elements}}$$

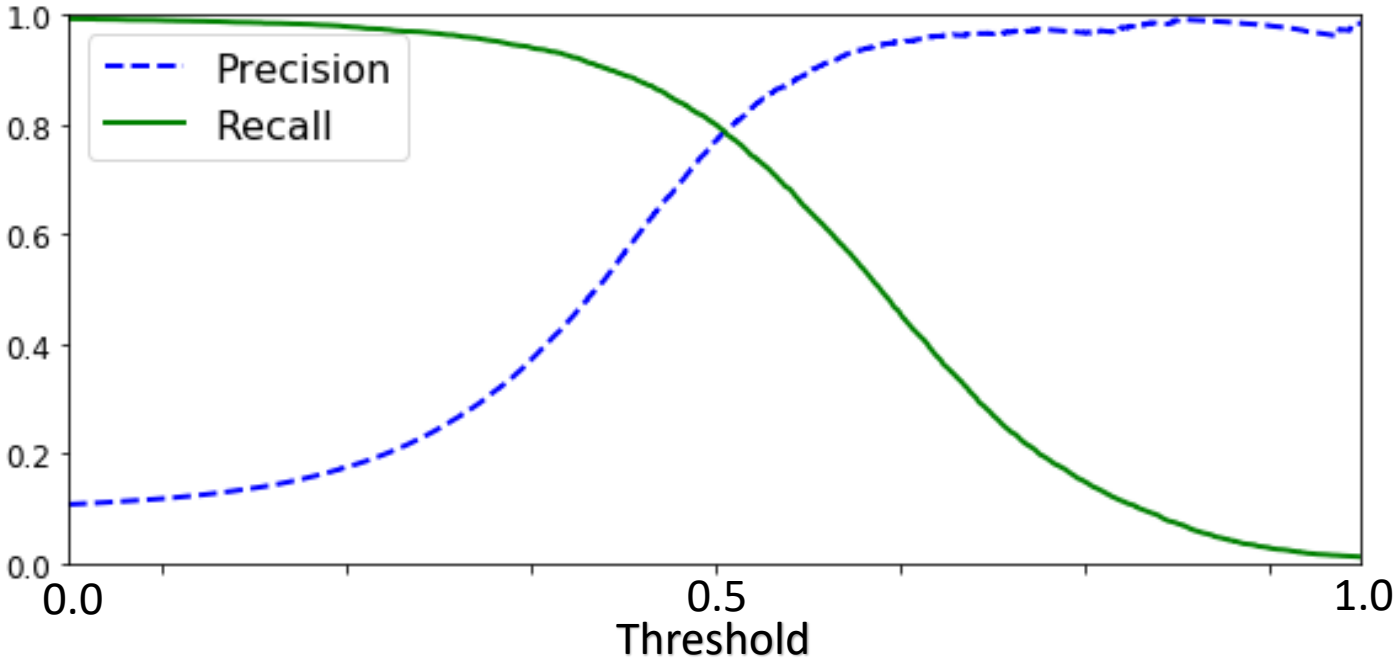
How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{relevant elements}}$$

Example: Object detection
Goal: Locate objects with bounding boxes and classify the objects as well
Interpretation: Precision tells us how many of the bounding boxes we detected are actual objects, where recall tells us how many of the objects we actually found.

https://en.wikipedia.org/wiki/Precision_and_recall

Precision-Recall tradeoff



Our classification models predict the probabilities for each class. Precision-Recall curves summarize the trade-off between precision and recall **using different probability thresholds**.

Binary classification: In an ideal scenario where there is a perfectly separable data, both precision and recall can get maximum value of 1.0. But in most of the practical situations, there is noise in the dataset and the dataset is not perfectly separable. There might be some points of positive class closer to the negative class and vice versa. In such cases, shifting the decision boundary can either increase the precision or recall but not both. Increasing one parameter leads to decreasing of the other. In other words, binary classifier will miss classify some points always. Miss classification means classifying data point from negative class as positive and from positive class as negative. This miss rate is either compromising precision or recall score.

