Microsoft

APAC Machine Learning & Data Science Community Summit

2017년 5월 20일(토)
상암동 누리꿈스퀘어 비즈니스타워 3층

Deep Learning – Fun with TensorFlow

Martin Andrews

Red Cat Labs

Outline

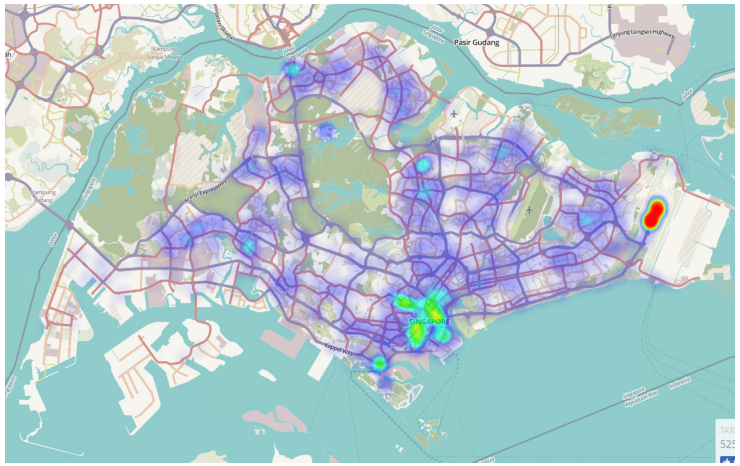
- About me + Singapore community + Workshops
- Something in-the-news :
  - What else I could have chosen for ths talk
- Actual talk content
  - Including lots of code (show of hands?)
- Deep Learning / Data Science human resources
  - Trying to fix the problem in Singapore
- Wrap-up

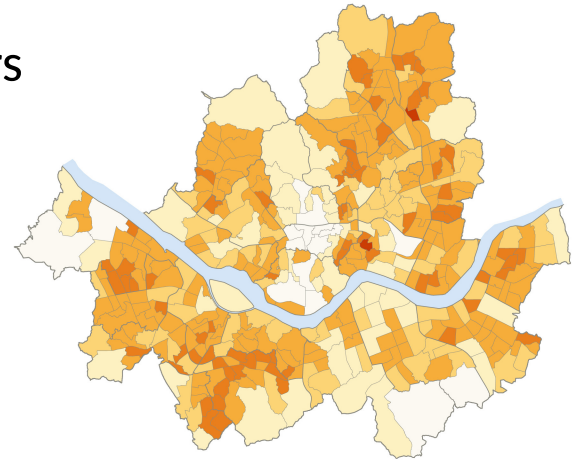
About me

- PhD in Machine Learning *in the 1990s*
- Since then : Finance / Analytics / Startups
  - Moved from NYC to Singapore in September-2013
- 2014 = 'fun' :
  - Machine Learning, Deep Learning, NLP
  - Robots, drones
- Since 2015 = 'serious' :: NLP + Deep Learning
  - & Open Source…
  - & Papers…
  - & Workshops…

Singapore Data Science community

- Singapore is a small, smart city-state on the equator
  - Country has very few natural resources
  - Data Science is seen as a good strategic fit
- Community activities :
  - DataScience SG : 5,100 members
    - Topics : Technical, strategy & marketing
  - PyData SG : 2,500 members
    - Topics : Maybe show code, beginners welcome
  - TensorFlow & Deep Learning SG : 1,000 members
    - Topics : Must show code. Beginners – Advanced
  - PyTorch & DL Group & Workshops…
    - First meeting in July





Deep Learning workshops

- Started at FOSSASIA in 2016
- Problem :
  - Want to teach Deep Learning “Hands on”
  - Machines difficult to set up
  - No WiFi
- Solution :
  - Pre-configured VirtualBox Appliance, loaded with models and data
  - Cross-platform, handed out on USB sticks
- All Open Source…

