Security Engineering and Machine Learning

Ross Anderson

Edinburgh and Cambridge

Outline of talk

- The prehistory of machine learning
  - Spam filters, fraud engines and SEO
  - From polymorphism to poisoning…
  - From redlining to sexism to snake oil…
- The neural network revolution
  - Should machine vision be robust or fragile?
  - Engineer for average case or worst case?
  - Do you need to sanitise input?
  - Apple's new abuse scanner
  - Do adversarial samples have any positive uses?
- Future directions

20th century

- Abuse was growing but detection was ‘conventional’
  - Anti-virus software searched for signature strings; so virus writers went polymorphic
  - RAID conferences from 1996 with NSA-sponsored intrusion detection competition: SNORT rules
  - Spam detection based on sender mail / IP addresses; so spammers learned to use compromised accounts / PCs
  - SEO just getting started…
- Jeff Kephardt and colleagues proposed neural network AV but couldn’t make it work yet (1996)

Early adoption – 2000s

- Paul Graham's 'Plan for Spam' made the case for Bayesian content filtering (2003)
- Countermeasures included poisoning
- Typical financial fraud engine by late 2000s: maybe 50 different signals extracted from transaction stream, then just try all the classifiers in Weka :-)
- Search engines avoided ML (Google until 2016, for reasons of control)
- People noticed insurers starting to redline again!

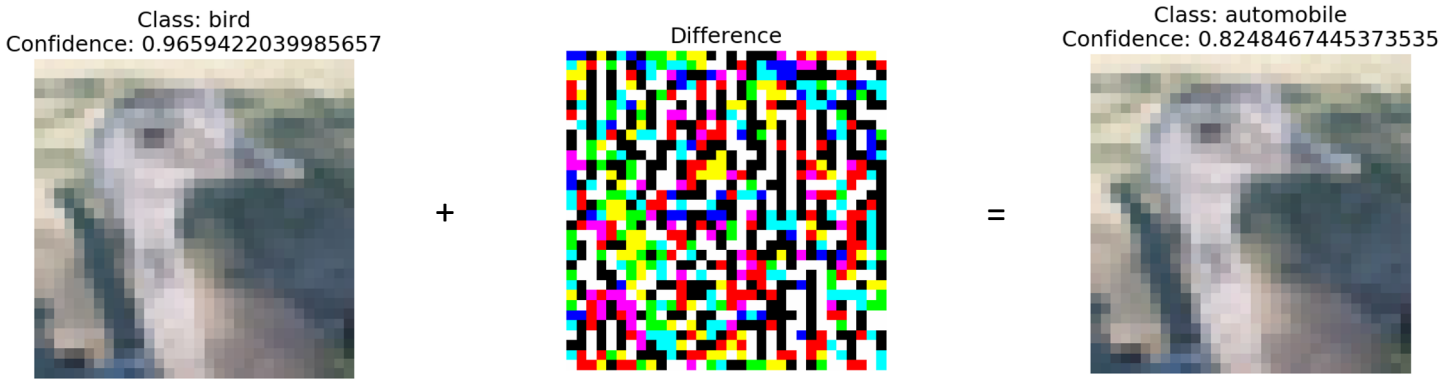
Security Engineering, v2, 2008

“If you build an intrusion detection system based on data mining techniques, you are at serious risk of discriminating. If you use neural network techniques, you'll have no way of explaining to a court what the rules underlying your decisions are, so defending yourself could be hard. Opaque rules can also contravene European data protection law, which entitles citizens to know the algorithms used to process their personal data.”

