Practical recommendations

E. Decencière

MINES ParisTech
PSL Research University
Center for Mathematical Morphology

Contents

Contents

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 a lot 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 a lot 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 a lot and read the reports by other practitioners

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

General recommendations

- Familiarize yourself with the problem and the data

General recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images

General recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it

General recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- Analyze the results on the validation data (look at the images!)

General recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- Analyze the results on the validation data (look at the images!)
- Beware of over-fitting! Envision regularization methods.

General recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- 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 recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- 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 recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- 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 recommendations

- Familiarize yourself with the problem and the data
- Choose the right representation for your images
- Choose an architecture and train it
- 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

Modeling your problem

Cast your problem into a convenient representation.

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

Modeling 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 images)

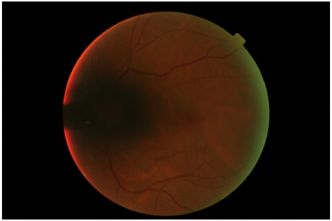
Modeling your problem

Cast your problem into a convenient representation.

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

Example: eye fundus image quality

Good quality



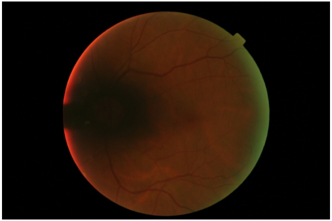
Low quality

- Quality criterion: are the macula and peripheral vessels visible?

Credits: images from OPHDIAT database

Example: eye fundus image quality

Good quality



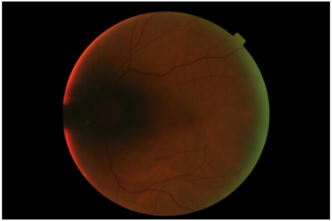
Low quality

- Quality criterion: are the macula and peripheral vessels visible?
- First solution: regression (center of macula position)

Credits: images from OPHDIAT database

Example: eye fundus image quality

Good quality



Low quality

- Quality criterion: are the macula and peripheral vessels visible?
- First solution: regression (center of macula position)
- Second solution: predict macula mask

Credits: images from OPHDIAT database

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

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)

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

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
- 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, of course, 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.

Contents

Optimizing your model

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

Hyperparameters tuning

Your options are:

- Manual tuning: works well if the number of parameter is small and the experience of the developer/researcher high
- Automatic tuning (grid search, random search): computationally time-consuming

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. In practice, you need many of them. Note that you can either buy them or rent them online.
- 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, a simple CPU is enough in most cases for running a given, already optimized, model.

References I