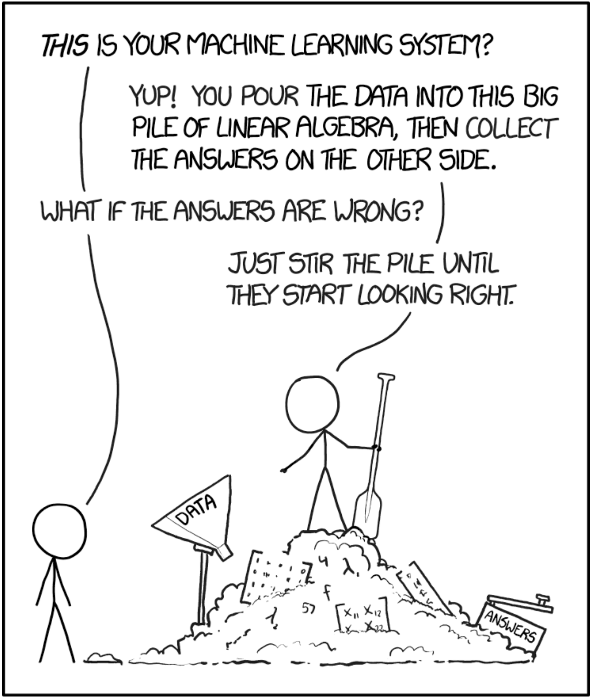
Deep learning in practice

E. Decencière

MINES ParisTech
PSL Research University
Center for Mathematical Morphology



Credits: xkcd.com
(CC BY-NC 2.5)

Practicing deep learning

In every discipline theory and practice are important. In deep learning, practice is essential.

- We lack theoretical understanding of the success of deep learning

Practicing deep learning

In every discipline theory and practice are important. In deep learning, practice is essential.

- We lack theoretical understanding of the success of deep learning
- Many common techniques have been adopted for empirical reasons

Practicing deep learning

In every discipline theory and practice are important. In deep learning, practice is essential.

- We lack theoretical understanding of the success of deep learning
- Many common techniques have been adopted for empirical reasons
- In order to improve your skills, you have to practice and read the reports by other practitioners

Practicing deep learning

In every discipline theory and practice are important. In deep learning, practice is essential.

- We lack theoretical understanding of the success of deep learning
- Many common techniques have been adopted for empirical reasons
- In order to improve your skills, you have to practice and read the reports by other practitioners

Practicing deep learning

In every discipline theory and practice are important. In deep learning, practice is essential.

- We lack theoretical understanding of the success of deep learning
- Many common techniques have been adopted for empirical reasons
- In order to improve your skills, you have to practice and read the reports by other practitioners

This state of affairs can of course be a problem in domains where security and interpretability are important, such as clinical applications.

General workflow

- Familiarize yourself with the problem and the data

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
    - Analyze the results on the validation data (look at the images!)

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
    - Analyze the results on the validation data (look at the images!)
    - Beware of over-fitting! Envision regularization methods.

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
  - Analyze the results on the validation data (look at the images!)
  - Beware of over-fitting! Envision regularization methods.
  - Use data augmentation. If correctly used, it cannot hurt and will probably improve the generalization power of your network.

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
  - Analyze the results on the validation data (look at the images!)
  - Beware of over-fitting! Envision regularization methods.
  - Use data augmentation. If correctly used, it cannot hurt and will probably improve the generalization power of your network.
  - Do you need preprocessing? Post-processing?

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
  - Analyze the results on the validation data (look at the images!)
  - Beware of over-fitting! Envision regularization methods.
  - Use data augmentation. If correctly used, it cannot hurt and will probably improve the generalization power of your network.
  - Do you need preprocessing? Post-processing?
  - Iterate, while precisely logging all your experiments.

General workflow

- Familiarize yourself with the problem and the data
- Cast the task at hand into a machine learning problem
- Build your data set
- Build an architecture and train it or choose a pre-trained model and use transfer learning
    - Analyze the results on the validation data (look at the images!)
    - Beware of over-fitting! Envision regularization methods.
    - Use data augmentation. If correctly used, it cannot hurt and will probably improve the generalization power of your network.
    - Do you need preprocessing? Post-processing?
    - Iterate, while precisely logging all your experiments.
- Only at the end: test!

