CS11-747 Neural Networks for NLP

Adversarial Methods

Zihang Dai & Qizhe Xie

Carnegie Mellon University
Language Technologies Institute

Site
https://phontron.com/class/nn4nlp2018/

Overview

- Generative Models (historical context)
- Generative Adversarial Networks (GANs)
- Generalized Adversarial Methods
- Applications in Text

Generative Models

- Model a data distribution P(X) or a conditional one P(X|Y)

- Typical approaches in **deep** generative models
  - **Auto-Regressive** Model: $P(X) = \prod_t P(X_t \mid X_{<t})$
    - e.g. RNN language model (RNNLM)
  - **Latent Variable** Model: $P(X) = \sum_Z P(X \mid Z)P(Z)$
    - e.g. Variational Auto-Encoder (VAE) - next lecture

What do we want from generative models?

- A “**perfect**” generative model
  - Evaluate **likelihood**: P(x)
    - e.g. Perplexity in language modeling
  - Generate **samples**: x ~ P(X)
    - e.g. Generate a sentence randomly from P(X) or conditioned on some other information using P(X|Y)
  - Infer **latent attributes**: P(Z|X)
    - e.g. Infer the “topic” of a sentence in topic models

No Generative Model is Perfect (so far)

	Auto-Reg. (PixelCNN)	RBM	VAE	GAN
Likelihood	☆☆☆☆	☆	☆☆	(☆)
Generation (image)	☆☆(☆)	☆	☆☆	☆☆☆☆
Inference		☆☆(☆)	☆☆☆	☆☆

- Mostly rely on **MLE** (Lower bound) based training
- **GANs** are particularly good at **generating** continuous **samples**

