Training neural nets: A methodical approach

Sayak Paul

“ML/AR Developer Day”

Organized by GDG Kolkata and DSC HIT

May 30, 2019 (Kolkata, India)

This talk is about

- Discussing the intricacies of training neural nets in general and their importance
- Discussing the most common pitfalls that are occurred during training neural nets
- Discussing a few tips and tricks to train neural nets in a holistic way

Slides available at http://bit.ly/MLAR19

This talk is *not* about

- Showing how to design a super complex architecture and get it to fail
- Showing the modern practices presented at ICLR, NIPS (and others)
- Showing any specific application of deep learning

Slides available at http://bit.ly/MLAR19

Motivation

As a result, (and this is reeaally difficult to over-emphasize) a "fast and furious" approach to training neural networks does not work and only leads to suffering. Now, suffering is a perfectly natural part of getting a neural network to work well, but it can be mitigated by being thorough, defensive, paranoid, and obsessed with visualizations of basically every possible thing. The qualities that in my experience correlate most strongly to success in deep learning are patience and attention to detail. - Andrej Karpathy

Slides available at http://bit.ly/MLAR19

Central problem

Neural networks can fail silently!

- Implementation bugs
- Model's sensitivity towards hyperparameter choices
- Dataset construction and others

Slides available at http://bit.ly/MLAR19

What can we do about it?

- Be thorough
- Be pessimistic
- Be interrogative
- Be obsessed with *prints* and *plots*

Source

Slides available at http://bit.ly/MLAR19

The central ideas

- ★ Becoming one with the data
  - Know everything possible about your data
  - Be insanely careful about setting up train/validation/test splits
  - Make sure the model does not learn anything about the order (until required)
- ★ Embracing simplicity
  - Human baselines (Bayesian error)
  - Working with a small subset of data and simple model
- ★ Gradually ramping up model complexity
  - Add more layers
  - Go for a more complex architecture

Slides available at http://bit.ly/MLAR19

The central ideas (contd.)

- ★ Evaluate, overfit, dropout and take steps
  - Data augmentation (not always)
  - Regularize
  - Does the model overfit as you expected?
  - What the model is learning btw? (Visualize the activations)
  - ... a few more
- ★ Hyperparameter tuning and going beyond
  - Hyperparameter tuning with random search
  - Training networks with modern practices
  - Model stacking and ensembling

Slides available at http://bit.ly/MLAR19

Let’s discuss a few of this!

Making friends with data

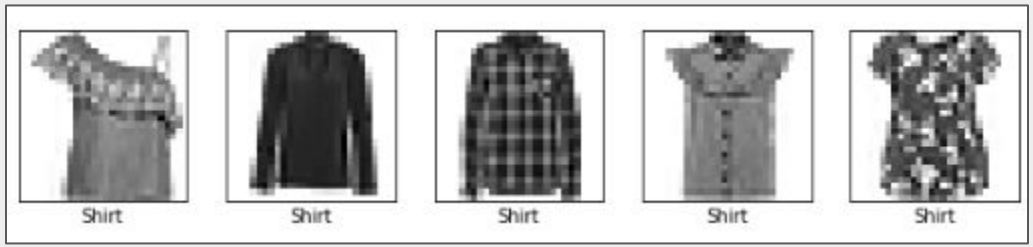
- Is the data representative of the problem statement?
- How good/bad is the data?
- Do you know everything you need to know about the data?
  - How are they labeled?
  - Is there any class imbalance? (Supervised learning)
  - Is there any label noise?
- How would you preprocess the data?
- How would you construct the train/validation/test splits?

Slides available at http://bit.ly/MLAR19

	Class	Observations
0	9	6000
1	0	6000
2	3	6000
3	2	6000
4	7	6000
5	5	6000
6	1	6000
7	6	6000
8	4	6000
9	8	6000