Contents

Representing your problem

Cast your problem into a convenient representation:

- Understand the problem definition - discuss with the end-user

Representing your problem

Cast your problem into a convenient representation:

- Understand the problem definition - discuss with the end-user
- Familiarize yourself with the data (input and output)

Representing your problem

Cast your problem into a convenient representation:

- Understand the problem definition - discuss with the end-user
- Familiarize yourself with the data (input and output)
- Choose the right representation for your images.

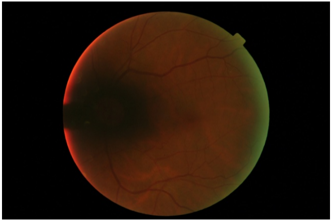
Representing your problem

Cast your problem into a convenient representation:

- Understand the problem definition - discuss with the end-user
- Familiarize yourself with the data (input and output)
- Choose the right representation for your images.
- Define an evaluation procedure

Example: eye fundus image quality

Good quality



Low quality

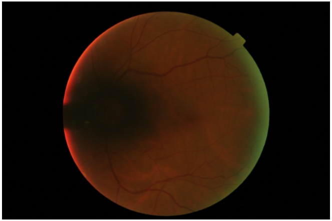
Problem definition

Quality criterion defined by the end-user: are the macula and peripheral vessels visible?

Credits: images from OPHDIAT database

Example: eye fundus image quality

Good quality



Low quality

Problem definition

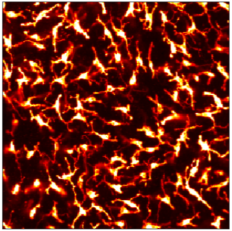
Quality criterion defined by the end-user: are the macula and peripheral vessels visible?

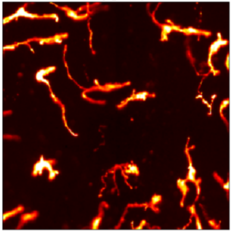
- First solution: classification (is the macula visible?)
- Second solution: macula segmentation

Credits: images from OPHDIAT database

Counting cells

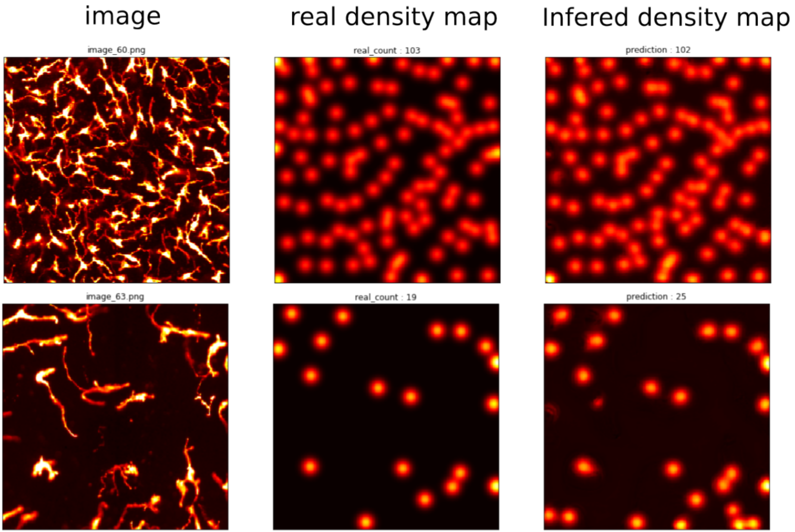
image





Credits: Tristan Lazard, master thesis. In collaboration with L'Oréal.

Counting cells



Credits: Tristan Lazard, master thesis. In collaboration with L'Oréal.

Performance evaluation

- Choose the right metrics and try to use a loss function that is as close as possible to these metrics
- Define an objective

Contents

Building the data sets

- Gather your images in order to build a data set that conveniently represents your problem
- How many images do you need?
- Build a proper ground-truth

Building the data sets

- Gather your images in order to build a data set that conveniently represents your problem
- How many images do you need?
- Build a proper ground-truth

Is database constitution the main step?