VAE vs. GAN

- Over-emphasis of **common** outputs, **fuzziness**

Real VAE GAN

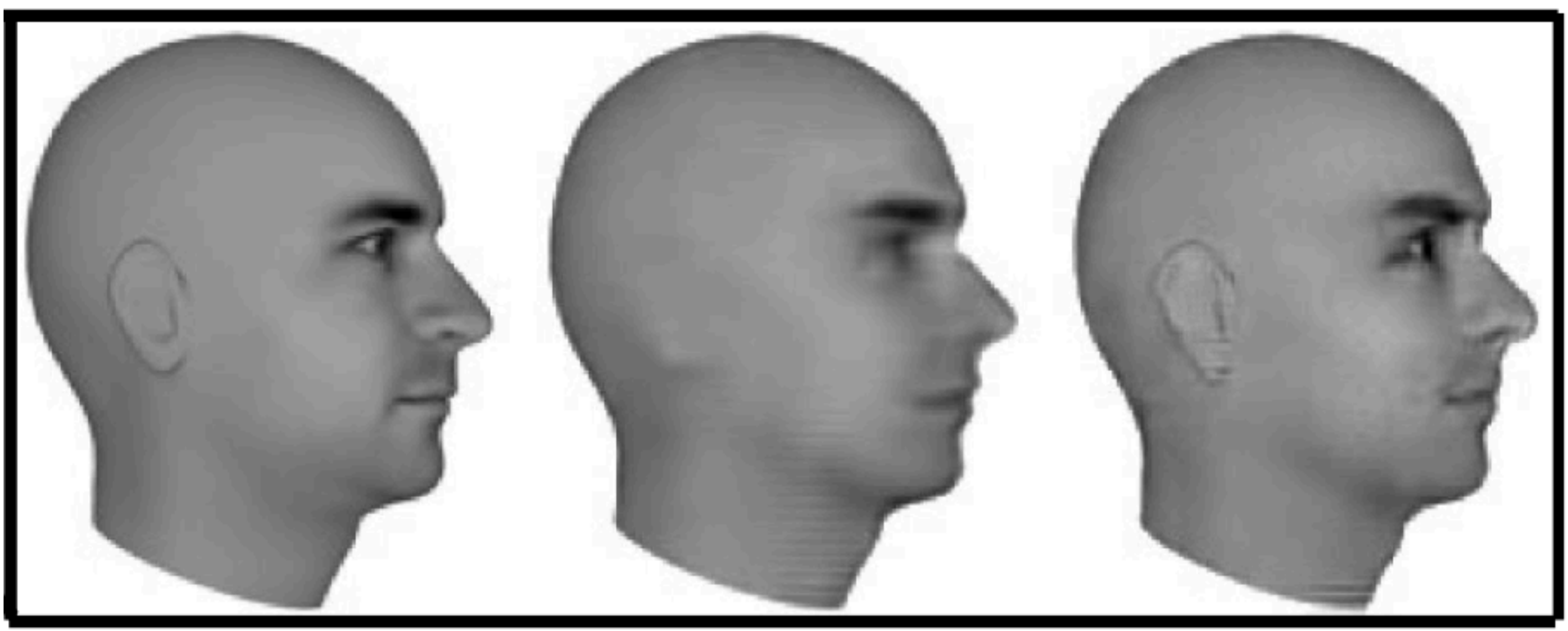
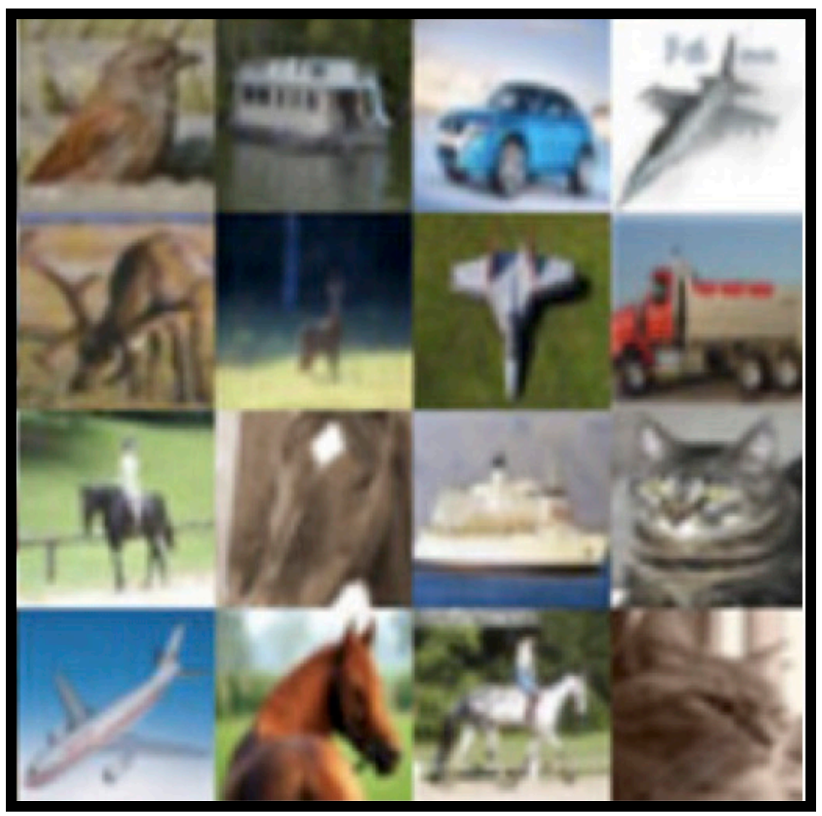