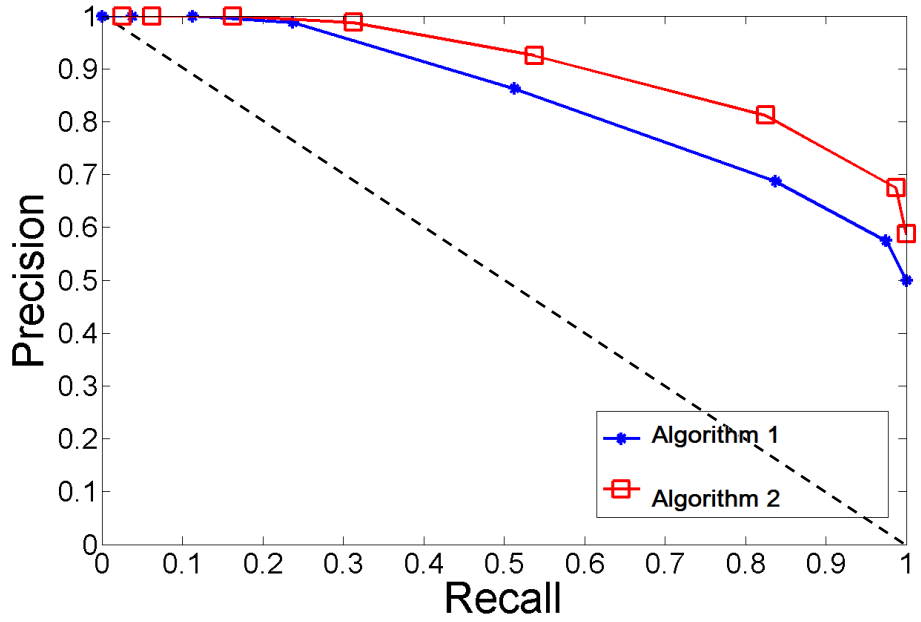
https://www.machinelearningaptitude.com/topics/machine-learning/what-is-precision-recall-tradeoff

Precision-Recall curves

- Precision-Recall curves summarize the trade-off between precision and recall **using different probability thresholds**.
- Our classification models predict the probabilities for each class.
- The reason for this is to provide the capability to choose and even calibrate the threshold for how to interpret the predicted probabilities.
- For example, a default might be to use a threshold of 0.5, meaning that a probability in [0.0, 0.49] is a negative outcome (0) and a probability in [0.5, 1.0] is a positive outcome (1).
- This threshold can be adjusted to tune the behaviour of the model for a specific problem.
- An example would be to reduce one or another type of error: FP or FN
- A precision-recall curve is a plot of the precision (y-axis) and the recall (x-axis) for different thresholds, much like the ROC curve.

Precision-Recall curves

- A precision-recall curve is a plot of the precision (y-axis) and the recall (x-axis) for different thresholds.
- For a dataset with an equal number of positive and negative cases, the “no-skill line” is straight line between [0.0,1.0] and [1.0,0.0].
- Points above this line show skill.
- A model with perfect skill is depicted as a point at [1.0,1.0].
- A skilful model is represented by a curve that bows towards [1.0,1.0] above the flat line of no skill.
- Model selection: Which threshold corresponds to the best model? Pick the threshold closest to [1.0,1.0].



