Adversarial attacks on language classification models

Antoni Maciąg

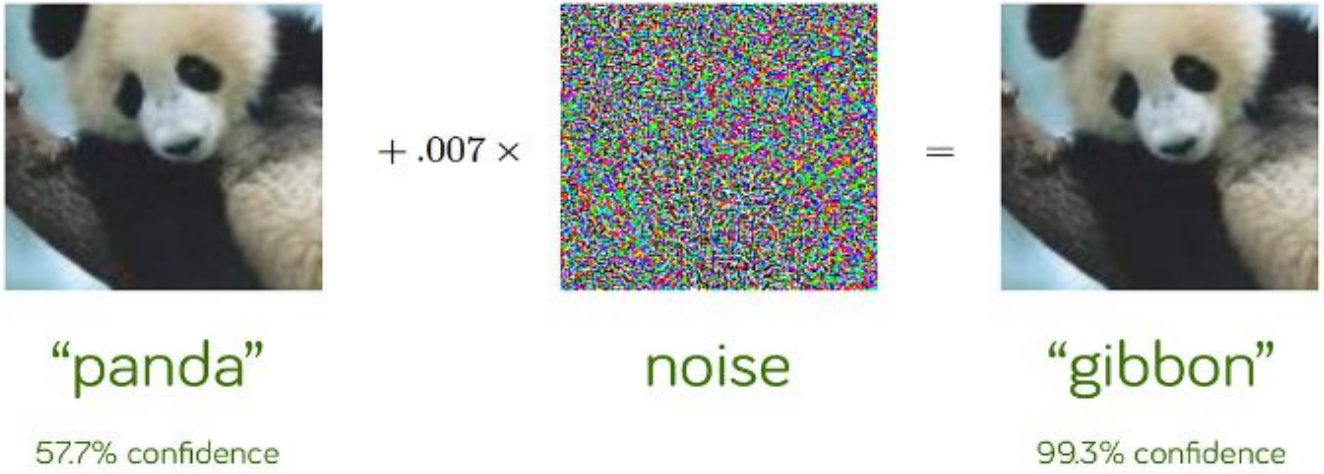
IAlab sharing, X 2023

Overview

- About adversarial attacks on text
- What I've been up to
- Why attacks are needed and interesting
- Non-generative attacks: review, shortcomings
- Generative attacks: review
- Planned experiment
- Your suggestions!

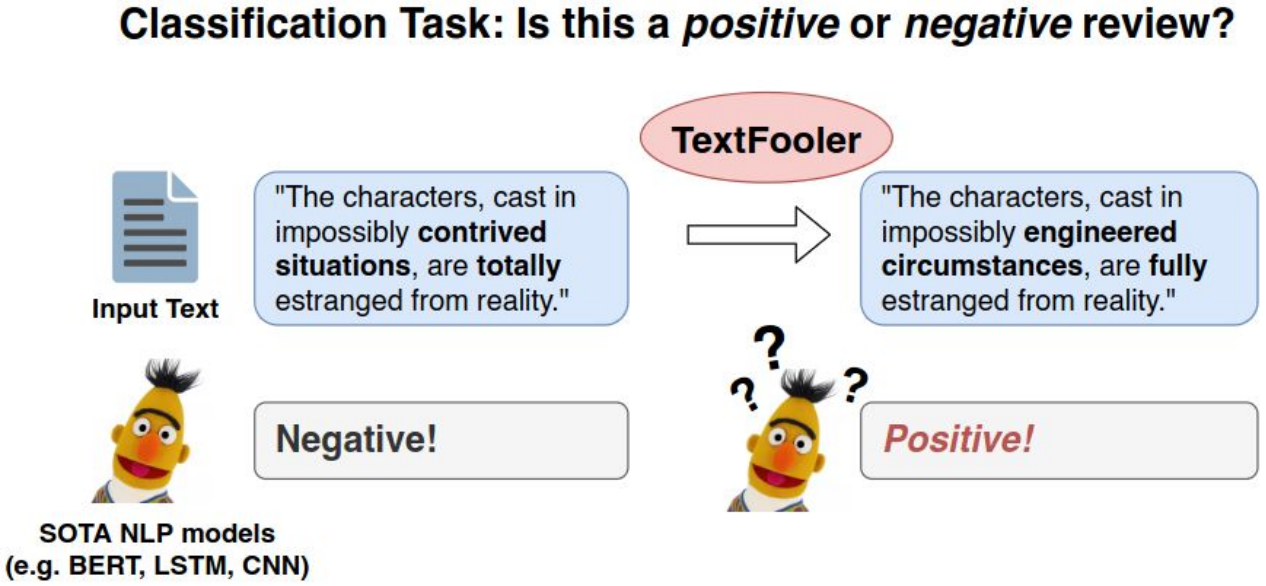
Adversarial attacks

- In general: artificially create inputs that make ML models output wrong predictions
- A classic example for image processing models that we've seen in a meeting before:
-