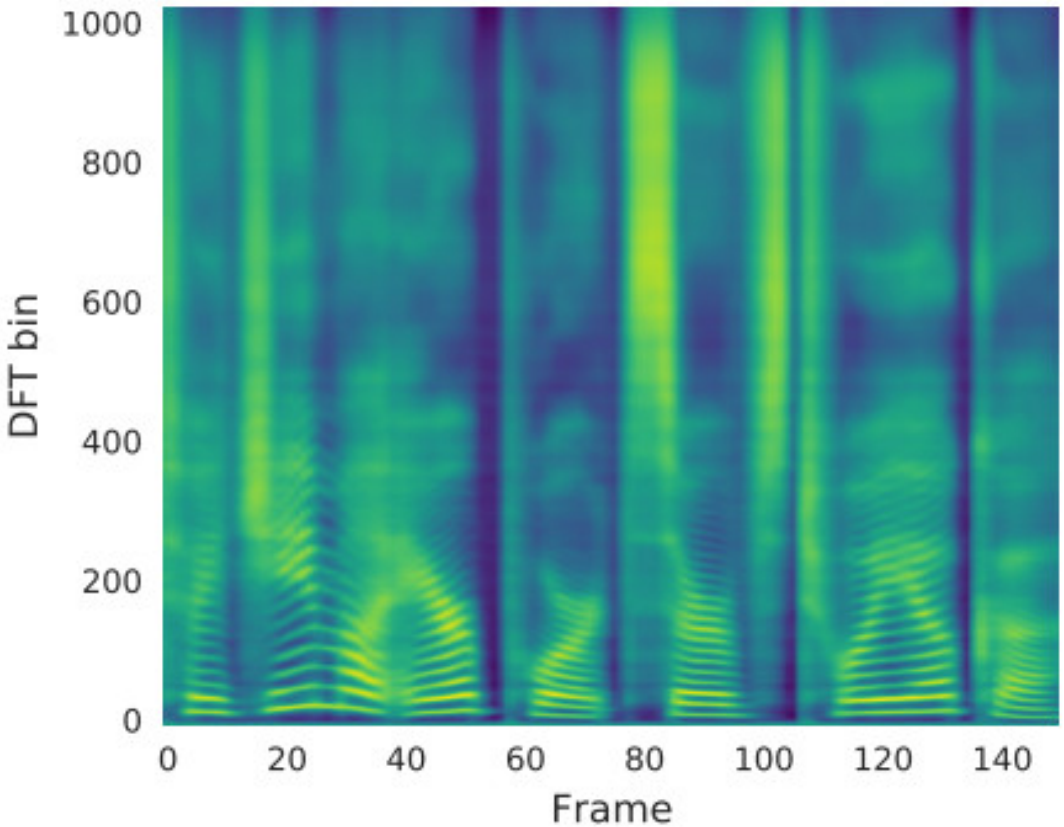
- This talk is a “Taste” - without time for hands-on

Something in the news…

- As well as introductory material, want to show “Hot Stuff”
- Major criteria :
  - Must be ‘fun’
  - Can’t use too much data (or require downloads)
  - Should be trainable in steps of < 5 minutes
- Recent interesting things :
  - WaveNet; DeepVoice; Tacotron (DeepMind, Baidu, Google)
  - pix2pix (community)
  - A:A’ :: B:B’ (Microsoft)
  - CNN for language translation (Facebook)
  - Objects from optical flow (Facebook)
- Final winner today is in the news for other reasons …

Appendix : tacotron

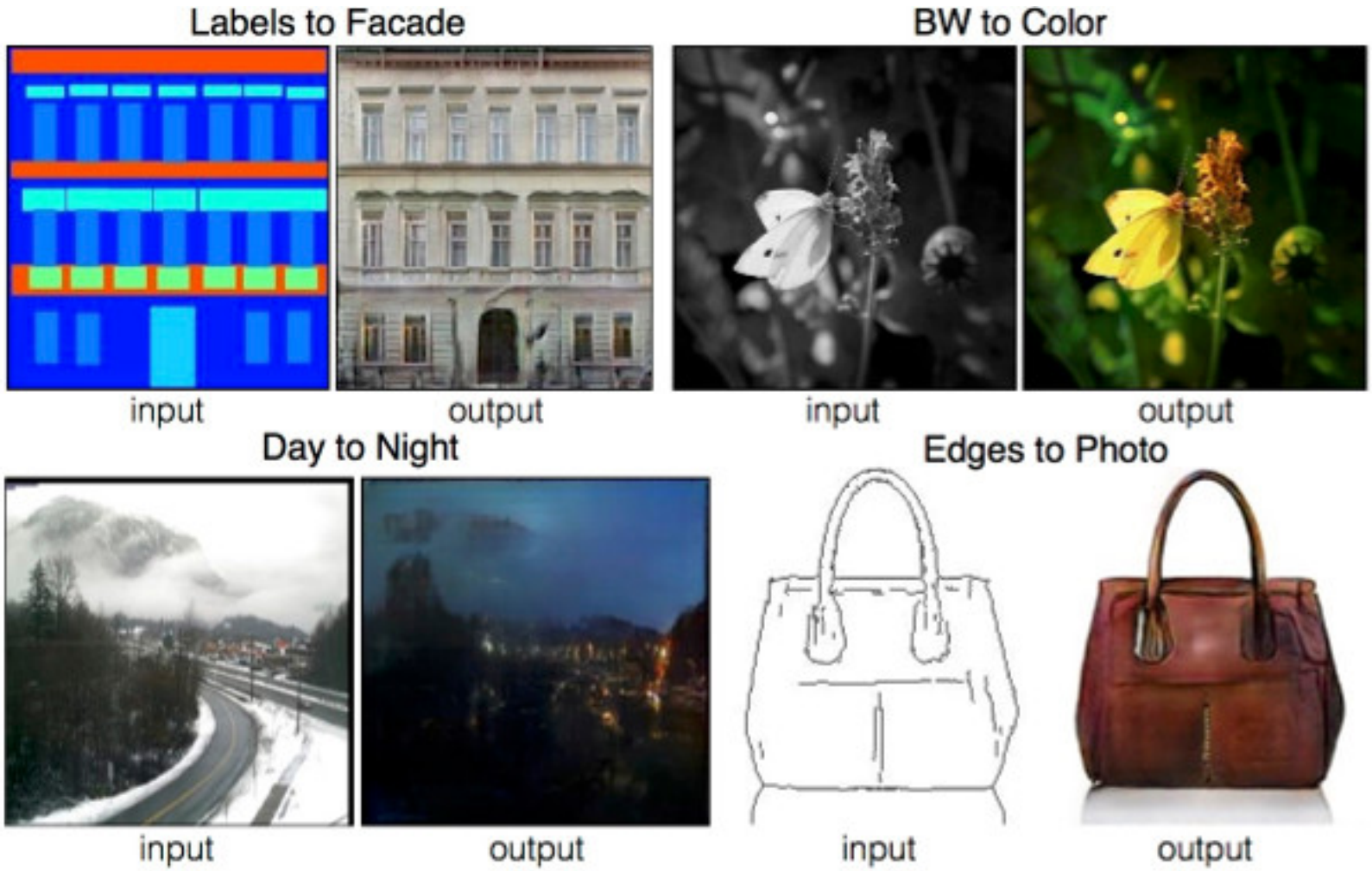
- Problem : no Korean native speakers in SG office



