Introduction to Machine Learning

An overview for physicists

Lecture 9
In this lecture

● Recurrent neural networks (RNN)
  ○ Gated recurrent units (GRU)
  ○ Long short-term memories (LSTM)

● Generative adversarial networks (GAN)

● Extra topics

● Conclusion
Recurrent neural networks

What if we want to model a time series?

We want to reuse weights, but cannot use the whole data as input.

Can we keep information in the network? → Recurrent Neural Network

Very used in natural language processing and similar tasks.

May have problems with vanishing gradients...
Gated recurrent units

“Memory units” to keep track of long term correlations.

**Update gate:**
\[ z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z) \]

**Reset gate:**
\[ r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r) \]

**New hidden state:**
\[ h_t = (1 - z_t)h_{t-1} + z_t \tanh(W_h x_t + U_h (r_t h_{t-1}) + b_h) \]
Long short-term memories

Other popular gated units with more parameters:

Cell state, forget gate, input gate, output gate

\[
\begin{align*}
    f_t &= \sigma (W_f \cdot [h_{t-1}, x_t] + b_f) \\
    i_t &= \sigma (W_i \cdot [h_{t-1}, x_t] + b_i) \\
    \tilde{C}_t &= \tanh (W_C \cdot [h_{t-1}, x_t] + b_C) \\
    C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \\
    o_t &= \sigma (W_o \cdot [h_{t-1}, x_t] + b_o) \\
    h_t &= o_t \cdot \tanh (C_t)
\end{align*}
\]
Generative adversarial networks (GANs)

Coupled training of competing generative and discriminating networks.
Generative adversarial networks (GANs)
Extra topics
Evolutionary algorithms

Optimization algorithms:

1. Generate a population of “individuals” (states)

2. Evaluate fitness

3. Produce new individuals from the fittest (crossover/mutation) and repeat

Lots of freedom in defining fitness and breeding methods, works best in genetic algorithms, also used to optimize NN.
Decision tree learning and random forests

A decision tree performs classification/prediction based on splitting of input variables.

In random forests, many decision trees are built out of samples of the data and/or features, then their predictions are averaged.

They are good automatic models and the random component prevents overfitting.
## Transfer learning

In some NN architecture, it is possible to train part of a network on one problem, then reuse the weights to train for another problem.

E.g.: part of a network trained for image recognition can be trained for new categories or new tasks, speeding up considerably the overall training time.

One must be careful whether the features remain meaningful...
Neural networks and ML/data science software

- Tensorflow [C++/python]
- Torch (PyTorch) [Lua/python]
- Caffe [C++/python]
- R
- Python
- C/Fortran
Conclusions

ML is about data, models and probability: manipulating data to find the best representation that extracts the relevant information for your problem.

Problems lie in defining the approximating function (architecture), figure of merit (cost function) and optimization algorithm.

There are several options, with advantages and drawbacks...

Some of it is theory, some is alchemy... keep an open view and find what works.
That's all folks!