Deep Learning - COSC2779

Practical methodology & Hyperparameter tuning

Dr. Ruwan Tennakoon

RMIT UNIVERSITY

Aug 23, 2021

Reference: *Chapter 11: Ian Goodfellow et. al., "Deep Learning", MIT Press, 2016.*

Outline

Revision

RMIT UNIVERSITY

A look back at what we have learned:

- Deep neural network Building blocks:
  - **Week2**: Feed forward NN Model and cost functions.
  - **Week3**: Optimising deep models challenges and solutions.
  - **Week4**: Convolution neural network: for data with spatial structure.
  - **Week7-8**: Recurrent neural network: for data with sequential structure.
- Case study:
  - **Week5**: Famous networks for computer vision applications

A look back at what we have learned:

- Deep neural network Building blocks:
  - **Week2**: Feed forward NN Model and cost functions.
  - **Week3**: Optimising deep models challenges and solutions.
  - **Week4**: Convolution neural network: for data with spatial structure.
  - **Week7-8**: Recurrent neural network: for data with sequential structure.
- Case study:
  - **Week5**: Famous networks for computer vision applications

Successfully applying deep learning techniques requires more than just a good knowledge of what algorithms exist and the principles that explain how they work.

Outline

A good machine learning practitioner:

- Know how to choose an algorithm for a particular application.
- Know how to setup experiments and use the results to improve a machine learning system.

Practical Methodology

A good machine learning practitioner:

- Know how to choose an algorithm for a particular application.
- Know how to setup experiments and use the results to improve a machine learning system.
  - Gather more data?
  - Increase or decrease model capacity?
  - Add or remove regularizing features?
  - Improve the optimization of a model?
  - Debug the software implementation of the model

A good machine learning practitioner:

- Know how to choose an algorithm for a particular application.
- Know how to setup experiments and use the results to improve a machine learning system.
  - Gather more data?
  - Increase or decrease model capacity?
  - Add or remove regularizing features?
  - Improve the optimization of a model?
  - Debug the software implementation of the model

All these operations are at the very least time consuming to try out, so it is important to be able to determine the right course of action rather than blindly guessing.

Typical Procedure: Model development

- **Determine your goals**: Error metric and target value. Problem dependent.
- **Default Baseline Model**: Identify the components of end-to-end pipeline including - Baseline Models, cost functions, optimization.
- **Setup the diagnostic instrumentation**: Setup the visualizers and debuggers needed to determine bottlenecks in performance (overfitting/underfitting, weight changes, learned features, etc.).
- **Make incremental changes**: Repeatedly make incremental changes such as gathering new data, adjusting hyperparameters, or changing algorithms, based on specific findings from your instrumentation.

Outline

Identify Error Metric

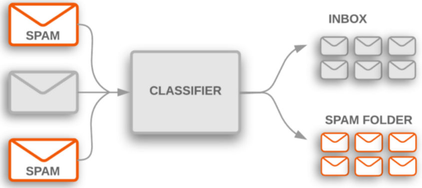
Performance metrics are usually different from the cost function used to train the model.

Performance metrics needs to be more intuitive.

- Cost function - Cross-entropy: Suited for gradient based optimization. Value not easy to interpret.
- Performance metric - Accuracy: Easy to interpret.

The performance metric depends on the specifics of the problem: One type of error might be important than the other. Classification vs regression vs clustering. Presence of outlier/noise in data ...

SKLearn Documentation Metrics and scoring



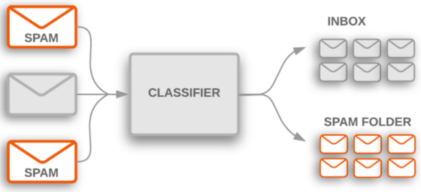
Binary classification problem.

- False positive (Type I): Important email in spam folder.
- False negative (Type II): Spam email in inbox.

Accuracy is not a good measure here. Type I error is more important that Type II.

Precision is the fraction of spam detection by the model that were correct.

Recall is the fraction of true spam emails that were detected.



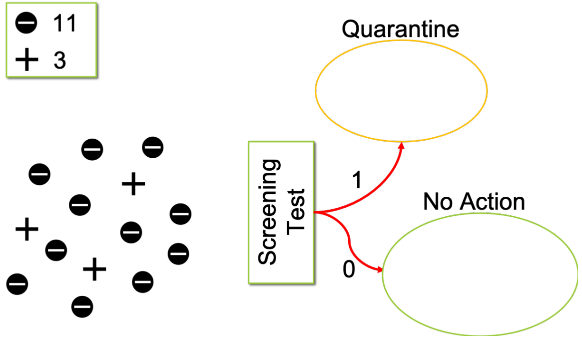
Binary classification problem.

