Lecture & Leveson Notes

By James Camacho

RNN's

- Process sequences
  - Classify images by series of glimpses of image
  - Generate images one piece at a time (e.g. oil paint simulator)
- Key idea: internal state that updates as sequence is processed
- seq2seq = many to one + one to many, used for language translation
- backprop can take lots of memory, so only store some of the hidden states
- LSTM - i/o/forget/gate gates on how much to change hidden values

Attention

- Finds how aligned each hidden vector is in the current context
- Adds these together
- E.g. with image captioning: Has alignment scores that update every time a letter is outputted by the NN. These give attention weights which multiply with the hidden units, before continuing on in the RNN to get the next letter.
- Attention weights can be used for interpretability.
- Multihead self-attention = what words are connected to each other in the same sentence. Multiheaded lets it learn multiple contexts.
- Transformer = self-attention + residual connection + layer normalization + multilayer perceptron (MLP) + residual connection + one more layer normalization

DL for NLP

- Machines better than humans for some language translations
- Super human performance on some question answer datasets
- Embed words as vector.
- Feed-forward network means bounded context, RNN unbounded (but slow)
- Transformer = positional encoding + input embedding -> multi-head attention, feed forward. Not recurrent. Masks self-attention so words can't look into the future. Parallelizable

- Lots of tricks/fine-tuning to get it working
- Self-attention is quadratic, which can take awhile to compute for long sequences (but can truncate)
- How to get sequence from probabilities? Beam search > greedy or exhaustive search.
- Top-k reduces risk of one bad sample ruining all future words
- seq2seq
  - encoder stack - each input attends to all other inputs
  - decoder stack - each input attends to all inputs and all previous outputs
  - Can get more German -> English data by translating English -> German with another model.
- word2vec = exactly what it says, uses nearby words but bad because it misses context.
- Lots of better ways of doing this.
- Open questions: How to integrate world knowledge, model long documents, multi-task learning, fine-tuning better, do these models really understand language?

Reinforcement Learning

- Supervised learning = learn a function to map data x to labels y
- Unsupervised = no labels!
- RL = Performs actions to get rewards. Try to maximize reward for actions.
- Sees state, takes action, gets reward.
- Cart-pole is common problem
- Markov property = current state completely characterizes state of world (not always true)
- Q function = like q table but a nn
- Reinforce algorithm (policy gradient) = Give distribution of actions in current state, can be used to get future states/actions?
- Actor-critic = actor predicts actions (policy gradient), critic predicts rewards (q function)
- model-based = model the state function, then use something else (like beam search) to pick the best states
- imitation learning = imitate experts (e.g. chess grandmasters)
- inverse RL = imitate reward function, use RL on that
- adversarial learning = pretend to be a real human player
- AlphaStar - Really good AI Starcraft II player.
- Can also use RL to train nondifferentiable neural nets

Leveson Notes

- Start with chemical plant failure.
  - Who's fault? Worker who didn't put in slip valve? People who had tea instead of investigating the alarm? Etc.

  - Answer: No "root cause", the circumstances were the issue.
  - Unsafe conditions, alarms blaring 10-20 times a week, made it so failure would eventually happen.
- Other failures due to circumstances. Sometimes things work out, but when they don't it's not one person's fault for not fixing things.
- Software doesn't fail like humans, but it can be designed poorly to make circumstantial failures inevitable.
- Flawed requirements usually results in failing software.
- We need systems that can handle unknown situations to prevent failures.
- Much harder to analyze modern systems (that use software), need "system theory".
  - In between statistics (rng averages) and flowcharts
  - STAMP is one flowcharty thing but on a higher level that helps people analyze systems. Premise is accidents are caused by lack of control.
  - STPA stands for "Systems Theoretic Process Analysis"
- Leveson bragged about how her model was easier to use and worked just as well/better than the traditional flowchart models.