- In practical, real-world applications, this is becoming the most time-consuming step
- If the data set does not conveniently represent your problem you will run into difficulties


François Chollet a retweeté
Andrew Ng @AndrewYNg · 11h
I'm with @fchollet on this. There're some best-practices on creating and organizing data that experienced applied ML people use, but we still need to flesh out and widely disseminate these ideas. This will be key to getting more ML systems deployed.
François Chollet @fchollet · 24 janv.
ML researchers work with fixed benchmark datasets, and spend all of their time searching over the knobs they do control: architecture & optimization. In applied ML, you're likely to spend most of your time on data collection and annotation -- where your investment will pay off.
Afficher cette discussion
19
117
685

Anecdote: tank detection

In the first years of artificial neural networks, a perceptron was trained to detect images containing tanks. Its results were quite good...

Anecdote: tank detection

In the first years of artificial neural networks, a perceptron was trained to detect images containing tanks. Its results were quite good...

... but in fact images containing tanks were acquired during sunny days, while images without tanks were shot with overcast weather. The network was simply detecting lighter images!

Anecdote: tank detection

In the first years of artificial neural networks, a perceptron was trained to detect images containing tanks. Its results were quite good...

... but in fact images containing tanks were acquired during sunny days, while images without tanks were shot with overcast weather. The network was simply detecting lighter images!

This anecdote might be a urban legend, but nevertheless is a good illustration of the problems one might run into during database preparation. More information available from:

`https://www.gwern.net/Tanks`

What quality is needed for the ground-truth?

- Deep learning models tend to be robust with respect to ground-truth errors
- In the case of segmentation, you do not need a pixel-precision high quality segmentation [Heller et al., 2018]

Preprocessing

- Standard statistical preprocessing: not always useful, and sometimes problematic, when applied to images. It is often enough to divide by 255!
- Use other preprocessing only if really necessary.

Data augmentation

- Geometrical transformations: similarities
- Elastic transformations
- Noise
- Grey level or colour modifications
- Specific methods: articulated objects, ...

Example: plankton classification

Plankton classification: hundred classes - a few dozen examples per class.

Data augmentation:

- Geometric transformations
- Detect joints and simulate their functioning

Credits: Kaggle plankton classification challenge (https://www.kaggle.com)

Using simulated data

- Using simulated data is convenient...

Using simulated data

- Using simulated data is convenient...
- ... but it has to be as similar as possible to the real data

Using simulated data

- Using simulated data is convenient...
- ... but it has to be as similar as possible to the real data
- A transfer learning method with real data will probably be necessary

Using simulated data

- Using simulated data is convenient...
- ... but it has to be as similar as possible to the real data
- A transfer learning method with real data will probably be necessary
- Your test data should be real

Contents

Architecture choice

- Begin with a standard architecture

Architecture choice

- Begin with a standard architecture
    - Classification problem: VGG, GoogLeNet, ResNet...

Architecture choice

- Begin with a standard architecture
  - Classification problem: VGG, GoogLeNet, ResNet...
  - Semantic segmentation or image transformation problem: U-Net

Architecture choice

- Begin with a standard architecture
    - Classification problem: VGG, GoogLeNet, ResNet...
    - Semantic segmentation or image transformation problem: U-Net
    - Instance segmentation: Mask R CNN

Architecture choice

- Begin with a standard architecture
  - Classification problem: VGG, GoogLeNet, ResNet...
  - Semantic segmentation or image transformation problem: U-Net
  - Instance segmentation: Mask R CNN
- If you are dealing with a complex problem, start with pre-learned weights and use transfer learning to adapt them to your application

Architecture choice

- Begin with a standard architecture
  - Classification problem: VGG, GoogLeNet, ResNet...
  - Semantic segmentation or image transformation problem: U-Net
  - Instance segmentation: Mask R CNN
- If you are dealing with a complex problem, start with pre-learned weights and use transfer learning to adapt them to your application

Architecture choice

- Begin with a standard architecture
  - Classification problem: VGG, GoogLeNet, ResNet...
  - Semantic segmentation or image transformation problem: U-Net
  - Instance segmentation: Mask R CNN
- If you are dealing with a complex problem, start with pre-learned weights and use transfer learning to adapt them to your application

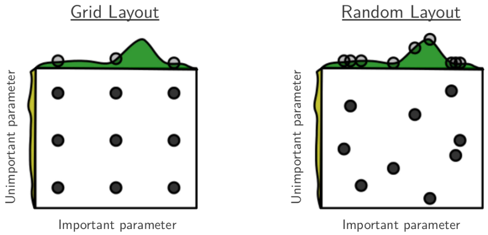
It is interesting to note that the rate of publication of new architectures tends to decrease.

