Automating Fake Reviews

MIDS Summer 2018

Natural Language Processing Project

Kalvin Kao

Fake Online Reviews | Motivation

fake product / service reviews are a growing problem

"deceptive opinion spam"
"crowdturfing"

businesses pay workers to write reviews intended to deceive

active area of research:
using machine learning to both generate and detect fake reviews

Research Question:

How can automated reviews be improved to escape detection?

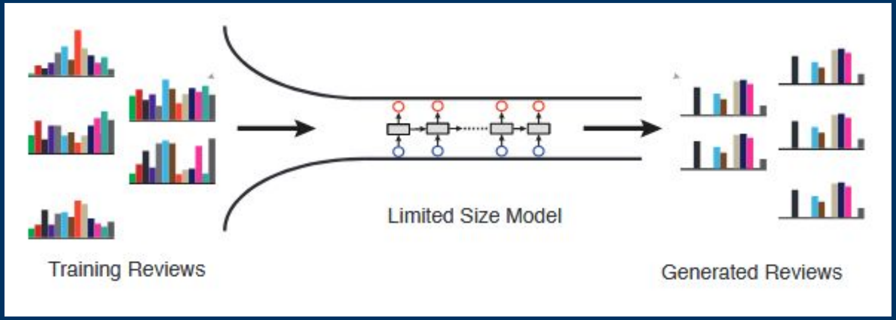
understanding the potential for attack is helpful for developing a defense

Previous Work | The RNN Language Model

"Automated Crowdturfing Attacks and Defenses in Online Review Systems", Zhao et. al.

'attack' model

1. pre-process review text
2. train character-level LSTM
   - 2 layers, 1024 hidden units (a4!)
   - 617k reviews (304M characters)
3. generate examples
4. post-process review text



'defense' model

exploits weakness in learned character distribution of RNNLM

LSTM 1

- learns character distribution of real reviews

LSTM 2

- learns character distribution of machine-generated reviews

to classify,

- feed example into each LSTM
- get prediction probability for each character
- form negative log-likelihood ratios between the predictions of the 2 LSTMs