(b) With post-processing net

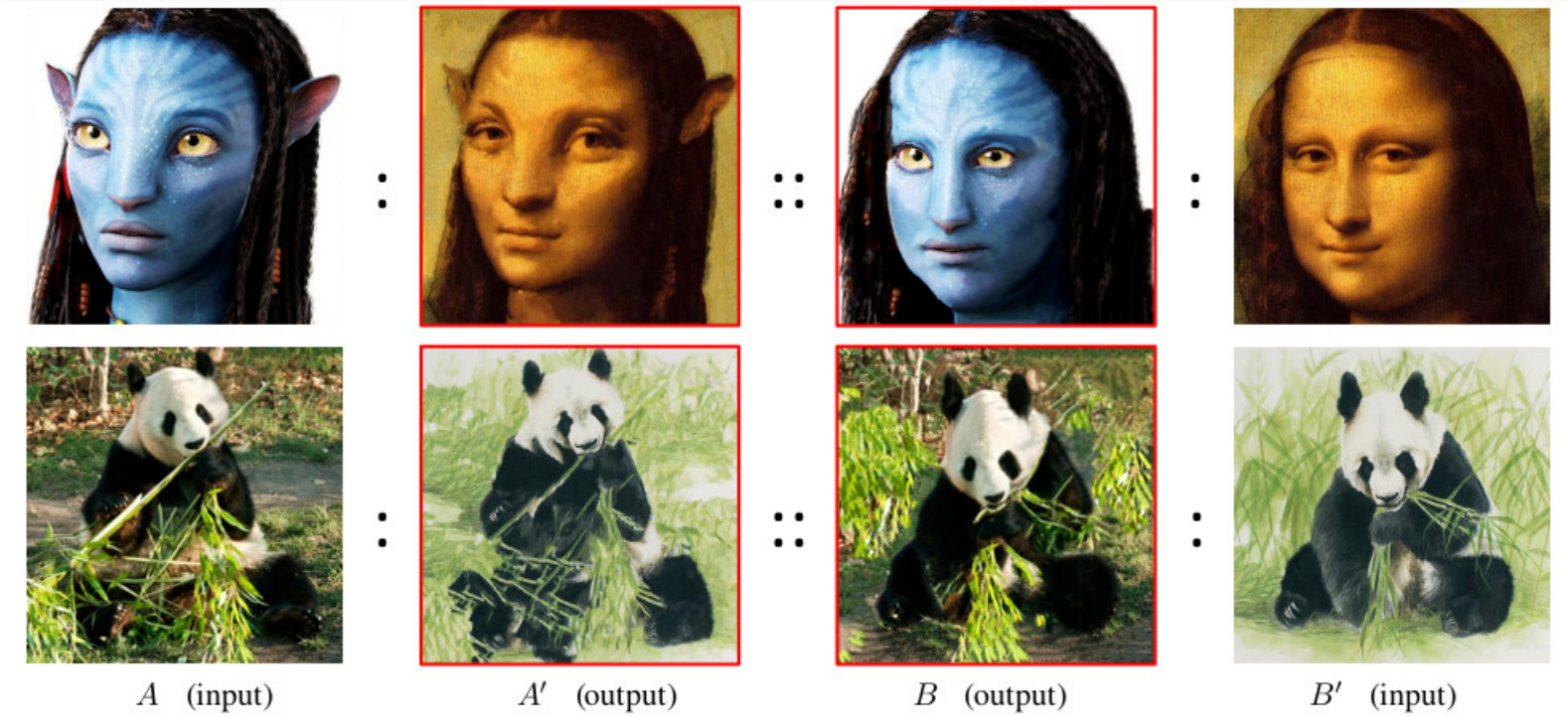
Appendix : pix2pix

- Problem : training time