Automatic driver assistance systems

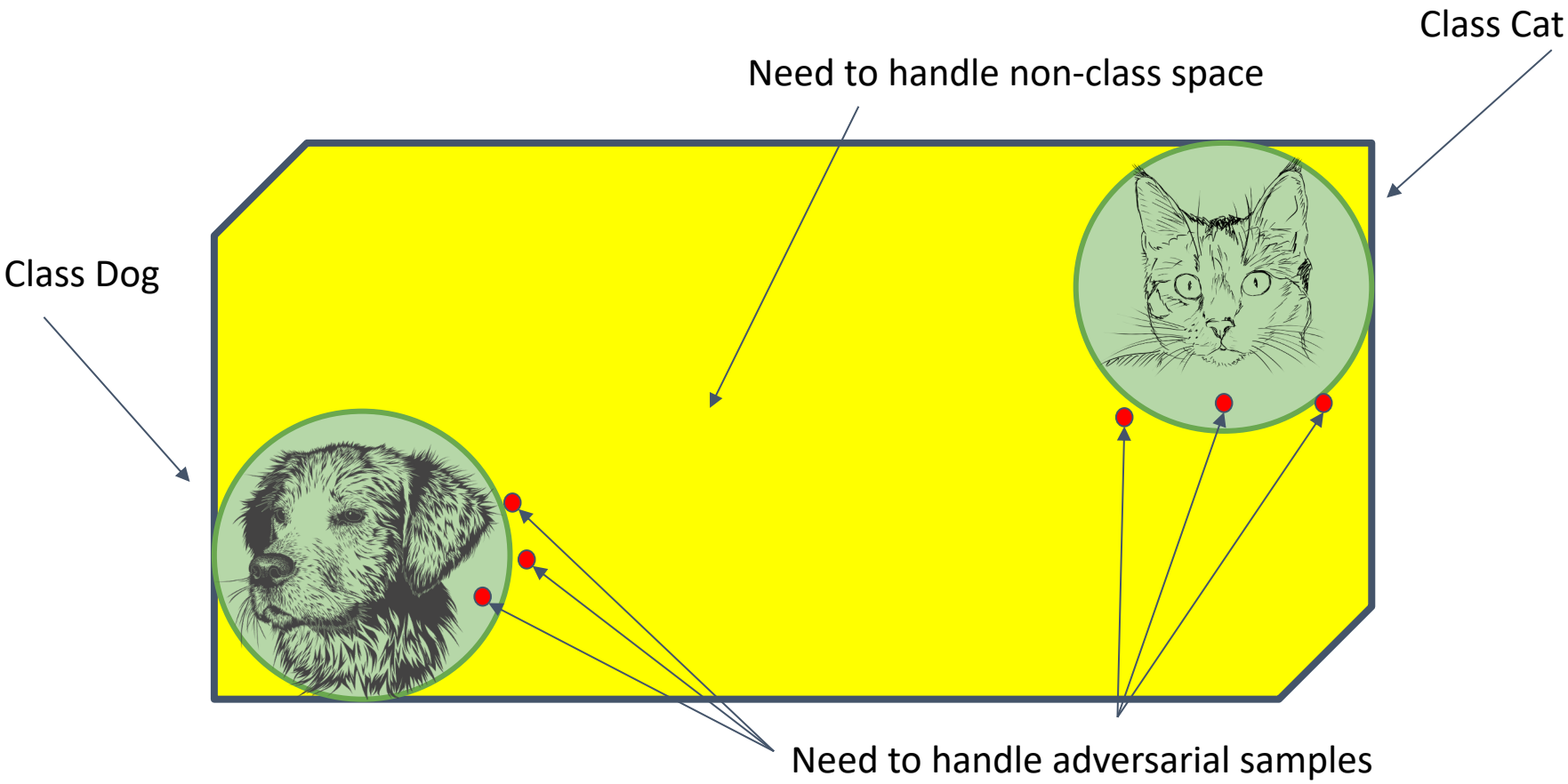
- My student Ilia Shumailov is funded by Bosch, which makes car parts, including machine vision
- Since 2012, DNNs work way better than anything else for image recognition
- But they are vulnerable to adversarial inputs, and making them robust damages performance
- So: if attacks are possible, but rare and not scalable, can’t we just detect them instead, and alarm?
- But how can you make the alarm hard to defeat?