Berkeley

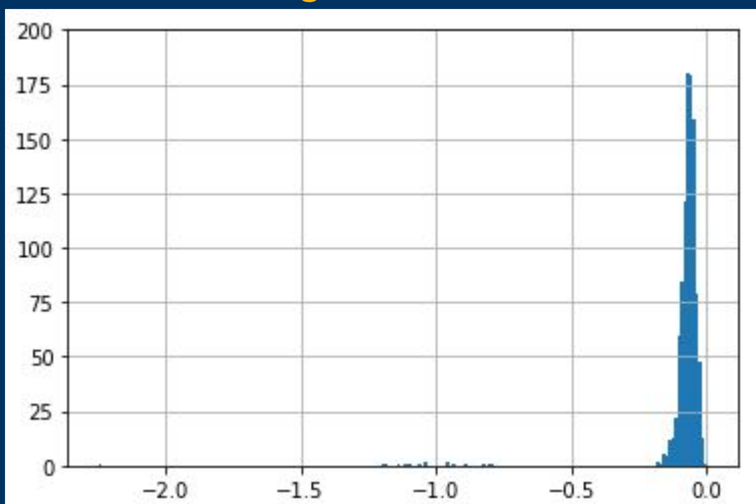
Baseline Model | Performance

77.2% test set accuracy

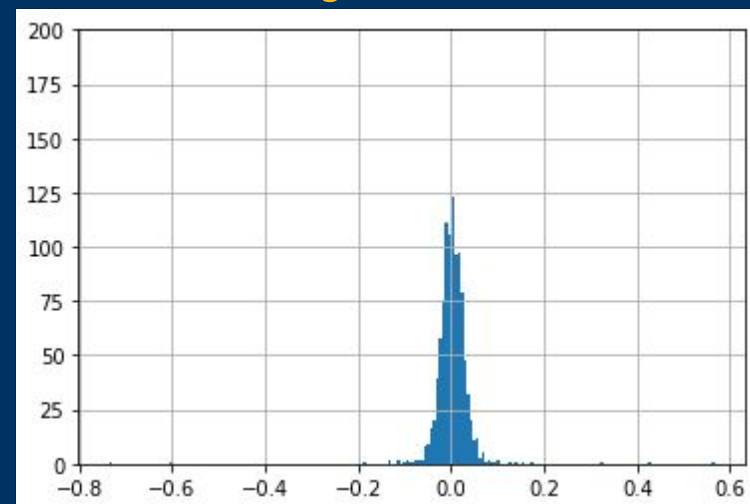
99.9% accuracy on real reviews
54.5% accuracy on generated reviews

(LR > 0 predicts fake review)

likelihood ratio distribution of human-generated reviews



likelihood ratio distribution of machine-generated reviews



- LSTM trained on real reviews has more certainty in predictions
- ‘attack’ model not trained well enough (replicated versions too small)

the one real review that was flagged as fake

"<SOR>what amazing service. i ordered 3 pizzas and 3 salads to be picked up the following day. i was instructed to make the order they the catering line. when i did this the salads were the catering size. i only wanted the personal size salads. when i explained this to gavin his response was no problem we can fix that. he was amazing. i will continue to recommend oreganos<EOR>"

- short sentences with repetitive structure and content

no notable difference between flagged & unflagged artificial reviews

- much is nonsense, but some coherent language at phrase-level

Some examples (from attack model):

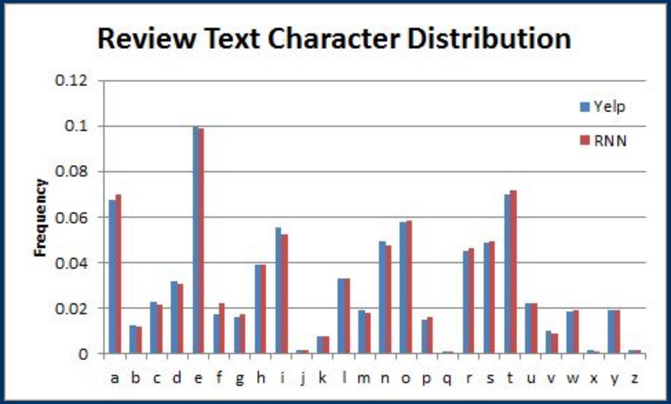
- <SOR>:)<EOR>
- <SOR>went in, and loved it!!!<EOR>
- <SOR>my waiter, and let be meeting on a waiting door office. (yes, the paleage home to shrimp across the back tacos rango", with efficients basically celeb
- <SOR>{pair of crispy but which took a little cooked with tempura sprinkles, and their fact that their world was so helpful and private and knowledgeable, i

Berkeley

Baseline Model | Context

marginal character distribution

single character frequency of generated reviews matches corpus, but context is what matters



review length

no evidence so far that sampling degrades over sequence length

56.1% flagged

52.3% flagged

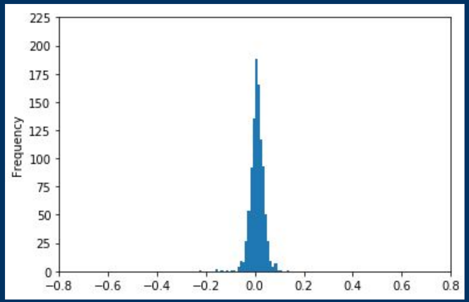
truncated back-propagation

75 time steps

- 48.4% flagged

300 time steps

- 66.4% flagged



- similar LR distribution

Berkeley

Baseline Model | Attack Model Character Embeddings

nearest neighbors

```
Nearest neighbors for '('
1.000 : '('
0.143 : '''
0.075 : 's'
0.069 : ')'
0.068 : '-'


Nearest neighbors for '.'
1.000 : '.'
0.048 : '7'
0.048 : 'h'
0.044 : '8'
0.041 : '^'
```

analogies

```
'(' is to ')' as '{' is to ___
0.675 : '{'
0.419 : ')'
0.172 : 'l'
0.157 : 'g'
0.145 : '_'


'{' is to '}' as '(' is to ___
0.627 : '}'
0.495 : '('
0.098 : '''
0.075 : '*'
0.065 : 'w'
```

- similar results for the LSTMs in the defense model
- none of the baseline LSTMs capture punctuation relationships well

How can the 'defense' model be beat?