Image Credit: Lotter et al. 2015

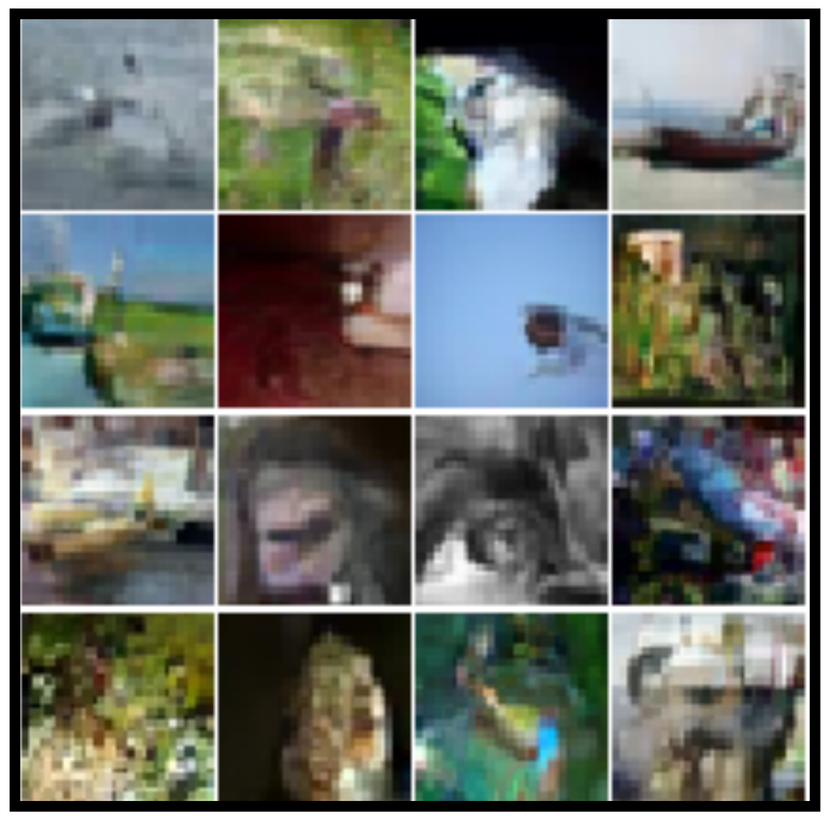
Auto-Reg. vs. GAN

- Local details vs. Global structure

Real



Auto-Reg.