- False positive (Type I): Important email in spam folder.
- False negative (Type II): Spam email in inbox.

Accuracy is not a good measure here. Type I error is more important that Type II.

Precision is the fraction of spam detection by the model that were correct.

Recall is the fraction of true spam emails that were detected.

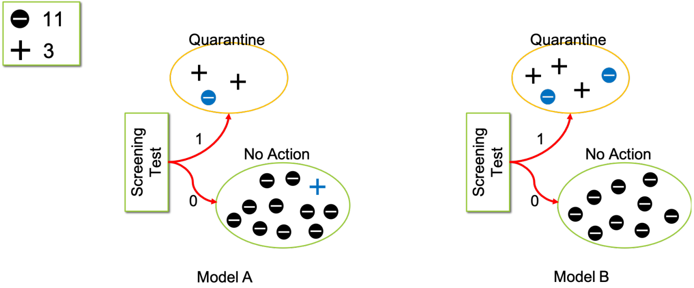


Binary classification problem.

Screening done at airport:

- False positive (Type I): Non-COVID patients detected as COVID by test.
- False negative (Type II): True COVID patient not detected by system.

Case study: Screening COVID-19 at an Airport



Binary classification problem.

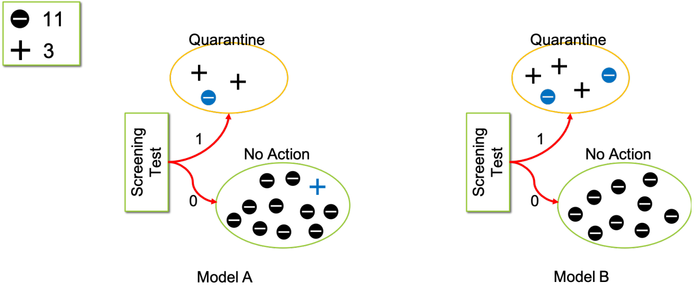
- False positive (Type I): Non-COVID patient sent to quarantine.
- False negative (Type II): COVID patient sent home.

Accuracy is not a good measure here. Type II error is more important that Type I.

Precision is the fraction of Covid patients identified by the model that were correct.

Recall is the fraction of true true Covid patients that were detected.

Case study: Screening COVID-19 at an Airport



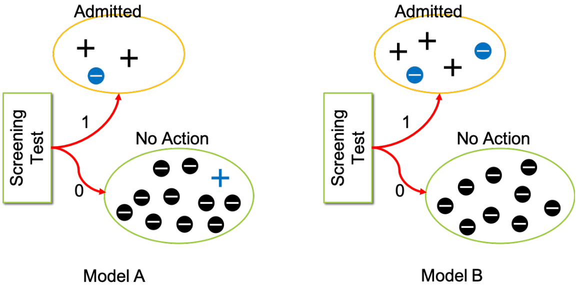
Binary classification problem.

- False positive (Type I): Non-COVID patient sent to quarantine.
- False negative (Type II): COVID patient sent home.

Accuracy is not a good measure here. Type II error is more important that Type I.

Precision is the fraction of Covid patients identified by the model that were correct.

Recall is the fraction of true true Covid patients that were detected.



Binary classification problem.

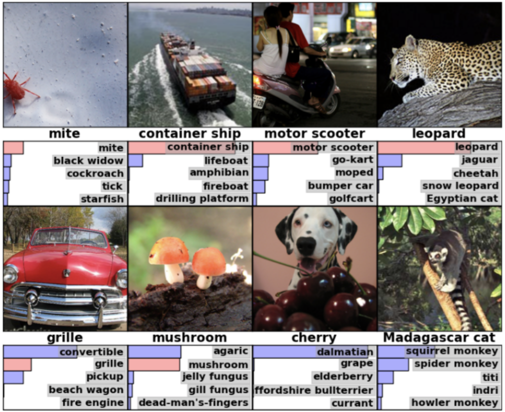
- False positive (Type I): Non-COVID patient admitted to hospital.
- False negative (Type II): COVID patient sent home.

Accuracy is not a good measure here. Not many Covid+ patients. Can have large accuracy when predicting every person as non-Covid.

Precision is the fraction of Covid patients identified by the model that were correct.

Recall is the fraction of true true Covid patients that were detected.