Optimizing your model

- Choose an optimizer
- Use regularization (L_1, L_2, dropout, noise layer ...)
- Add batch normalization if convergence is difficult

Hyperparameters tuning

- Manual tuning: might work if the number of parameter is small and the experience of the developer/researcher high
- Automatic tuning:
  - Grid search
  - Random search
  - Population based approaches
  - Etc.



Credits: [Bergstra and Bengio, 2012]

Computing power

DL became feasible in practice thanks to the use of Graphical Processing Units (GPU). Beyond theoretical research on the subject, to work with DL you need specific hardware:

- CPUs: with many of them, and using libraries that allow parallelization, this could be a solution - in practice, it is seldom done.
- GPUs: this is the most common solution adopted for deep learning.
- TPU: Tensor Processing Units are integrated circuits specifically developed by Google for deep learning.

Computing power

- DL research and development is extremely computationally time-consuming.
- However, running predictions with an already optimized model is much faster

Contents

Problem formulation

In computer vision:

- Databases can be huge, requiring substantial computing power and making learning complex
- In many practical applications the learning data base can be small

Transfer learning brings a solution to these problems.

Definitions [Pan and Yang, 2010]

Domain and task

- A domain D is a probability space $(X,\ P)$, where X is finite.
- Given a domain $D = (X, P)$, a task T consists of two components: a label space $\mathcal{Y}$ and a function $f : X \to \mathcal{Y}$, that is only known on a training set $\{(x_i, y_i), 1 \leq i \leq n, x_i \in X, y_i \in Y\}$.

Definitions [Pan and Yang, 2010]

Domain and task

- A domain D is a probability space (X, P), where X is finite.
- Given a domain $D = (X, P)$, a task T consists of two components: a label space $\mathcal{Y}$ and a function $f : X \to \mathcal{Y}$, that is only known on a training set $\{(x_i, y_i), 1 \leq i \leq n, x_i \in X, y_i \in Y\}$.

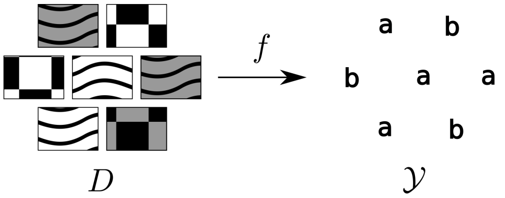
Transfer learning

Let us consider

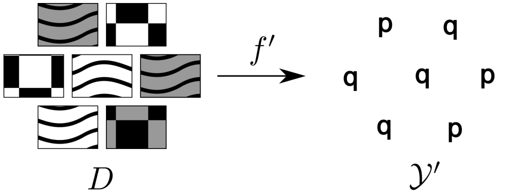
- a *source* domain D_S and a task T_S on that domain, and
- a *target* domain D_T and a task T_T on that domain.

Transfer learning from (D_S, T_s) to (D_T, T_T), where $D_S \neq D_T$ or $T_S \neq T_T$, consists in using the knowledge in (D_S, T_s) to improve the learning of task T_T.