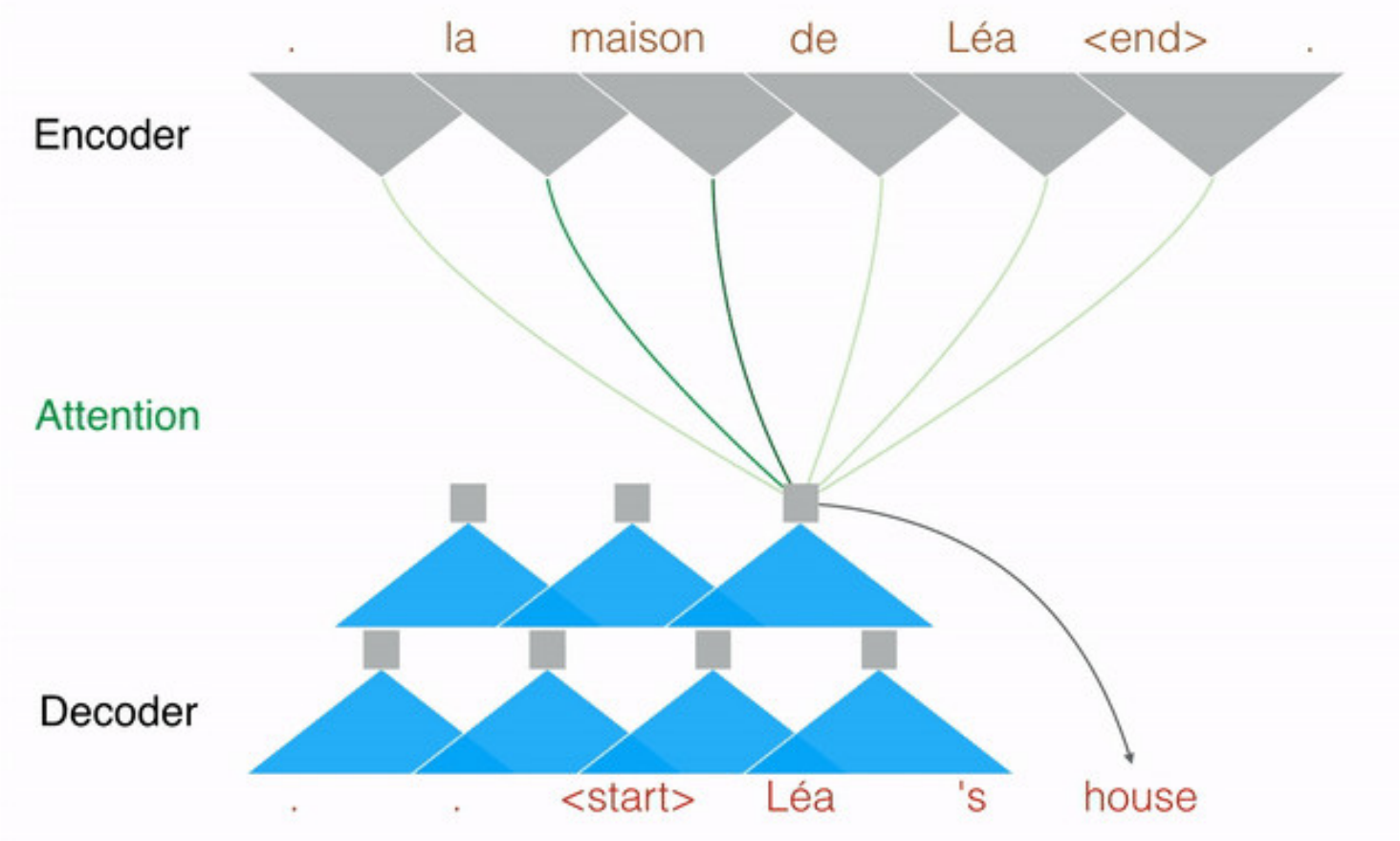
Appendix : Deep image analogies

- Problem : not really ‘Deep Learning’