Generated reviews have a distribution limited to what is observed during training

- How to add variability to learned character distribution?

As sampling proceeds, context drifts further from 'ground truth'

- "exposure bias"
- How to improve quality of text later in sampling sequence?

Original RQ: Do generative adversarial networks produce more realistic reviews than a pure recurrent neural network model?

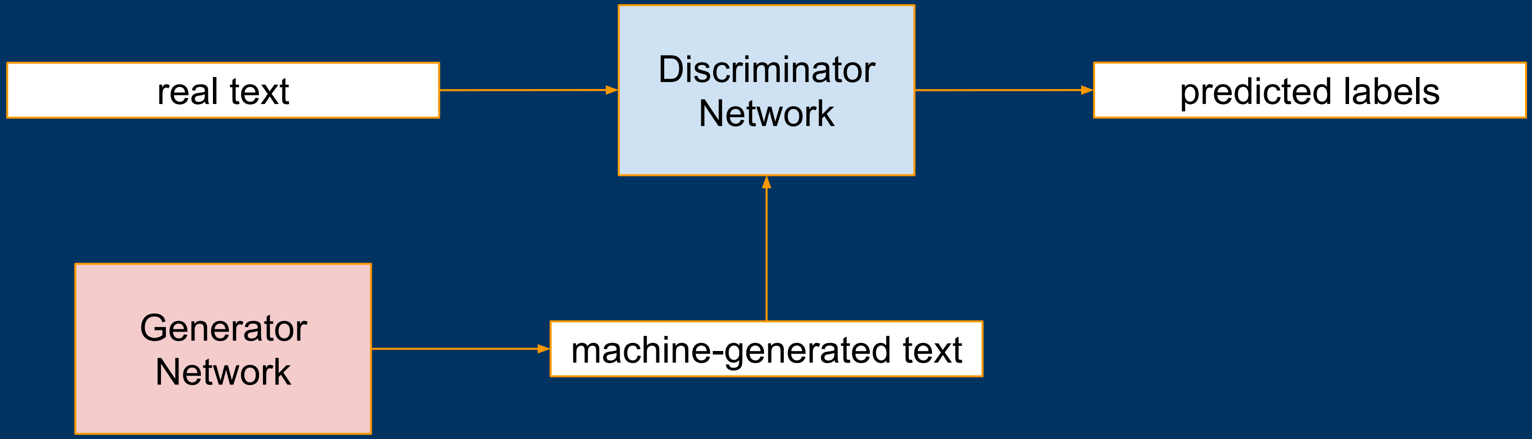
Berkeley

Generative Adversarial Networks | A game of cat-and-mouse

Consists of ‘Generator’ and ‘Discriminator’

Discriminator tries to catch examples produced by generator
- trained on both real and artificial examples

Generator tries to fool discriminator
- trained on opposing loss function for its generated examples