Adversarial inputs

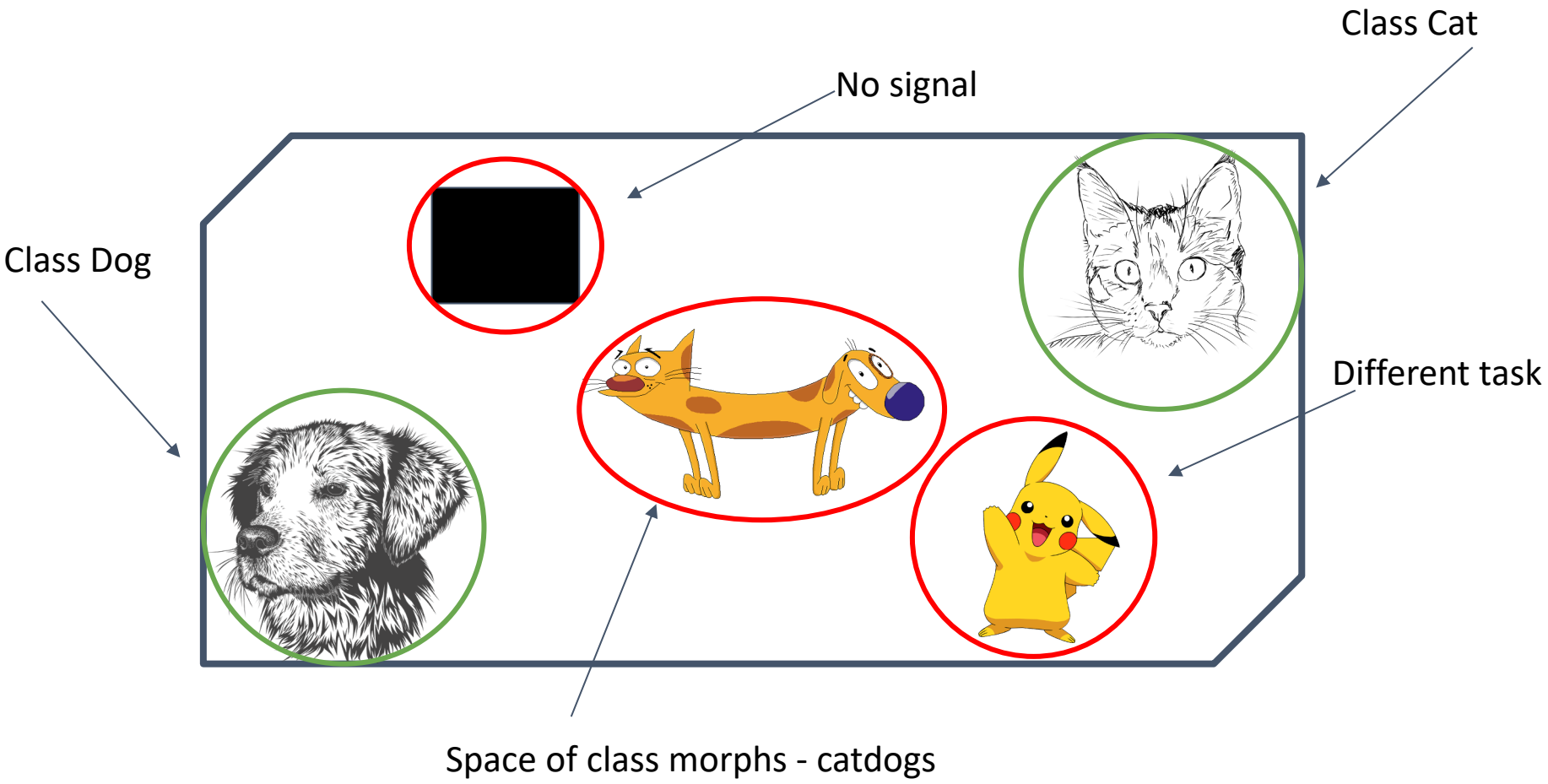


- From bird to car with a few tiny tweaks!
- Adversarial examples exist for all DNN models
- Attacks are findable and often transferable

Attack Detection



Attack Detection (2)



Idea: the Taboo Trap

- You train your kids to have beautiful manners
- Then they go off to school and within a week know some words your mother doesn't like!
- Breaking taboos => exposure to adversarial input!
- Can we set taboos (on outputs or activations) during training, and alarm when we see them?
- Answer: yes, this works rather well.
- Can diversify with different taboos – like crypto keys! (first interaction of crypto with ML…)