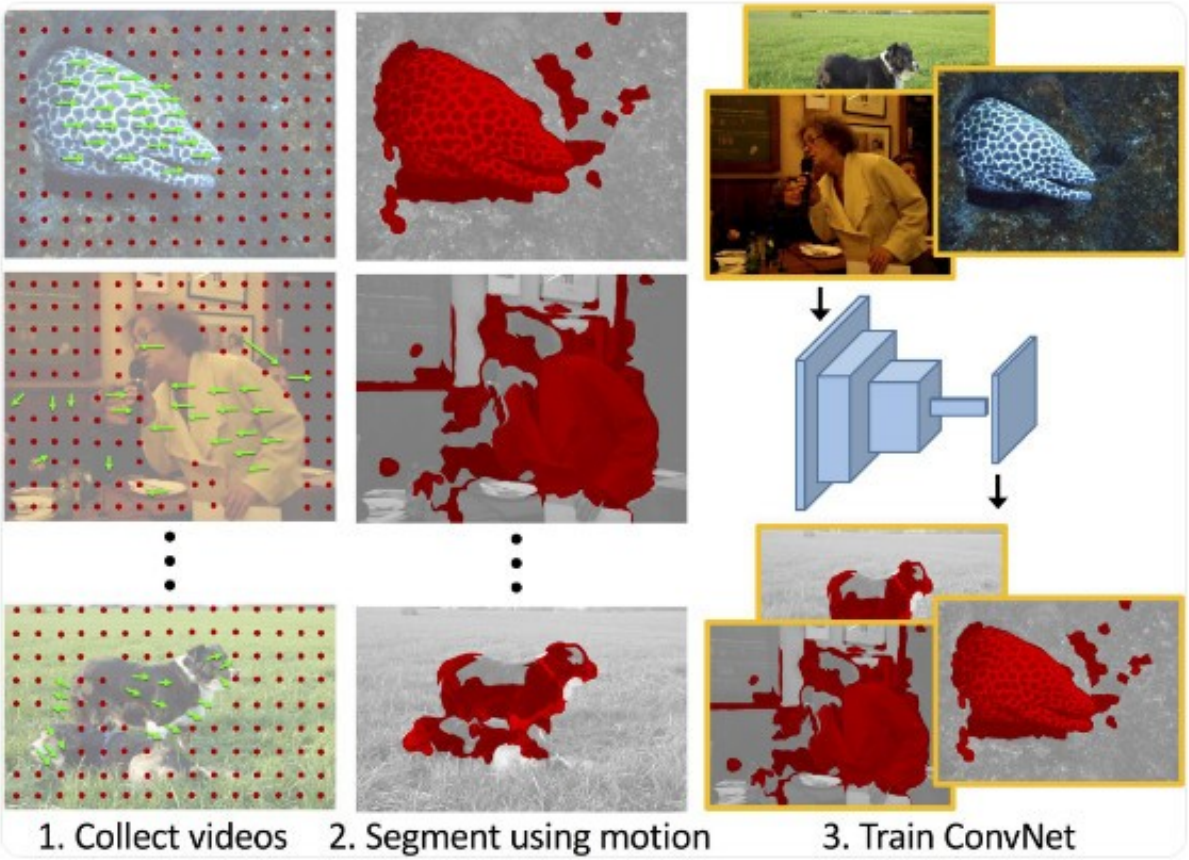
Appendix : CNN for translation

- Problem : Korean language
  - Statistically different from other languages
  - Seems to combine words and ‘extras’ : RAN OUT OF TIME



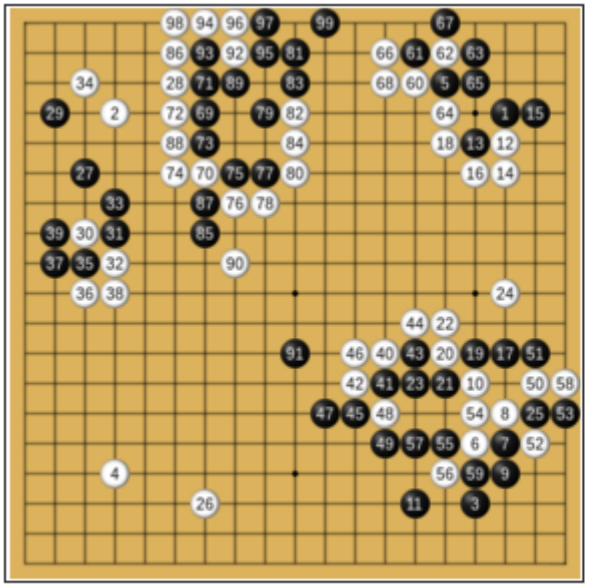
Appendix : Objects from optical flow

- Problem : Need to prepare some photos from videos
  - No time…



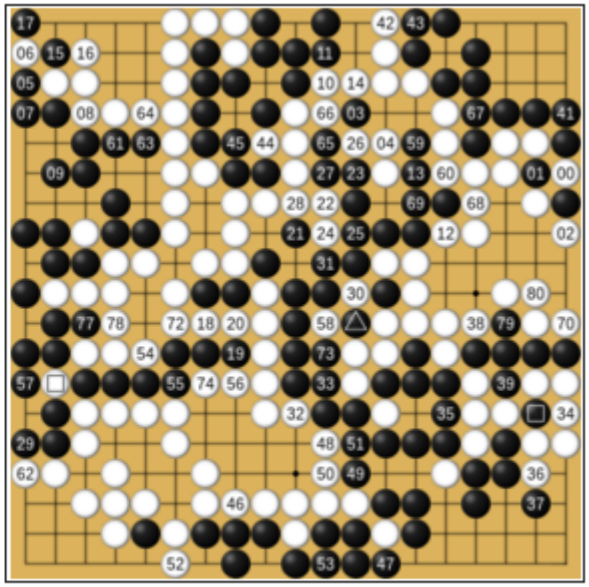
News (again) : AlphaGo

- Having achieved success in 2016…
  - Will soon be playing again against Chinese player Ke Jie
  - Has probably been self-playing continually since last year…
  - Also surfaced for a series of ~60 anonymous games (undefeated)



First 99 moves

Moves 100–199 (118 at 107, 161 at ▢)