Scores

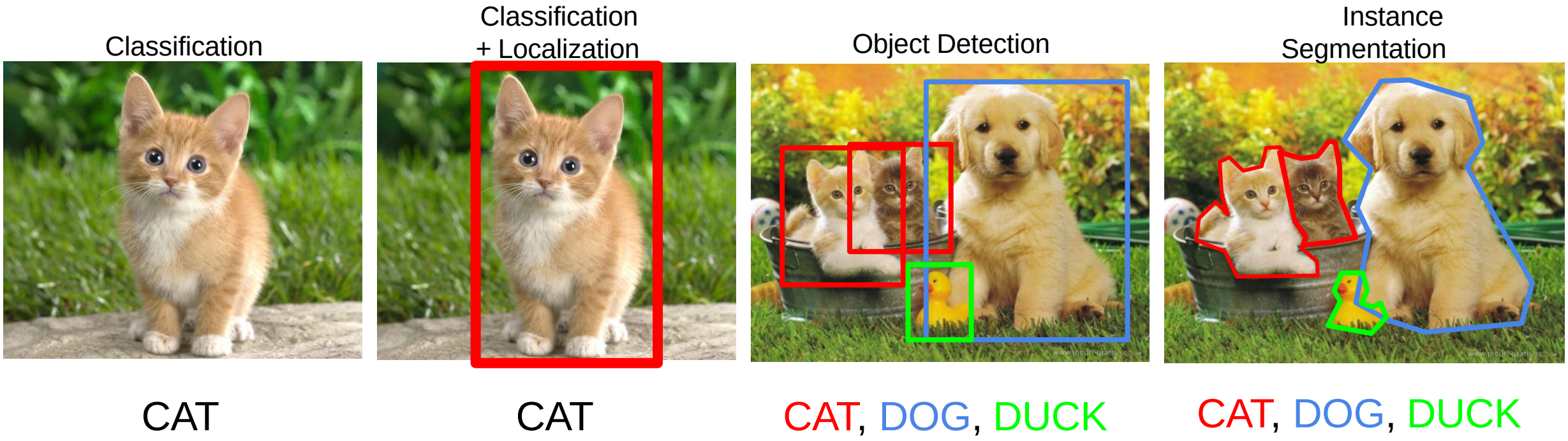
- There are also composite scores that attempt to summarize the precision and recall; three examples include:
- **Average Precision (AP)**: AP summarizes a precision-recall curve as the weighted mean of precisions achieved at each threshold, with the increase in recall from the previous threshold used as the weight: $AP = \sum_n (R_n - R_{n-1}) P_n$, where P_n and R_n are the precision and recall at the n'th threshold.
- **F score** or F1 score: that calculates the harmonic mean of the precision and recall (harmonic mean because the precision and recall are ratios).
- **Area Under Curve (AUC)**: summarizes the integral or an approximation of the area under the precision-recall curve.
- In terms of model selection, F1 summarizes model skill for a specific probability threshold, whereas AP and AUC summarize the skill of a model across thresholds.

LSVRC 2013

- ImageNet Large Scale Visual Recognition Challenge (LSVRC) 2013
- Same as LSVRC 2011 + new task: **Object detection**
- Dataset:
  - Same images as LSVRC 2010 + New data set with 200 categories with annotated bounding boxes and multiple objects per image.
  - **Training data:** 400,000 images, **Validation set:** 20,000 images, **Test set:** 150,000 images (without labels).
- Benchmark/evaluation criteria
  - New task: Object detection
  - For each image, produce a set of annotations (c_i, b_i, s_i) of class labels c_i, bounding boxes b_i and confidence scores s_i.
  - This set is expected to contain each instance of each of the 200 object categories. Objects which were not annotated will be penalized, as will be duplicate detections (two annotations for the same object instance). The winner of the detection challenge will be the team which achieves first place accuracy on the most object categories.

Note: Object detection vs localization

- With **object localization** the network identifies where the object is, putting a bounding box around it. This is what is called “classification with localization” in LSVRC.
- **Object detection** takes care of detecting and localizing multiple objects within the image.



COCO dataset

- Common Objects in Context (COCO)
- COCO is a large-scale object detection, segmentation, and captioning dataset.
- COCO has several features:
  - Object segmentation
  - Recognition in context
  - Superpixel stuff segmentation
  - 330K images (>200K labeled)
  - 1.5 million object instances
  - 80 object categories
  - 91 stuff categories
  - 5 captions per image
  - 250,000 people with keypoints
- Paper: https://arxiv.org/abs/1405.0312

COCO detection evaluation

- Latest research papers tend to give results for the COCO dataset only.
- For COCO, AP is the average over multiple IoU (the minimum IoU to consider a positive match).
- **AP@[.5:.95]** corresponds to the average AP for IoU from 0.5 to 0.95 with a step size of 0.05.
- **mAP** (mean average precision) is the average of AP. In some contexts, we compute the AP for each class and average them. But in other contexts, they mean the same thing. For example, under the COCO context, there is no difference between AP and mAP.

```
Average Precision (AP):
   AP                  % AP at IoU=.50:.05:.95 (primary challenge metric)
   AP^IoU=.50          % AP at IoU=.50 (PASCAL VOC metric)
   AP^IoU=.75          % AP at IoU=.75 (strict metric)
AP Across Scales:
   AP^small            % AP for small objects: area < 32^2
   AP^medium           % AP for medium objects: 32^2 < area < 96^2
   AP^large            % AP for large objects: area > 96^2
Average Recall (AR):
   AR^max=1            % AR given 1 detection per image
   AR^max=10           % AR given 10 detections per image
   AR^max=100          % AR given 100 detections per image
AR Across Scales:
   AR^small            % AR for small objects: area < 32^2
   AR^medium           % AR for medium objects: 32^2 < area < 96^2
   AR^large            % AR for large objects: area > 96^2
```