Class distribution

Random batches of data



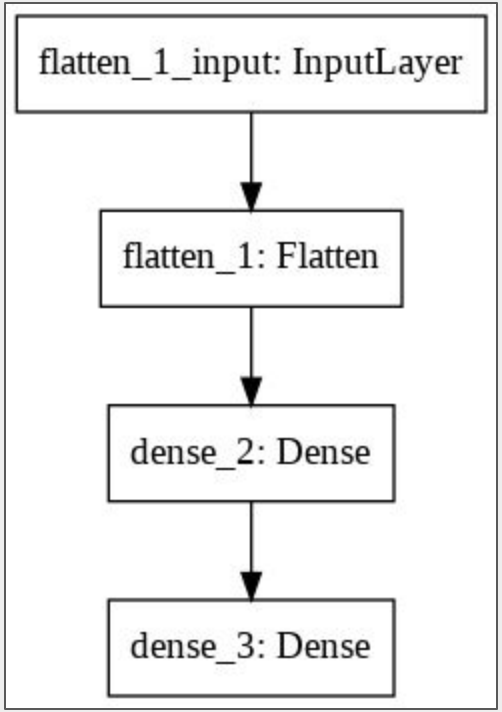
Label noise

Slides available at http://bit.ly/MLAR19

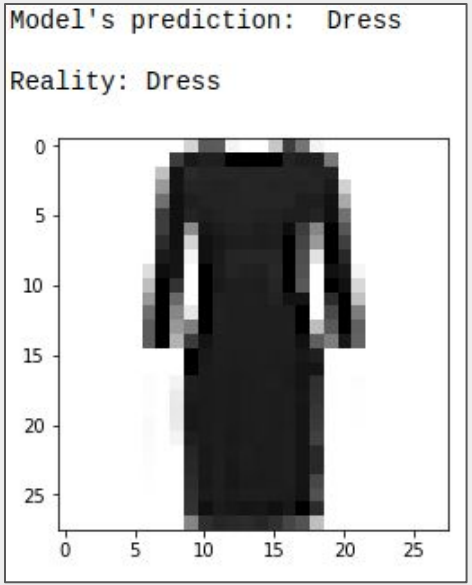
“Simplicity is the ultimate sophistication”

- !(Data augmentation + ResNet152)
- Fix every random seed
- Start with a simple model which makes least no. of assumptions
  - *Don’t use any default hyperparameter values*
- Start with a small subset of data that can fit in memory
- Manually annotate a subset of your data and record the prediction dynamics of the model
- Go for a human interpretable metric to check model’s performance

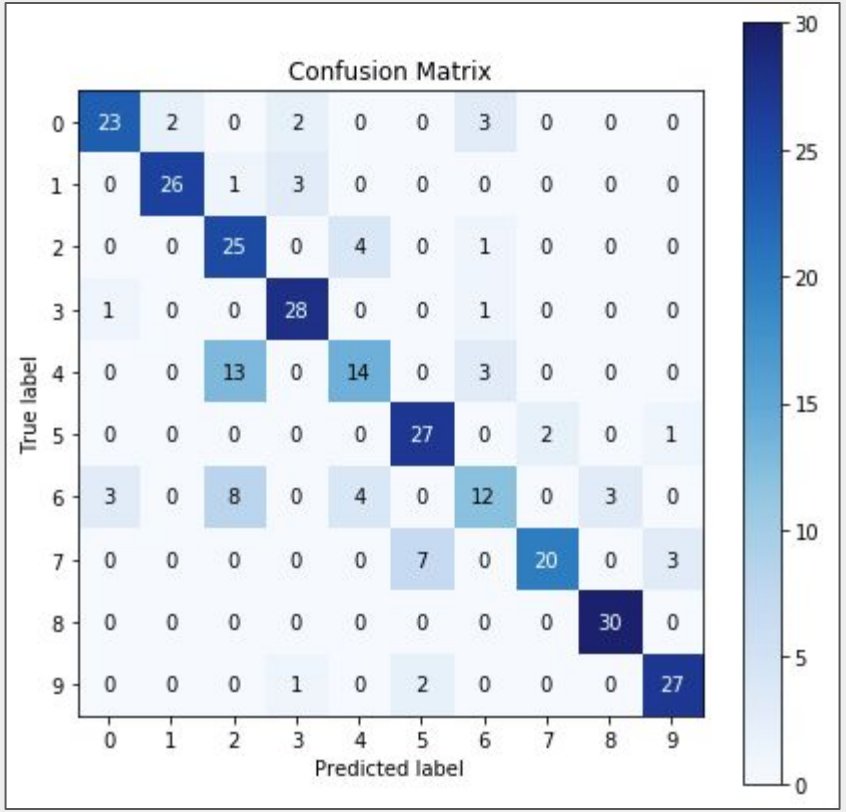
Slides available at http://bit.ly/MLAR19



Simple MLP



Model's performance



Model's confusion

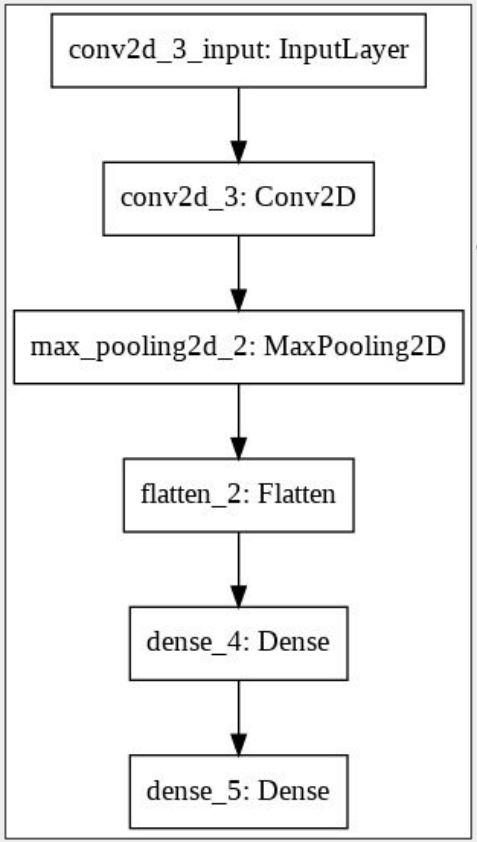
Slides available at http://bit.ly/MLAR19