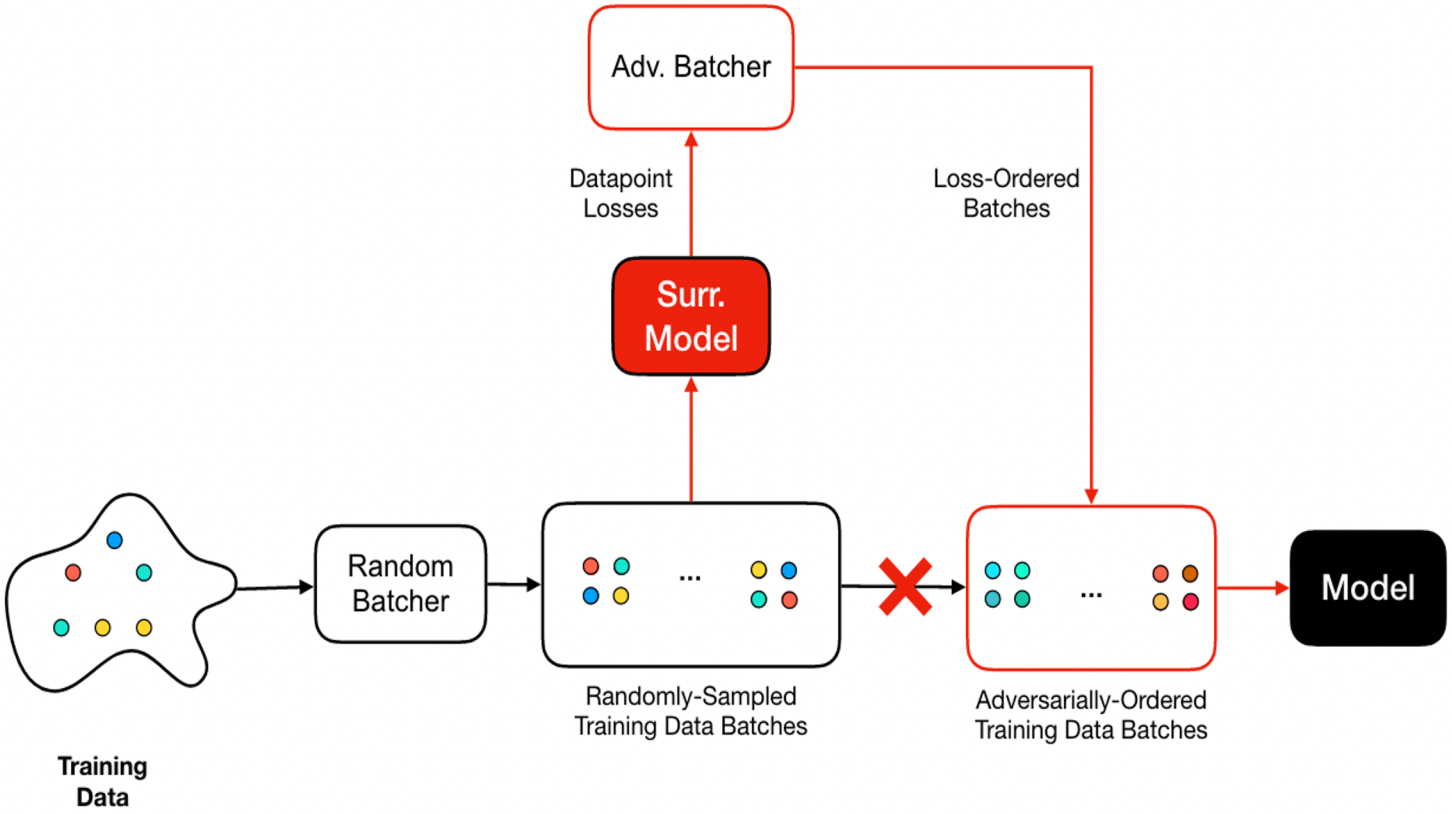
Developing the Taboo Trap …

- First pass: train to avoid some outputs
- Next: train to avoid some internal activations too
- Then: optimize for AUC, key space…
- Quite a lot of details! See '*Towards Certifiable Adversarial Sample Detection*', I Shumailov, YR Zhao, R Mullins, R Anderson, CCS AISec 2020, arXiv:2002.08740
- But embedding alarms at training time won't help if the adversary can get at the training pipeline…

Data-ordering attacks

- Most large DNNs are trained using stochastic gradient descent. ‘Stochastic’ is a fancy word for random, and many systems break when an adversary can get at the random number generator
- Does the same happen here?
- Large DNNs are trained on huge datasets, often under the control of different departments or even different firms, and often sent to data centres via content delivery networks…
- You need to be aware of these dependencies!

Data ordering attacks (2)



Manipulating SGD with Data Ordering Attacks, I Shumailov, Z Shumaylov, D Kazhdan, YR Zhao, N Papernot, MA Erdogdu, R Anderson, arXiv:2104.09667 (2021)