Adversarial attacks on NLP models

- Same idea: small perturbations, not noticeable for humans, resulting in wrong predictions by the model
-



What I've been up to and what for

- Trying to train and adversarially attack fake news detection models
- So many things to analyze!
  - Which attacks are successful and why
  - Which models are robust and why
  - What is relevant to their predictions
  - What changes if we include an external source of knowledge (graph, textual context), so that information can be *actually* verified, rather than just judging reliability based on style?
- Improve understanding, explainability, interpretability, trust in the models
- Measure and prove robustness

Practical justification

- TextFooler, a simple adversarial attack on BERT from 2020, completely *annihilated* the model
- https://arxiv.org/pdf/1907.11932.pdf
- Accuracy on fake news detection dropped from 96% to 16%
- Lower still on the other investigated tasks (topic classification, sentiment analysis, textual entailment)
- A CNN model literally brought down to 0%
- In the case of fake news, this can have tangible real-world consequences
- Conclusion: testing robustness to such attacks *needs* to be part of the evaluation of a fake news detector, or of any model that it might be in someone's interest to attack

So what does TextFooler do?

- Algorithm:
  - Find the words most relevant to the model's prediction (check logit change after deletion)
  - Choose one of them to replace with its n synonyms (words with similar embedding), choose the replacement that minimizes the model's confidence in the correct label
  - Repeat until prediction changes or the resulting sentence is no longer similar enough to the original
- Requires access to the model's logits, *but* the results are partly (>50%) transferable between the models they tested
- More abstractly: given a piece of text and a model, construct a piece of text whose perception by humans is very close to that of the original, but its "perception" by the model (i.e. logits) is different

What do other models do? Review

- Overwhelmingly, they remain within the framework of synonym substitution
- Main pain: discrete space
  - Discrete and huge, expensive to search
  - Methods from CV not applicable (images are more continuous)
  - Permutations always at least noticeable upon inspection
  - No space to freely move around in (instead, jumping between words)
- Nonetheless, various ways of performing synonym substitution have been developed
- One paper identifies words to replace based on the gradients of the target model
- One uses genetic algorithms
- And many others, which won't be discussed here

Limitations of synonym substitution

- Most obviously: there's a realm of possible but unexplored changes, potentially more effective
- As mentioned, high computational cost
- Little control over loss of naturality
- As exemplified by TextFooler, it often works nonetheless
    - However, since then we got models like **Ro**BERTa which are supposed to be more **Ro**bust and harder to attack
- What more general framework can we work out?

Text generation for attacks

- Same high-level goals (similarity, fooling), but now we'll explicitly express them as a loss function and let a model worry how to optimize it 🤭
- Instead of modifying the original text, the model will eat it and generate a new piece of text from scratch (seq2seq framework)
- Advantages we hope for:
  - Solving the problems of low flexibility and high computational cost (e.g. single model pass to generate a token, constant time)
  - Moving closer to continuous latent spaces
    - Trying to slightly perturb a vector in the sentence encoding space to get to the other side of the model's decision boundary
    - Intuitively, this space is more densely covered with sensible sentences than word space (i.e. "less discrete")

Text generation for attacks - initial idea

- Input: Model M for binary classification, outputting confidences for the labels *true* and *fake*, summing up to 1
- Similarity function S: (text, text) -> [0, 1], evaluating similarity in a way that aligns with human perception
- Given text T, learn to generate an adversarial example A:
  - Define F(M, T, A) as *fooling factor* - model M's confidence that A is true, if it judges T to be fake (or the other way around)
  - Maximize the following loss function L:
    - $L(M, T, A) = \sigma S(T, A) + \phi F(M, T, A)$
  - Intuition: move towards the model's decision boundary, not straying far from the start. The Greek determines what's more important
- Has someone done that yet?
  - Who are we kidding... of course

Text generation for attacks - previous work

- Exists! But it's surprisingly hard to find
- A paper from a Ms. Wong, Stanford, 2017
- Generation still unpopular, hardly mentioned in surveys from >=2020
- Another one from Le et al. generates comments that change fake news detector's predictions for an article (concatenation attack)
- They and a few others are discussed in subsequent slides

DANCin SEQ2SEQ (Wong, 2017)