Going for the *(a + ib)* landscape

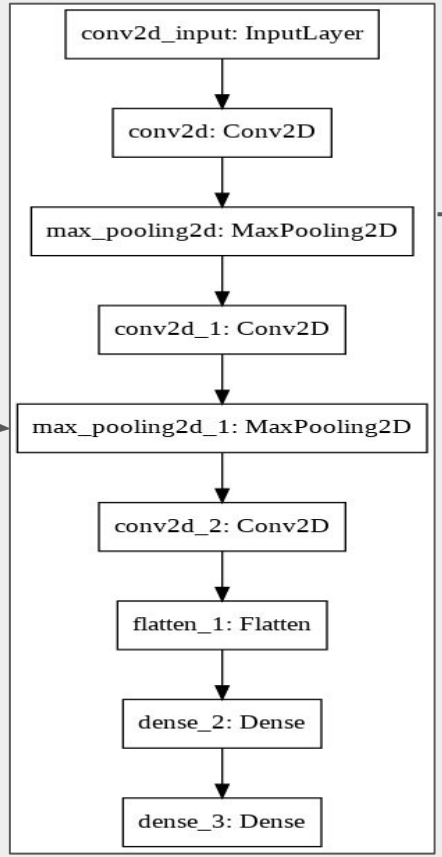
- We know that simple models are not going to work for the problem
- The idea is to ***gradually*** increase the complexity
- Visualize the activations happening at each layer
- Data augmentation if needed
- Evaluate, deepen, repeat

Slides available at http://bit.ly/MLAR19

Complexity as a function of order



Visualize performance



Visualize performance

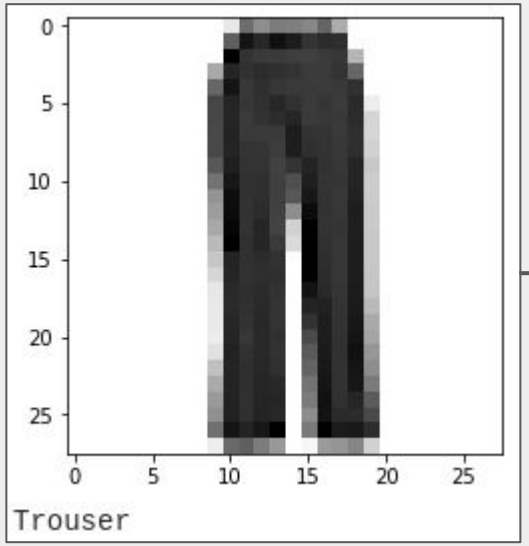
Deep learning *is a shout in the void and the oblivion is inevitable*

Slides available at http://bit.ly/MLAR19

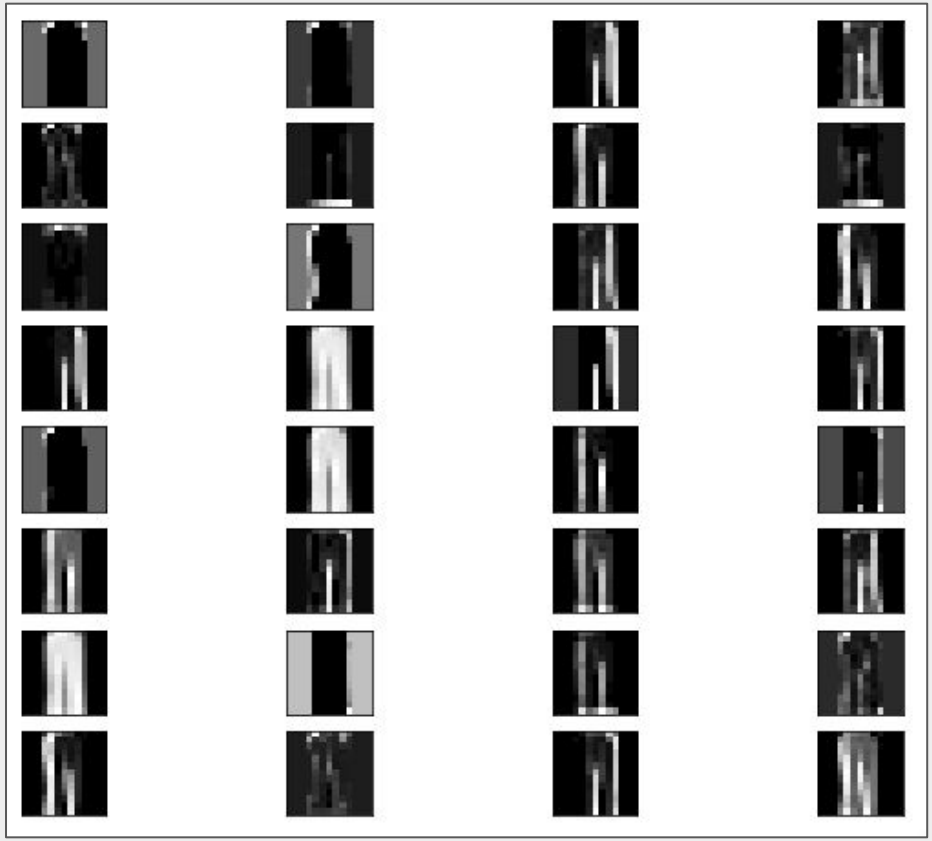
Complexity as a function of order (contd.)