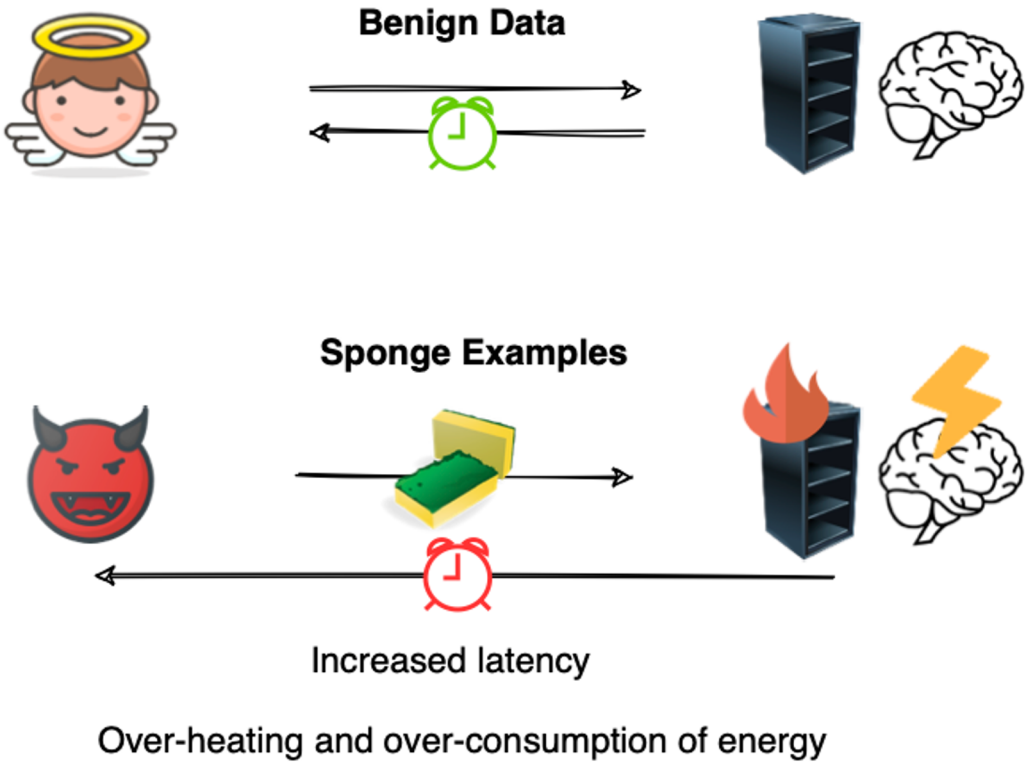
Data ordering attacks (3)

- This might have been expected!
- If you want your credit scoring system to be misogynist, but deniably so, start training it with ten rich men and ten poor women
- Then let the initialization bias do the work
- But it's much more general than that!
- It gives us the first attack on ML that uses clean data and clean labels
- So you need also alarms on bad output, and all the way along the pipeline if you can

Other alarms

- There are also attacks on ML confidentiality via model extraction (corporates worry about this more than about data subject privacy :-)
- Asokan and colleagues have developed neat ways to detect them (e.g. *'PRADA: Protecting against DNN Model Stealing Attacks', M Juuti, S Szyller, S Marchal, N Asokan, arXiv:1805.02628*)
- Perhaps query alarms will be the cheapest option…
- But what about availability? Classical computer security neglected this until the 1990s…

Sponge attacks



- *'Sponge Examples: Energy-Latency Attacks on Neural Networks',* I Shumailov, YR Zhao, D Bates, N Papernot, R Mullins, R Anderson, arXiv:2006.03463

Sponge attacks (2)

- We've discovered a very wide range of sponge attacks, on all the hardware /algo optimization
- NLP systems are particularly vulnerable! You can use double meanings (the 'conundrum attack'), or just drop a few Chinese characters in Russian text to stall a translation
- So ML systems must be designed for worst-case rather than average-case, or place limits on computation
- Real system engineers have known this stuff for decades, but today's ML enthusiasts ignore it!