COCO detection evaluation

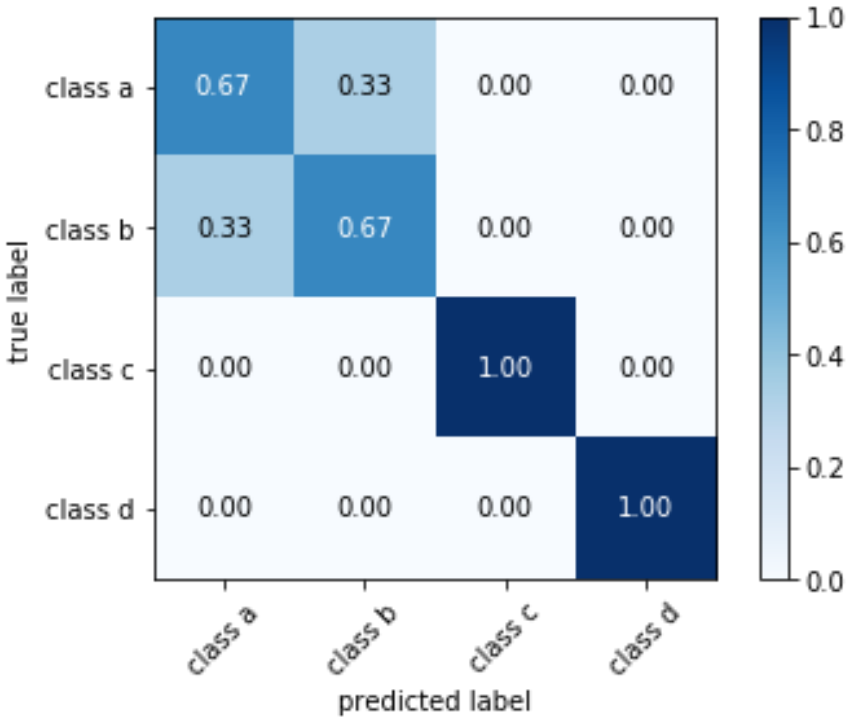
- And, this is the AP result for the YOLOv3 detector.

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [13]	DarkNet-19 [13]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9, 2]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [2]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [7]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [7]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