- Very similar to my original idea
  - She even came up with the same loss function! Nice
- Similarity function as cosine similarity between sentence embeddings
- Showed need to control grammaticality as well as the other two objectives mentioned. Similarity embeddings don't capture that (probably not trained on ungrammatical data)
- Attacked a Naive Bayes classifier, unsurprisingly the model turned out to replace words
- Trained like a GAN (next slide), with the discriminator discriminating the labels instead of generated vs. sampled
- Unstable training, attributed to GANs

Adversarial vs. adversarial (digression)

- Two meanings of the word:
  - Adversarial training as in Generative Adversarial Networks (training a generator-discriminator pair), intended to create two models
  - Adversarial attacks, intended to fool ML models
- These concepts are somewhat related but it seems to be a coincidence that they’re described with the same word
- Let’s try not to get confused
- Like I did, spending a day reading GAN papers before this presentation for no good reason

MALCOM (Le et al.)

- Malicious Comments (concatenated to an article)
- Three objective functions (similarity, fooling, grammaticality), maybe taking cue from Wong
- Trains them alternatingly until convergence
  - Which I'm not sure is guaranteed
- Reinforcement learning, also GAN-style training

Text generation for attacks - honorable mentions

- Zhang et al. pretend not to do synonym substitution, by perturbing in latent space and rounding up to nearest word embedding
- Li et al. have an encoder and decoder and perturb the encoding, but they mention that such perturbations result in ungrammatical decoded sentences, so instead they map to another latent space and perturb there, then map back and decode
  - I don't understand why that would help
- Yoo et al. learn to decode a sentence with a given parse tree, so they perturb the embedding and decode with the original tree, ensuring grammaticality
  - But not naturality
  - Kinda overkill, too

Planned framework

- Train to fool a *single* detection model. The fooled model is a part of the input data during training, not during the attack
  - Some attacks are model-agnostic. Some mention transferability of the adversarial examples. I won't be investigating that for now
- Access to the model's logits, nothing else
  - Which literature still counts as black-box, maybe not realistically
- Implementation note: process sentences separately instead of the whole text. Cheaper, but some subtle perturbations won't be allowed

New plan

- Enlightened by the literature, let's update the loss function from earlier
- Define N(A) as naturality of the adversarial example
  - $L(M, T, A) = \sigma\, S(T, A) + \phi\, F(M, T, A) + \nu\, N(A)$
- Now we can more or less train that with RL
- How to implement N(A)?
  - Pretrain a discriminator with a GAN, but then keep it plugged in while training (other than papers only using pretraining, or alternating)
- Keep embedding cosine similarity as S
- Implementation note: Wong noticed a high score in one component offsets the loss in the others. Maybe instead use something like weighted harmonic mean?

Further considerations - fooling direction

- The model is expected to learn the behavior of the fooling function
- However, the way we formulated it, before it determines which way to try to move the logits, it needs to know the victim model's prediction
  - So it needs to be able to "simulate" it, consuming its capacity
- We can help it out by giving it a target label
- This rings the bell of style transfer generation
- If the victim is essentially style-based, we could apply existing techniques
  - It shouldn't be
    - It often is
- Then the task becomes: given a fake news article, generate an article with the same semantics, "in true news style" (or vice versa)

Further considerations - changing the actual label

- We can't automatically control the human-assigned labels stay the same
- We might learn to actually make true news fake (and vice versa?)
- But:
  - For style-based models, that probably won't happen
  - For actual verification models, it will be interesting if it happens
  - So no problem, I guess
- We hope to be defended by S
  - But does negating a sentence negate the embedding, or just slightly change it? (I don't have a quote for that)
- In some similar cases, with labeled relevant spans (claim detection, question answering), we could forbid the model from modifying them

Next

- Implement a model as described above
- Come back to you with results

謝謝大家聆聽！

antoni / 安德

Sources

https://arxiv.org/pdf/1901.09657.pdf

https://arxiv.org/pdf/1907.11932.pdf

https://dl.acm.org/doi/pdf/10.1145/3593042

https://aclanthology.org/N18-1170.pdf

https://www.sciencedirect.com/science/article/pii/S0031320321002855?ref=cra_js_challenge&fr=RR-1

https://arxiv.org/pdf/1912.10375.pdf

https://www.sciencedirect.com/science/article/pii/S0957417422021881

https://arxiv.org/pdf/2009.01048.pdf

https://arxiv.org/pdf/2005.05909.pdf

https://dl.acm.org/doi/pdf/10.1145/3374217

https://arxiv.org/pdf/1804.08598.pdf

https://ojs.aaai.org/index.php/AAAI/article/view/11957

https://arxiv.org/pdf/2010.02338.pdf

https://arxiv.org/pdf/1804.07998.pdf

https://arxiv.org/pdf/1712.05419.pdf

My notes, if someone's interested