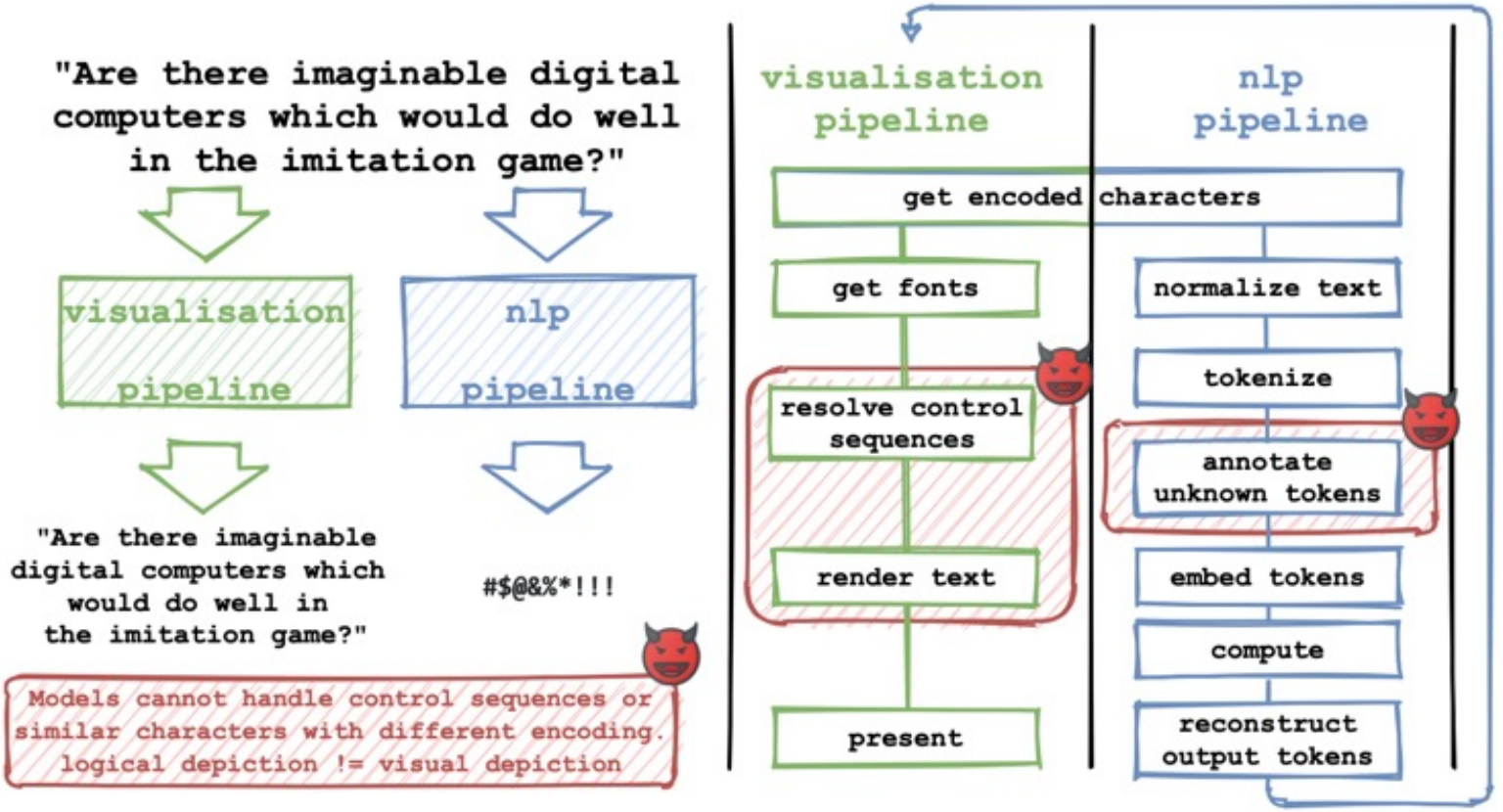
Bad characters

- Inspired by the discovery that Chinese characters hosed a Russian – English translation, my student Nicholas Boucher looked more carefully
- Unicode games were used in the early days of phishing to obscure URLs
- What sort of games can be played with machine translation systems?
- Plenty, it turns out!
- *Bad Characters: Imperceptible NLP Attacks, N Boucher, I Shumailov, R Anderson, N Papernot arXiv:2106.09898*

Homoglyphs

- Example: the normal ‘a’ and the Cyrillic ‘a’ render as the same glyph, but are different in Unicode
- You can often sabotage translation by swapping a handful of characters for homoglyphs
- You can often get a similar effect by dropping in a few zero-width spaces (yes, Unicode has them)
- This sabotages not just translation, but toxic content filtering
- Many potential abuse cases…

Mind the gap!


"Are there imaginable digital computers which would do well in the imitation game?"
visualisation pipeline
nlp pipeline
"Are there imaginable digital computers which would do well in the imitation game?"
#$@&%*!!!
Models cannot handle control sequences or similar characters with different encoding.
logical depiction != visual depiction
visualisation pipeline
nlp pipeline
get encoded characters
get fonts
normalize text
tokenize
resolve control sequences
annotate unknown tokens
render text
embed tokens
compute
present
reconstruct output tokens

Even more devious…

- Unicode also has directionality control characters, which let you swap text between left-to-right and right-to-left
- E.g. to embed an English phrase in an Arabic newspaper
- So: we can write an email in English saying “please pay $1000 to account 123”
- Google Translates it to Spanish as “to account 321”
- MS / G / IBM should know to sanitise all inputs… !

Human – ML interaction

- Ilia's adversary, in Berkeley, 2019

Human – ML interaction (2)

- As robots – systems incorporating ML components – mix and interact with humans, we can expect tension and conflict
- As well as the familiar human-human tussles, we already see robots trying to deceive and bully humans, and humans trying to outwit robots
- We'll see robots fighting each other, whether drone swarms on the Azeri-Armenian border, or Bruce Schneier's nightmare of AI bots hacking each other with our systems becoming collateral damage

Human – ML interaction (3)

- Animals want to know ‘Am I safe, or under attack? Must I be alert, or can I relax?’ Chronic stress and fatigue are toxic!
- Yet situational awareness has been largely ignored in security research
- Cryptographic and access-control mechanisms are designed to stop all adversaries all the time
- But this is too expensive for humans – we can’t all live in castles and drive around in armoured cars
- As we’ve seen, it also expensive for ML systems!