- Does the more complex model overfit? (It should if it is large enough)
- Regularize appropriately if it overfits
- Does the training loss decrease as expected?

Understanding the model via plots



A sample test image

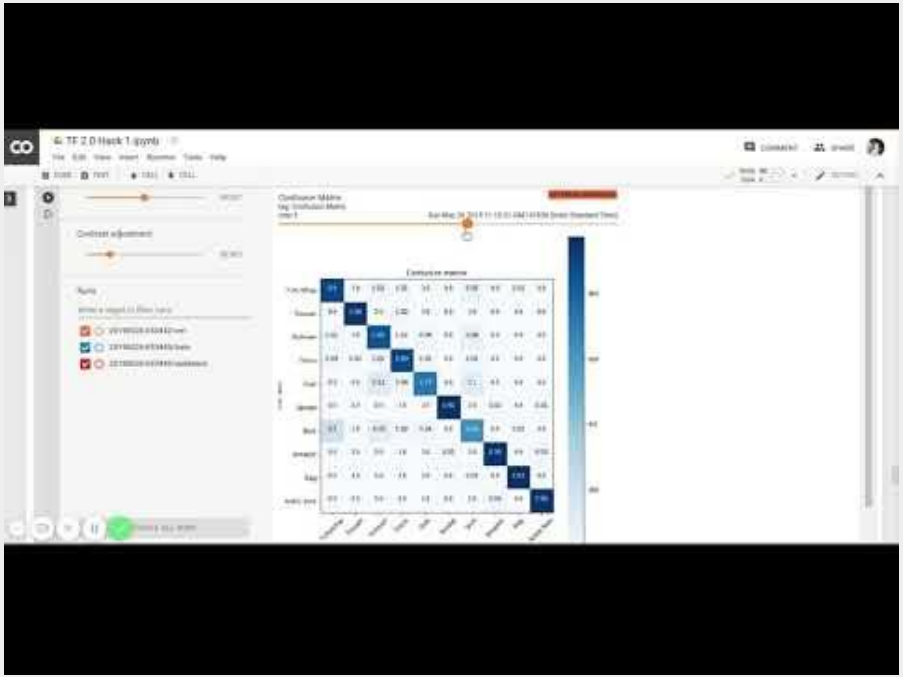


Through the model's eyes

Slides available at http://bit.ly/MLAR19

Evaluation with human friendly numbers

1. See how your model is performing batch wise



Slides available at http://bit.ly/MLAR19

Evaluation with human friendly numbers (contd.)

2. Monitor training and val/test loss very carefully
    a. Where the validation loss stops to decrease
    b. Set up early stopping
3. Where the model is getting confused?
    a. Is it getting confused by the nature of laws?
        i. If not then there is something off

Slides available at http://bit.ly/MLAR19

Wrapping up

- ★ What we covered so far?
  - The idea of knowing your data from a practical viewpoint
  - The power of starting simple and adding more lego blocks gradually
  - Monitoring model's performance in every way possible
- ★ What we did not cover?
  - Data augmentation as a generalization weapon
  - Use pre-trained models (the soft-corner of every DL practitioner probably)
  - Use modern practices like discriminative training, mixed precision, knowledge distillation and so on

Maybe next time!

Slides available at http://bit.ly/MLAR19

References

- ★ A Recipe for Training Neural Networks by Andrej Karpathy
- ★ Troubleshooting Deep Neural Networks by Josh Tobin
- ★ Structuring Machine Learning Projects by Andrew Ng

Slides available at http://bit.ly/MLAR19

See you next time

Find me here:
https://bit.ly/sayakpaul

Thank you very much :)