Types of transfer learning: inductive

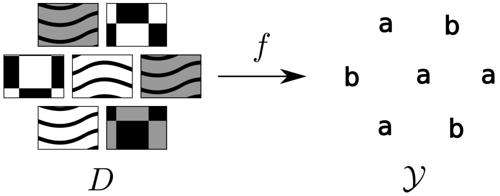


Initial configuration

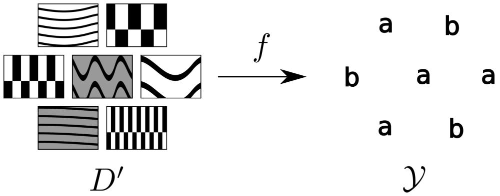


Inductive transfer learning

Types of transfer learning: transductive

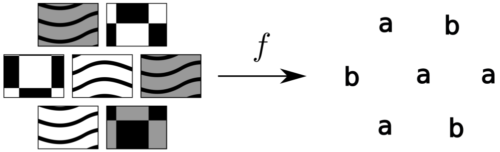


Initial configuration



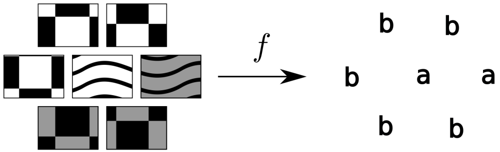
Transductive transfer learning

Types of transfer learning: transductive homogeneous



D $\mathcal{Y}$

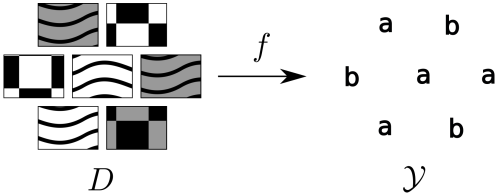
Initial configuration



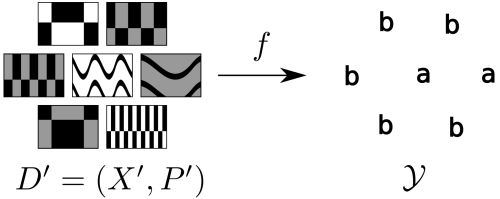
$D' = (X, P')$ $\mathcal{Y}$

Transductive homogeneous transfer learning

Types of transfer learning: transductive heterogeneous



Initial configuration



Transductive heterogeneous transfer learning

Transfer learning through fine-tuning

Suppose that thanks to a training set $(X_0, \mathcal{Y}_0)$ a model f_{θ_0} has been learnt.
Transfer learning through fine-tuning consists in learning another model f_θ from a training set $(X, \mathcal{Y})$ using as starting point f_{θ_0}.
For transfer learning to work, both training sets have to be somehow related and compatible.

General procedure

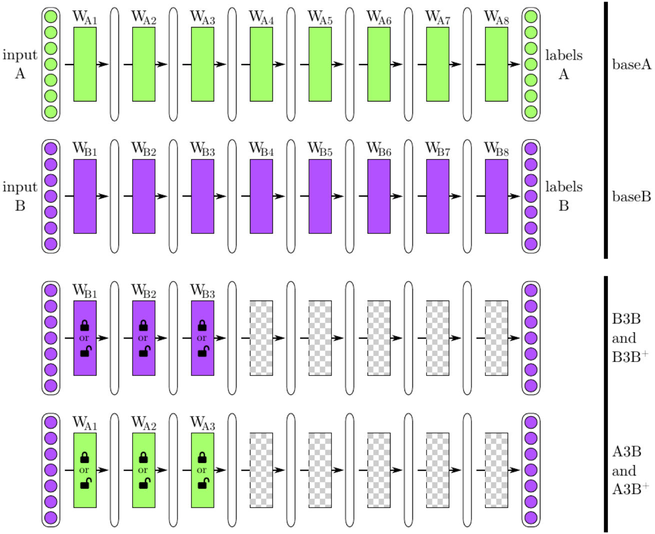
- Choose and existing model, optimized on a data base such as ImageNet. It should be able to process the new data
- Remove the last layers of the model and replace them with layers adapted to the task at hand
- Fine-tune the resulting model
  - Note that some pre-trained layers are often *frozen*

A reference paper [Yosinski et al., 2014]

Jason Yosinski, Jeff Clune, Yoshua Bengio, Hod Lipson. **How transferable are features in deep neural networks?** Neural Information Processing Systems, 2014.

The authors devised experiments on the ImageNet database in order to evaluate different fine-tuning strategies and improve our understanding of these methods.