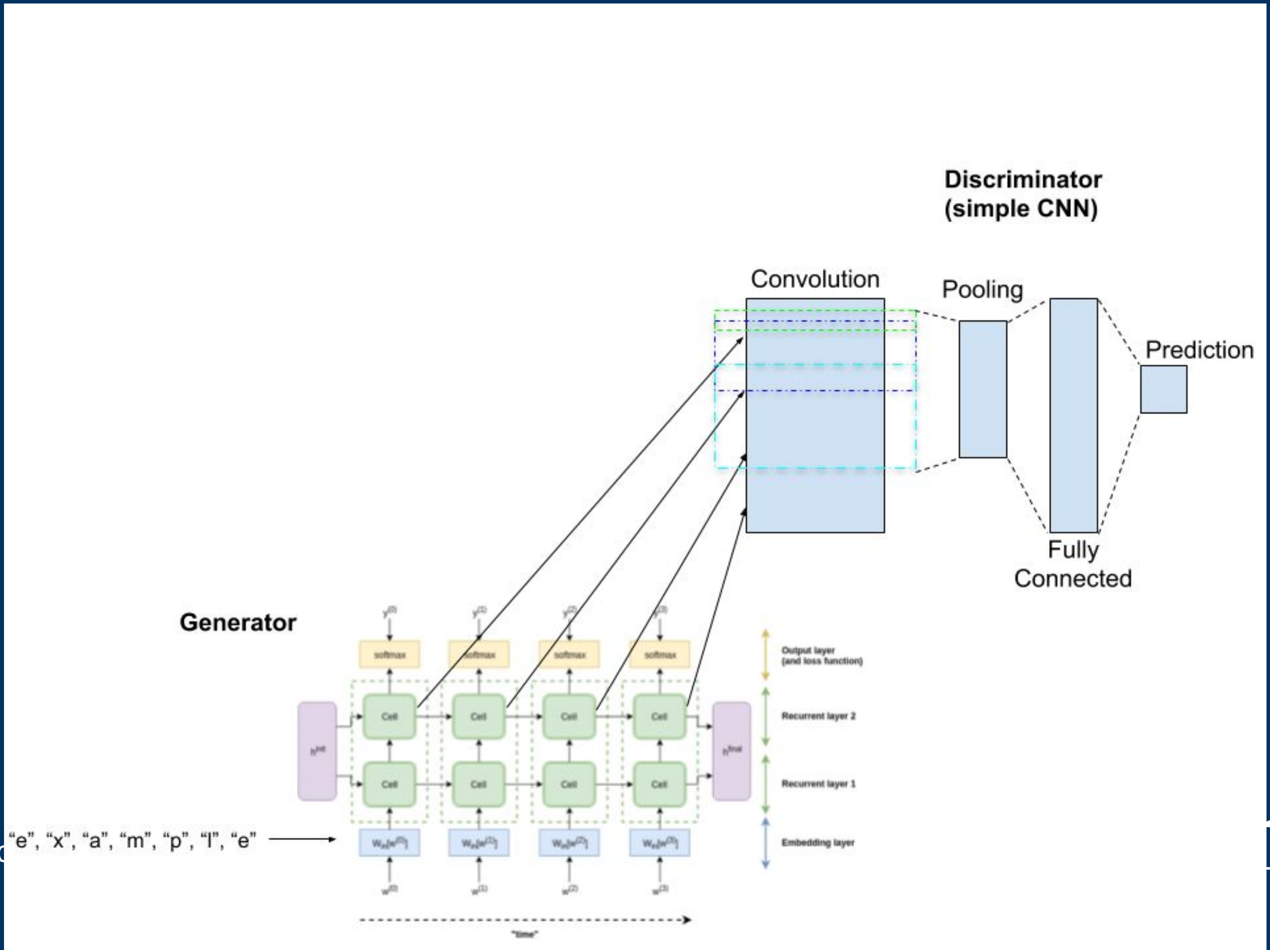


RNNs in GANs have a problem!
- RNNLM output text is discretized from softmax probabilities-- continuity lost!
- unclear how to back-propagate classification error from discriminator

LSTM Generator , CNN Discriminator

sampling discontinuity problem:

- standard technique is to use a “policy gradient” method
- non-standard approach:
    - feed RNN cell output from each timestep into discriminator input



Experimental GAN | Preliminary Results

before training...