Moves 200–280 (240 at 200, 271 at ▢, 275 at ▢, 276 at △)

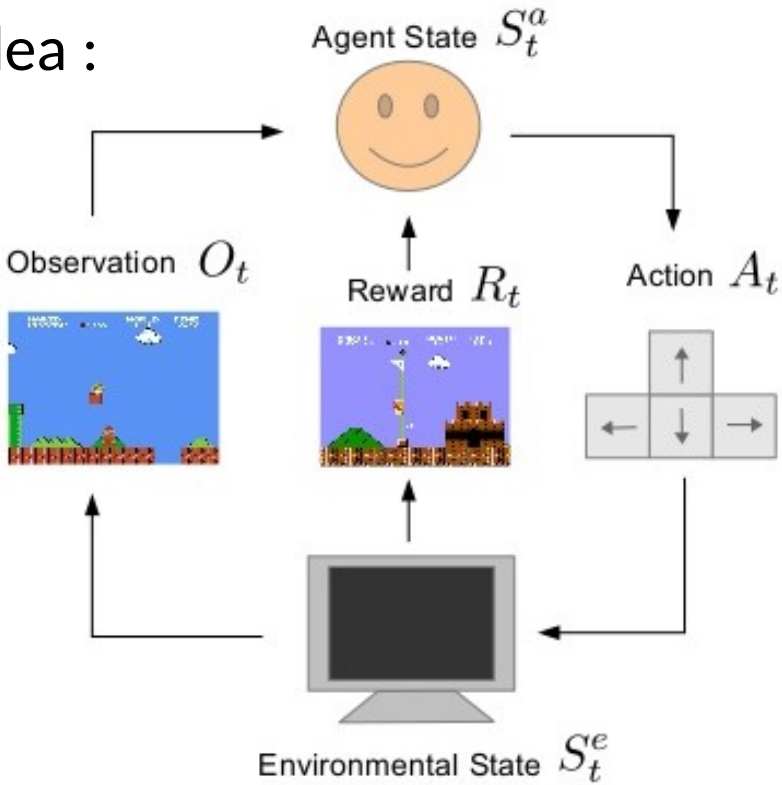
Reinforcement learning

- Techniques that focus on decision-making processes …
  - … where each decision/action affects the future options available
- Standard setting :
  - Playing Chess or Go (or games with hidden knowledge / randomness)
- Other application examples :
  - Deciding which advertisements to show
  - Dynamic pricing policies
  - Control of unknown ‘plant’ (e.g. air conditioning)
  - Robots “learning-by-example”

Reinforcement learning

- Learning to choose actions …
  - … which cause environment to change

- Agent idea :



- Rules of the game are unknown.
- No supervisor, only a reward signal.
- Feedback is delayed.
- Agent's actions affect the subsequent data it receives.

Q-Learning in one slide

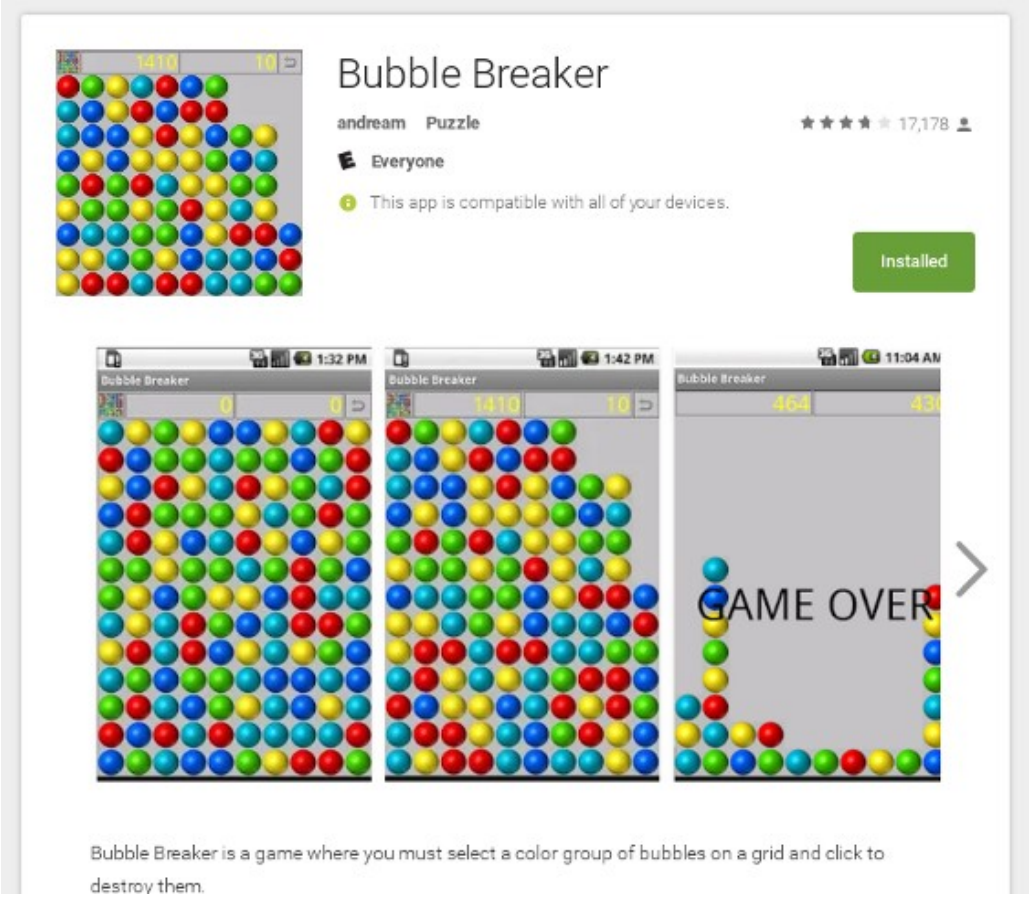
- Estimate value of entire future from current state
  - Let's call this function "Q(observable state)"
- Estimate value of next states, for all possible actions
  - i.e Q @ t+1 (states after each action A_i)
  - Remember to add on 'rewards' we earn for each one too
- Determine the 'best action' from estimates
  - By picking the A_i that gives us the best next Q @ t+1
- Do the best action A*
  - Check what state we actually get to, and rewards
- Now we can update Q(state) to the "better estimate" Q(A*)
  - But sometimes Q(state) is actually known (win / lose, for instance)