A role for manners?

- Animals have critical distances for flight and fight, which differ within and across species
- Humans have intimate, personal, social and public space
- These instincts are overlaid with social norms that regulate interaction with unknown people
- We build out systems of manners in all sorts of new interactions, such as driving
- This is now a showstopper for self-driving cars! E.g. how to interact when turning across traffic…

Pulling it all together

- Humans have evolved sensitivity to threat and risk
- As this is out of sync with the modern world, it is widely exploited by marketers and criminals
- But people are increasingly learning to be wary online and to ‘feel the hustle’
- Many ML systems will evolve a similar sensitivity, as alarms are often cheaper than engineered robustness
- Everything from cars to toxic content filters will need some form of situational awareness…

And then there's the snake oil

- See Arvind Narayanan's analysis of ML snake oil!
- ML has made:
  - Real progress for tasks of perception (e.g. faces, go)
  - Patchy progress for tasks of judgment (e.g. spam, toxic content filtering) – but needs constant human assistance with training, hard corner cases
  - No progress on tasks of social prediction (e.g. employee performance, future criminal behaviour)
- In short… dealing with people is hard!
- Most 'AI' startups don't use ML at all; most of the rest use ancient stuff like regressions

Apple's NeuralHash

- Apple's 'NeuralHash' algorithm will scan photos on your camera roll as you back them up to iCloud
- If it identifies thirty of them as suspect, it'll decrypt and report them to Apple
- Many system questions, e.g. why doesn't Apple scan iCloud already; Google, Facebook scan images
- Academics also can't touch sex-abuse material
- The algorithm was reversed though and it's easy to find collisions and pre-images

What will NeuralHash look for?

- Apple says they won't use any other images – so what about UK, Australia, Korea?
- If an Australian court orders them to put in images of Australian abuse victims, or missing children, or war-crime pics, will they withdraw all Apple staff?
- Are we moving from a world in which law enforcement had occasional wiretaps, subject to warrants, to one in which they have a listening post in everybody's private device?

And finally, some fun!

- Back in the 1990s we had fun messing about with steganography and copyright marking
- We developed a tool, stirmark, to see how robust hidden data was to distortions etc
- Now, tools like Photoshop offer machine-learning based ‘inpainting’ to fill in missing pieces of photos
- This makes it much easier to remove copyright marks, watermarks and reconstitute a usable image
- Are there any games we can play with this?

Inpainting

- Inpainting can combine local and global information in ways that previously required human intervention

Inpainter abuse?

- Inpainters make it even easier to hack together fake news pictures

Idea: markpainting

- Perturb the image so that any attempt to inpaint it will ‘patch in’ a known target image
- Uses the same basic technique as for adversarial sample generation: accumulate a suitable perturbation from scaled gradients
- Easy to target a known inpainter; with a bit more work you can make marks fairly transferable
- What cool tricks might we be able to play with this technique? (arXiv:2106.00660 & ICML, but was a fun project by David Khachaturov & Ilia Shumailov!)

Example: target = la Gioconda

- How a markpainted image gets inpainted, as a function of the perturbation budget