1. <SOR> a a t a t t t t t t t a t he a t a t t t a t t a t a t a a t a t a t t t t t t t t t t t a t t a a t t t t t t t t a t t t a t a t t t t t a a t a t a t a t t t t a t a t t t a t t t t a a a t t t t a t t a t o a t a t t a t t t t a t a t a t t a t a t a t a t a t t a t a t t t t t t a t a t a t t a t t a a t a t t t t a t a t t t t

2. <SOR>favorite at all their lenntry pospize so we may maybe the pats. id try on the besi and the table had vegas came bottle of $7 dish and close. the pizza was holively and a first time to sell the sushi, ever!!! what my course is only left to acrisentix for hair to tom. so gave it so you do i thought i even has me down with s

...after training

1. <SOR>arealli thes noand a yngst ht 2phave we lod pl win care and rerer prme therlyey dio ouchainande of exy wod exi retller. thens like gored sppho sp ed itee cend. socesevitr extey plerse pinked. nove doremenc dit cher bei rand thet i topy fre! tarnd the oned re cecents nere onle. necisingevelen res

2. <SOR>quatp..,dd..dn..eggg.g,gg,.!.g,,go,.,,o,n..,,.,!,,.,.o.,,...,.,.,n.ro,,,,,go,ogogo.,.goo,,,,,rnor.,..!g.,g!go,.,gr.!gr.n,og,gr.gno.ogoogo.otot,.goo,rorogrorro.ono,!.gooooo,,oogo...ooo!.ogrrroooogoo,!googo..,rogegoorroogrr!ororgroogorororogroroooogoooooogooogoogornoogroogrooronoogogorronoooooonroorrorororo.o.!gogr,ogogon!.rrr

the problems

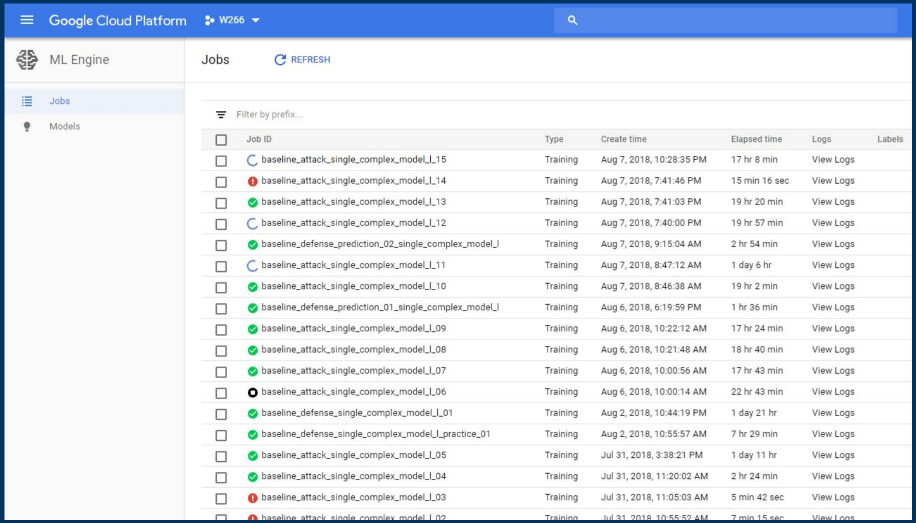
- discriminator often converged to a single class prediction
    - gradients are unclear or detrimental
    - should schedule updates and control learning rate
- after many rounds rounds, training is very detrimental

LSTMs in GANs

- standard methods reward conditional decision making
- experimental method rewards certain representations (or cell states)
  - CNNs are location invariant-- do not account for context
- use tf.nn.raw_nn instead of tf.nn.dynamic_rnn

Model Training

- Google ML Engine
  - could have made graph better suited for distributed training
  - helpful when I needed to run many experiments in parallel
  - not worthwhile for small experiments and studies
    - Compute Engine can use up to 96 vCPUs
  - library incompatibilities in training package can cause job restarts
  - distributed tensorflow has problems
  - mistakes can be expensive



Berkeley

Attack Model

- tokenize successive punctuation (i.e. emojis, “!!!”)
- tokenize pronouns
- vary the truncated back-propagation time

Defense Model

- weight likelihood ratios of later characters in the sequence

Experimental LSTM-CNN GAN

- fast pre-training when output layer is large
- minor post-processing

Thank you!

Questions?

Berkeley