Q-Learning practicalities

- Concretely in Go (one-step lookahead) :
  - Q() value is ~ ‘winning probability of this board’
  - But (at the beginning) these are all complete guesses
- Check every possible move :
  - Work out which move gives highest Q() value next (‘looks best’)
- Execute the “best” move
  - Add the training data Q(previous) -> Q(next @ best)
- But sometimes, there is no next move :
  - The game is WON or LOST
  - These are ‘truth’ for Q()
- Training teaches all the Q() values on a relative basis

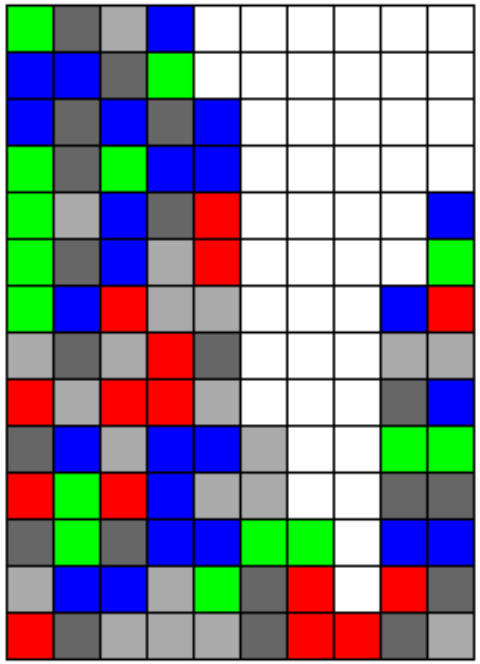
Workshop example

- Go is too difficult to train in 5 minutes …
- Basic principles can be seen in “Bubble Breaker”



Learning to play Bubble Breaker

- This is a very ‘clean’ version of the game
- Clicking on ‘joined’ bubbles kills the group
  - Bubbles fall down from the top to fill the space
  - Empty columns are filled by shifting columns over from the left
  - There are no special bubbles : just 5 colours
  - Game ends when there are no moves left



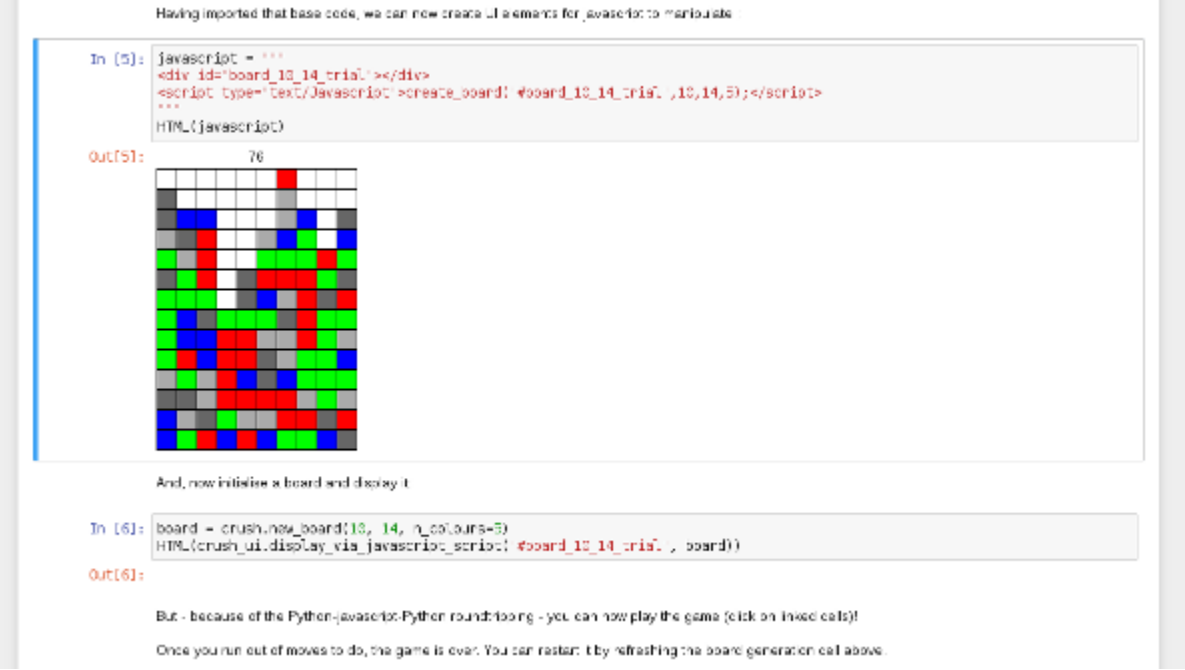
- Estimate the Q() values using a Neural Network
  - Inputs = current board features
  - Output = single number ‘Q()’