COCO for YOLOv3

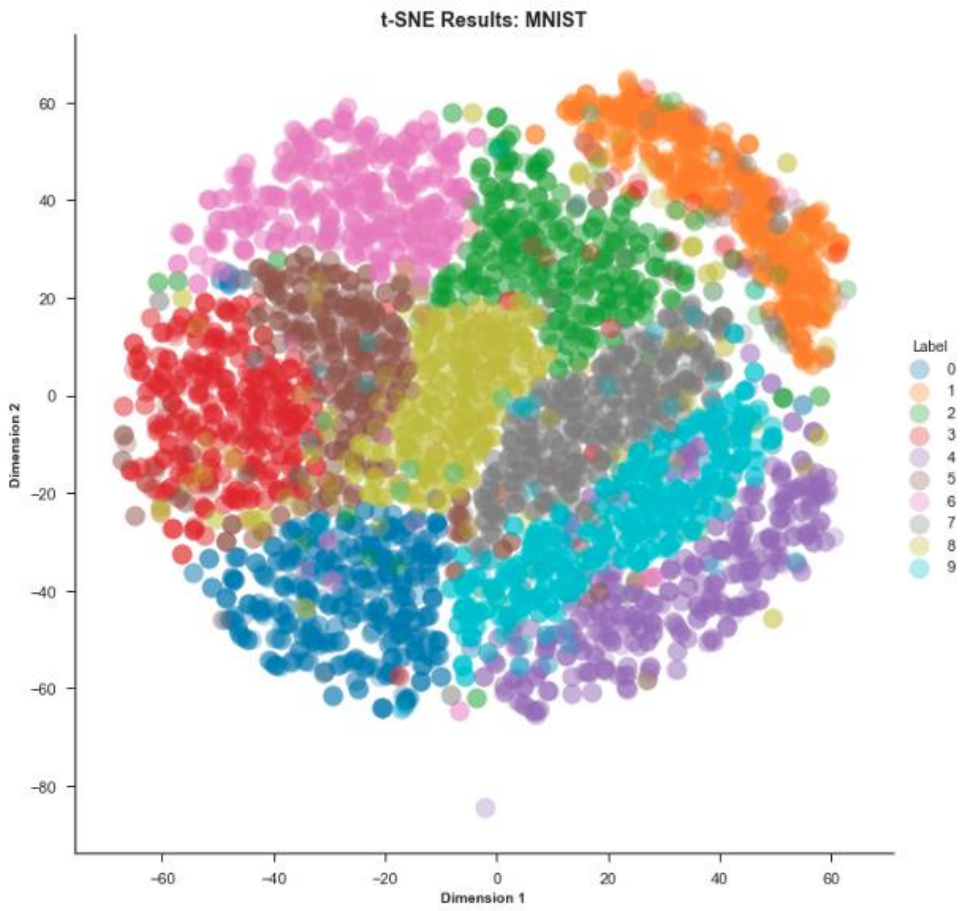
Other tools: Confusion matrices

- Confusion matrices make it easy to see if your model is confusing two classes (i.e. commonly mis-labeling one as another).
- More info: https://www.geeksforgeeks.org/confusion-matrix-machine-learning/



Other tools: t-SNE

- t-Distributed Stochastic Neighbor Embedding (t-SNE) is an unsupervised, non-linear technique primarily used for data exploration and visualizing high-dimensional data.
- t-SNE gives you a feel or intuition of how the data is arranged in a high-dimensional space.
- It was developed by Laurens van der Maatens and Geoffrey Hinton in 2008.
- What to look for? Do the data points cluster or are they just scattered randomly?
- More info: https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1



Other tools: CNN visualization

- There are few ways of visualizing your CNN outputs (intermediate as well as final layers), which can help you gain more insight into the inner working of your model.
- You can visualize
  - intermediate layer activations
  - learned filters (convolution kernels)
  - heatmaps of class activations
  - images that maximally activate a neuron
- For more info see slides for **Lecture 10** or:
  - https://towardsdatascience.com/visual-interpretability-for-convolutional-neural-networks-2453856210ce
  - https://towardsdatascience.com/understanding-your-convolution-network-with-visualizations-a4883441533b
  - https://glassboxmedicine.com/2019/06/11/cnn-heat-maps-class-activation-mapping-cam/
  - http://cs231n.github.io/understanding-cnn/

How about GANs?

- Unlike other deep learning neural network models that are trained with a loss function until convergence, a GAN generator model is trained using a second model called a discriminator that learns to classify images as real or generated.
- Both the generator and discriminator model are trained together to maintain an equilibrium.
- As such, there is no objective loss function used to train the GAN generator models and no way to objectively assess the progress of the training and the relative or absolute quality of the model from loss alone.
- Instead, a suite of qualitative and quantitative techniques have been developed to assess the performance of a GAN model based on the quality and diversity of the generated synthetic images.
- More info:
  - https://medium.com/@jonathan_hui/gan-how-to-measure-gan-performance-64b988c47732
  - https://machinelearningmastery.com/how-to-evaluate-generative-adversarial-networks/

Document your result with notebooks

- You should hand in a notebook that documents your work.
- I should be able to reproduce your results.
- If possible, please provide code to download your dataset.
- You can hand in the notebook as a separate file, or by providing a link to the online version.

Characteristics of good scientific writing

- Good scientific writing is:
  - **clear** - it avoids unnecessary detail;
  - **simple** - it uses direct language, avoiding vague or complicated sentences. Technical terms and jargon are used only when they are necessary for accuracy;
  - **impartial** - it avoids making assumptions (Everyone knows that ...) and unproven statements (It can never be proved that ...). It presents how and where data were collected and supports its conclusions with evidence;
  - **structured logically** - ideas and processes are expressed in a logical order. The text is divided into sections with clear headings;
  - **accurate** - it avoids vague and ambiguous language such as about, approximately, almost;
  - **objective** - statements and ideas are supported by appropriate evidence that demonstrates how conclusions have been drawn as well as acknowledging the work of others.
- Use active tense and check your grammar!
- Read more here: https://www2.le.ac.uk/offices/ld/resources/writing/writing-resources/science

Summary

- Good scientific writing takes practise.
- There is a logical structure: Introduction -> Related Work -> … -> Discussion. **Use it!**
- Use standard error metrics (accuracy, AP, mAP, etc.). **Don’t invent your own!**
- Lot’s of tools to help you visualize and interpret your model.
- Check your grammar.