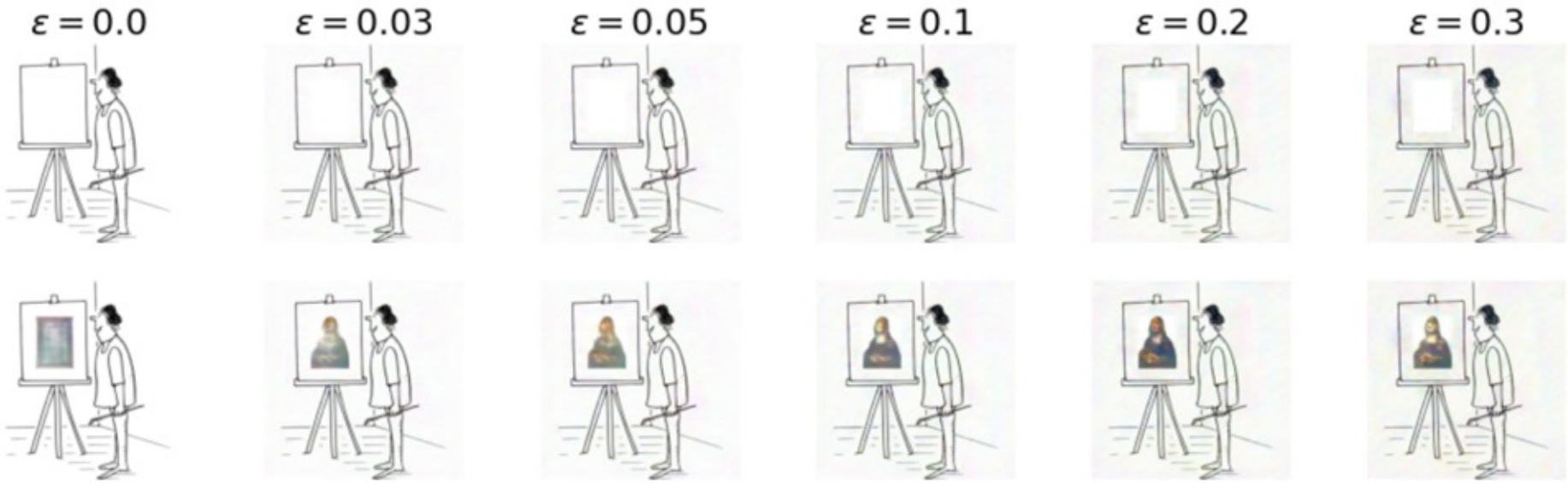
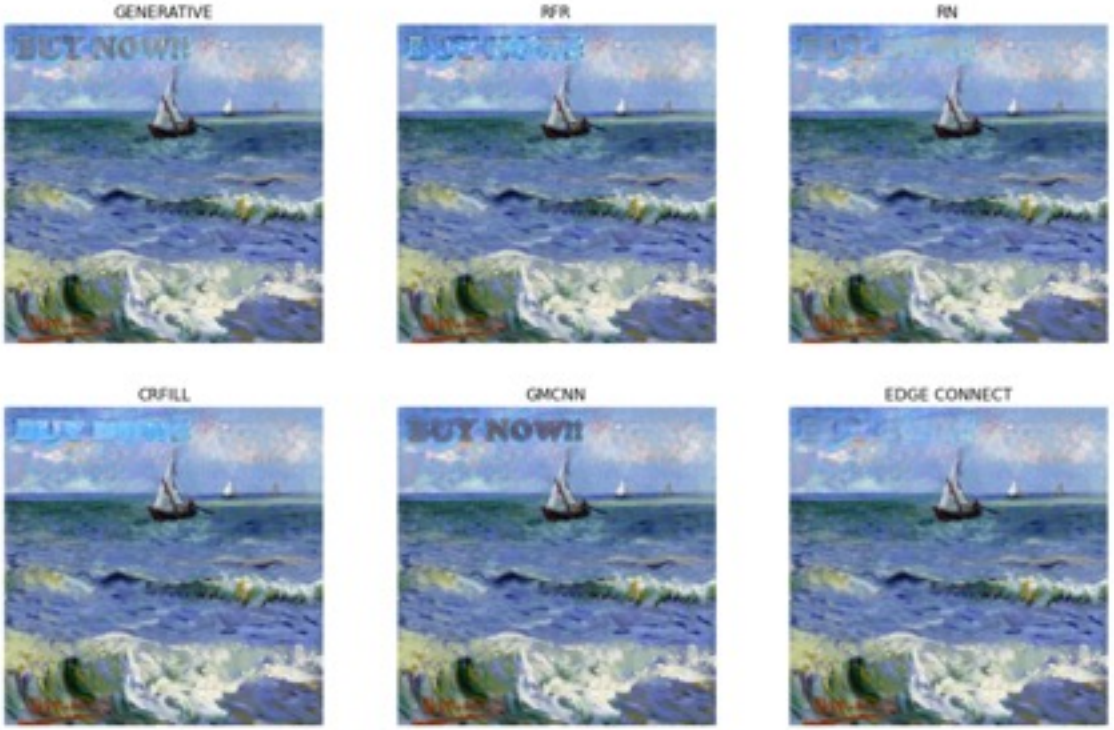


Figure 5. Inpainting with an increasing perturbation budget. Top row is the perturbed images generated using markpainting, and second row is the inpainted results of these perturbed images. We target the RN inpainter with 500 iterations and a step size of $\epsilon/100$. Note that this example is really hard, because we are filling a black and white image with color. Details are discussed in Section 5.2.

Making text hard to remove

- This might be a copyright mark, or just marketing



Some research questions

- How can we break machine-learning models?
- When we build systems that incorporate them, where are the new vulnerabilities?
- How can we develop situational awareness?
- How do we explore the new frontier, where ML models meet human complexity?
- Let’s see if we can get the basics right, such as robots that don’t bump into people!
- We also need a lot more honesty and openness!