Bubble Breaker key points

- Turning board into features
  - 5 colours are symmetrical
  - Use that to speed up by 5!=120x
- Next actions are generated by Python code
  - Which also gives us next boards
  - EXCEPT : can't 'know' new columns before actually doing the move
- Exploit vs Explore
  - Simple 10% rule
- Rewards
  - Using the 'score' promotes short-term gains
  - Using new-columns-added leads to 'better' play

Workshop code / demo

- LIVE DEMO TIME !



Having imported that base code, we can now create all elements for javascript to meriously use

```
In [5]: javascript = '''
        <div id="board_10_14_trial"></div>
        <script type="text/javascript">create_board( #board_10_14_trial ,10,14,5);</script>
        '''
        HTML(javascript)
```

Out[5]:

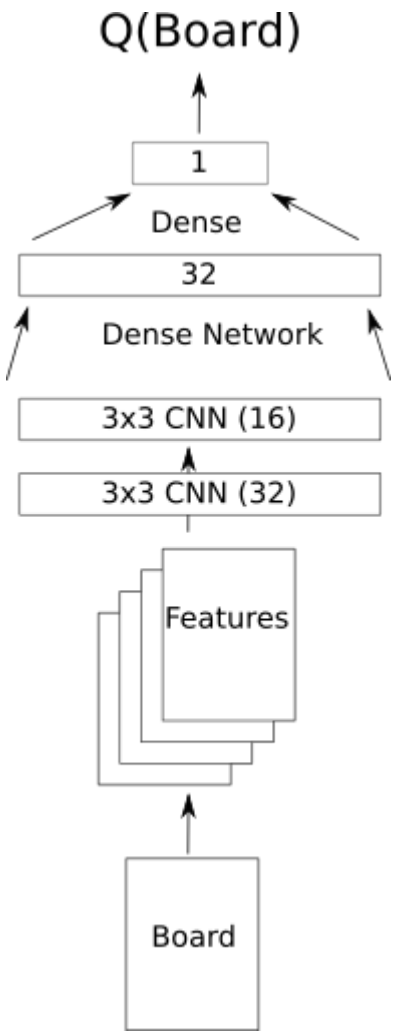
And, now initialise a board and display it

```
In [6]: board = crush.new_board(10, 14, n_colours=5)
        HTML(crush_ui.display_via_javascript_script( #board_10_14_trial , board))
```

Out[6]:

But - because of the Python-javascript-Python roundtripping - you can now play the game (click on inner cells)!

Once you run out of moves to do, the game is over. You can restart it by refreshing the board generation cell above.



AlphaGo extras

- Can get better estimates by looking several steps ahead
- But Go has too many possible next moves
  - This makes exploring the ‘tree’ of moves too difficult
- So AlphaGo also has a probable-next-move estimate
  - This prunes the game tree, so it can search more effectively
  - This estimator itself had a (low) Dan rank for single-step play
- Also, after tree has been traced:
  - Can teach Q() network at every level against every other one
- There was some analysis against human games
  - Vast majority of learning is now against previous versions of AlphaGo
- Actually used TPUs in early 2016

Deep Learning : training humans

- Data Science / Machine Learning / Deep Learning
- Difficult to hire people with right skills
  - As an employer, want to see practical experience
- Universities tend to lag
  - And team projects make for weak interviews
- MOOCs are good indicators of genuine interest
  - But coursework tends to be cookie-cutter
- Kaggle is cool. But now hyper-competitive (too much so)
- Starting in Singapore:
  - Deep Learning Developer Course ~ 8 weeks x 2 evenings
  - 50% teaching. 50% individual projects.

Wrap up

- Lots of exciting developments in DL
- Many can be simplified to their essence
- Best to learn hands-on :
  - Do projects from blog postings
  - Read papers; Make up your own projects
  - Contribute to open source
  - Do lightning talks; … Write papers

- All source code at :
  - URL : https://github.com/mdda
  - REPO : deep-learning-workshop (please *star*)
  - PATH : /notebooks/7-Reinforcement-Learning/3-BubbleBreaker.ipynb