Experiments configuration

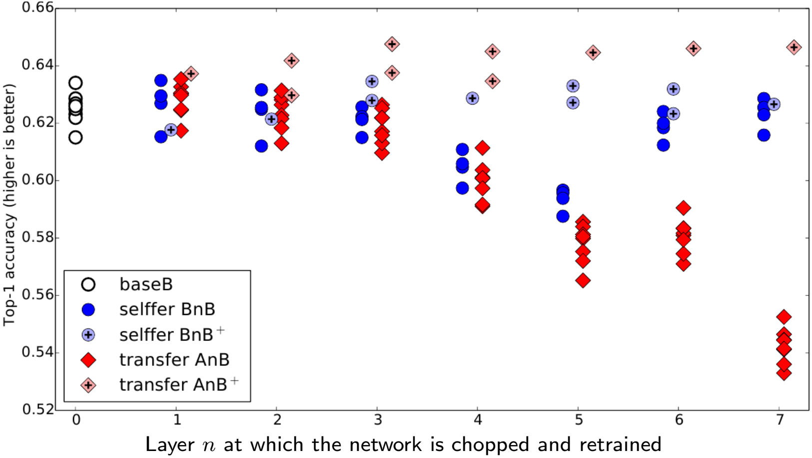


First experiment configuration

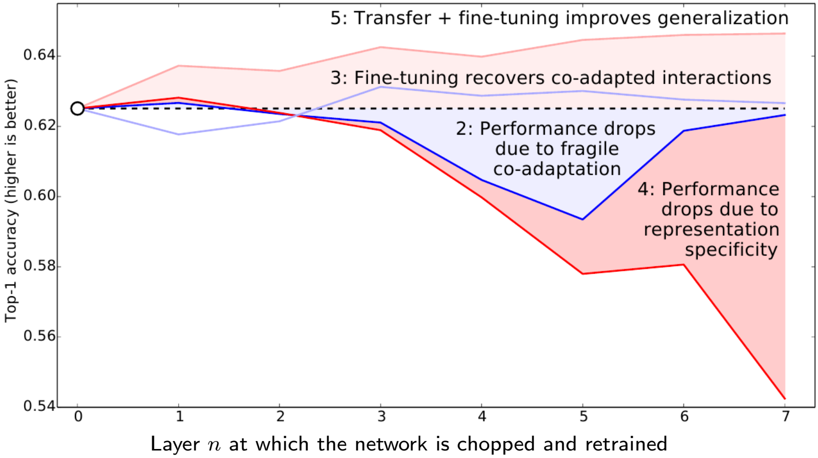
ImageNet data set split

- Set A: 500 randomly selected classes
- Set B: 500 other classes

First experiment results



First experiment results

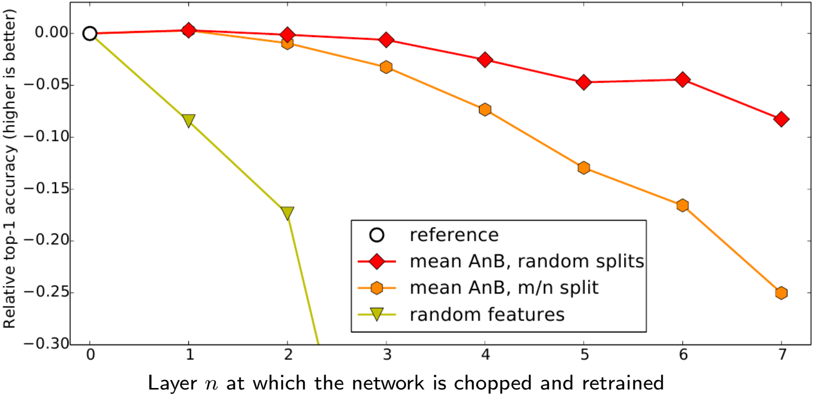


Second experiment configuration

ImageNet data set split

- Set A: Man-made objects (551 classes)
- Set B: Natural entities (449 classes)

Second experiment results



Experiments conclusions

- Two separate issues with transfer learning:
  - Specificity of high level features
  - Co-adaptation of neurons on neighboring layers
- Transfer learning is less efficient when the sets are more dissimilar (at least when the pre-trained weights are frozen)
- Generalization performance can be boosted by transfer learning

Conclusion

- You now know the basics to tackle many computer vision problems with deep learning.
- But there are many other topics of interest:
  - Attention mechanisms, transformers
  - Self-supervision, low supervision
  - Non supervised methods, including autoencoders and generative adversarial networks
  - Recurrent networks
  - Anomaly detection
  - etc.

References I

[Bergstra and Bengio, 2012] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

[Heller et al., 2018] Heller, N., Dean, J., and Papanikolopoulos, N. (2018). Imperfect Segmentation Labels: How Much Do They Matter? In Stoyanov, D., Taylor, Z., Balocco, S., Sznitman, R., Martel, A., Maier-Hein, L., Duong, L., Zahnd, G., Demirci, S., Albarqouni, S., Lee, S.-L., Moriconi, S., Cheplygina, V., Mateus, D., Trucco, E., Granger, E., and Jannin, P., editors, *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, Lecture Notes in Computer Science, pages 112–120, Cham. Springer International Publishing.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359. Conference Name: IEEE Transactions on Knowledge and Data Engineering.

[Yosinski et al., 2014] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 3320–3328, Cambridge, MA, USA. MIT Press.