F1-Score Balance between precision (p) and recall (r). $\mathrm{F1-Score} = \frac{2pr}{p+r}$



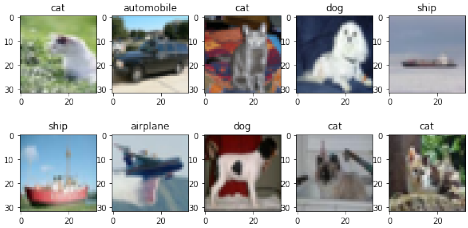
Top-5 score, you check if the target label is one of your top 5 predictions (the 5 ones with the highest probabilities)

Establishing a Target Value

We also need to establish a target value for out selected metric:

- Discussion with client or application expert and modelling.
  - e.g. In Covid screening at airport: If we have a recall of $\langle x \rangle$ and the number of patients arriving is $\langle y \rangle$, then we can achieve a r-value of $\langle z \rangle$.
- Literature review.
  - e.g. Imagenet challenge a top-5 error rate of 5.1. Human performance.

CIFAR10: The data set is comprised of 60,000 32x32 pixel color photographs of objects from 10 classes.



- What are the appropriate metrics?
- What will be the target values?

These Questions are usually answered independently of the techniques you are going to use.

Outline

Establishing Baseline Model

Is deep learning required to solve your problem?

What category of deep models should be tried:

- Supervised learning with fixed-size vectors: Deep Feed-forward models
- Input has topological structure: Use CNN.
- Input or Output is a sequence: LSTM or GRU (will be discussed in future).

CIFAR10?

Establishing Baseline Model

If your task is similar to another task that has been studied extensively, you will probably do well by first copying the model and algorithm that is already known to perform best on the previously studied task.

If the problem is novel (research). Then start with a simple model from the appropriate category that adhere to the well known norms of that category.

Select a reasonable optimizer: SGD with momentum, or ADAM for image classification.

Select a reasonable cost function: Problem dependent. In deep learning usually needs to be differentiable.

Outline

Setup the Diagnostic Instrumentation

RMIT UNIVERSITY

This includes setting up TensorBoard etc. We discussed thee in the lab.

Also include setting up data set folds appropriately: discussed in week 1.

Outline

Hyper Parameter Tuning

Most deep learning algorithms come with several hyper parameters that control many aspects of the algorithm's behavior.

- Affect the performance of the model.
- Affect the time and memory cost of running the algorithm.

Approaches to choosing these hyperparameters:

- **Manually** selecting hyperparameters. Requires understanding of what each hyperparameter do and how a model behaves when those are changed.
- **Automatic** hyperparameter selection. Reduce the need to understand hyperparameters.

Manual Hyperparameter Tuning

The primary goal of manual hyperparameter search is to adjust the effective capacity of the model to match the complexity of the task. *Bias Vs Varience.*

Hyperparameter search also involve selecting the parameters of the optimization procedure: controls the effective capacity of the model.

Mostly done by observing the training curves.

- Gather more data?
- Increase or decrease model capacity?
- Add or remove regularizing features?
- Improve the optimization of a model?

The primary goal of manual hyperparameter search is to adjust the effective capacity of the model to match the complexity of the task. *Bias Vs Varience.*

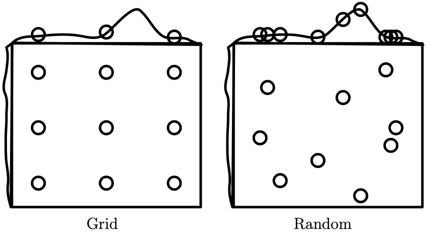


Image: Goodfellow, 2016.

hyperparameter optimization algorithms often have their own hyperparameters, such as the range of values that should be explored for each of the learning algorithm's hyperparameters.

Debugging Strategies

- *Visualize the model outputs*: e.g When training a model to detect objects in images, view some images with the detections.
- *Visualize the worst mistakes*.
- *Fit a tiny data set*: If you have high error on the training set, determine whether it is due to genuine underfitting or due to a software defect.
- *Monitor histograms of activations and gradient*: It is often useful to visualize statistics of neural network activations and gradients, collected over a large amountof training iterations (maybe one epoch).

Outline

Outline

1. Practical Methodology
   - Determine Your Goals
   - Default Baseline Model
   - Setup the Diagnostic Instrumentation
   - Make Incremental Changes
   - Debugging Strategies
2. Demo
3. Assignment 1: Discussion

Things to Consider

- Have you selected appropriate baseline model with justification?
- Have you setup the evaluation framework correctly and justified?
- Did you improve the model based on evidence (make appropriate decisions)?
- Did you consider task specific issues?
- Evidence based ultimate judgment (Not just best MSE $\rightarrow$ Best model).
- Did you identify the issues with applying your model to real scenarios through model/output investigation?

Note: To get $\geq$ DI for approach, you need to demonstrate skills that goes beyond what is in lectures and labs.