GAN

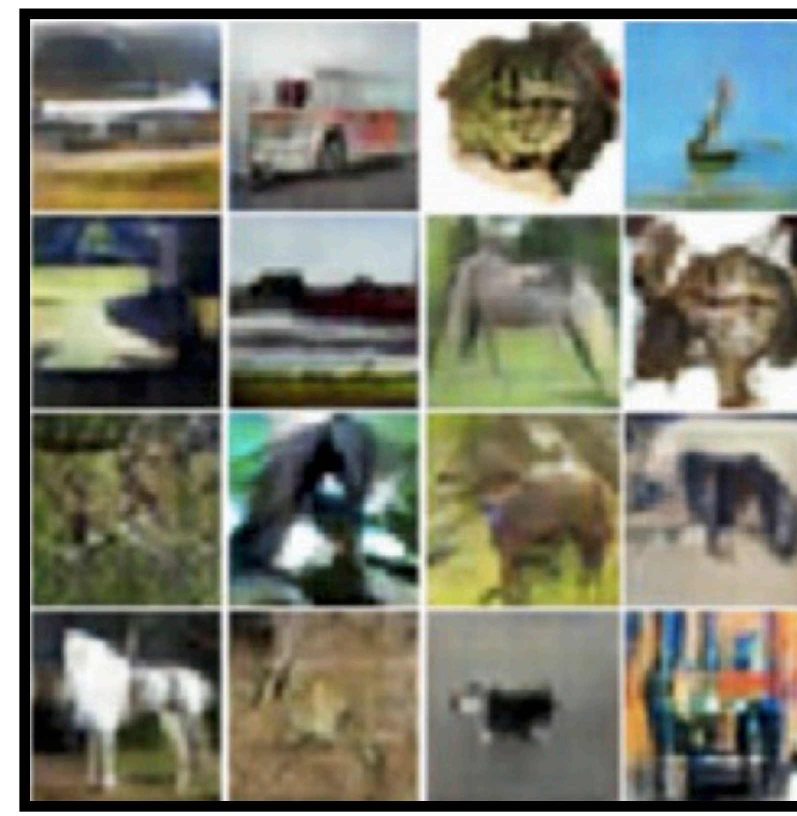


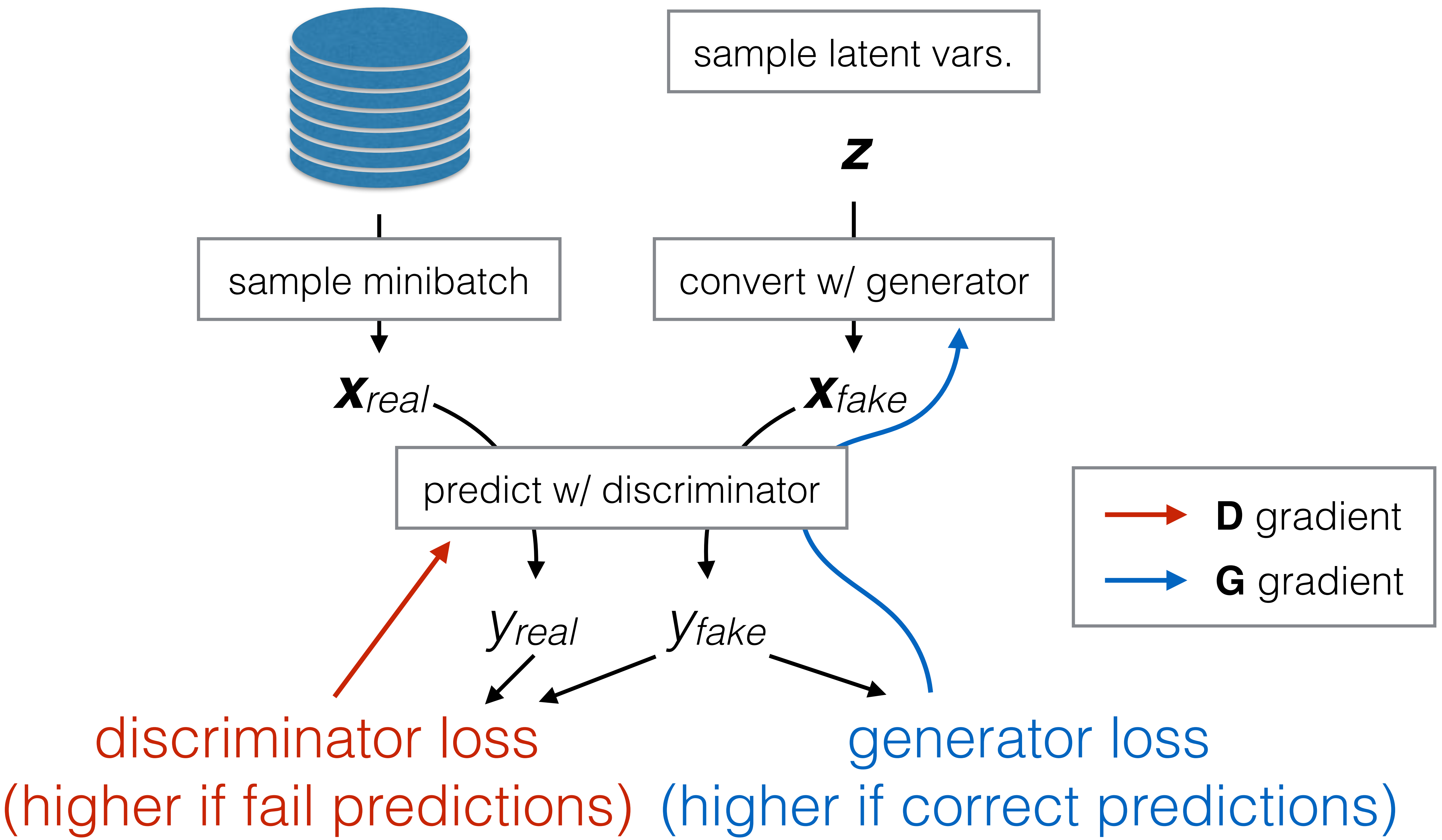
Image Credit: Salimans et al. 2016 (Improved GAN); 2017 (PixelCNN++)

Generative Adversarial Networks

Basic Paradigm

- Two players: generator and discriminator
  - **Discriminator:** given an image, try to tell whether it is real or not → P(image is real)
  - **Generator:** try to generate an image that fools the discriminator into answering “real”
- Desired result at convergence
  - Generator: generate perfect image
  - Discriminator: cannot tell the difference

Training Method



In Equations

- **Discriminator** loss function:

P(fake) = 1 - P(real)

$$\ell_D(\theta_D, \theta_G) = -\frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim P_{data}} \log D(\boldsymbol{x}) - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log(1 - D(G(\boldsymbol{z})))$$

Predict real for real data

Predict fake for fake data

- **Generator** loss function:
  - Make generate data **“less fake”** → Zero sum loss:

$$\ell_G(\theta_D, \theta_G) = -\ell_D(\theta_D, \theta_G)$$

  - Make generate data **“more real”** → Heuristic non-saturating loss:

$$\ell_G(\theta_D, \theta_G) = -\frac{1}{2}\mathbb{E}_{\boldsymbol{z}} \log D(G(\boldsymbol{z}))$$

  - **Latter** gives **better gradients** when discriminator accurate

In Pseudo-Code

- x_{real} ~ Training data
- $z \sim P(Z)$ → Normal(0, 1) or Uniform(-1, 1)
- $x_{fake} = \mathbf{G}(z)$
- $y_{real} = \mathbf{D}(x_{real})$ → $P(x_{real}$ is real)
- $y_{fake} = \mathbf{D}(x_{fake})$ → $P(x_{fake}$ is real)
- Train **D**: $\min_D - \log y_{real} - \log (1 - y_{fake})$
- Train **G**: $\min_G - \log y_{fake}$ → non-saturating loss

Why is GAN good?

- Discriminator is a **“learned metric”** parameterized by powerful neural networks
- Can easily pick up any kind of discrepancy, e.g. blurriness, global inconsistency
- Generator has **fine-grained** (gradient) signals to inform it what and how to improve

Problems in GAN Training

- GANs are great, but **training** is notoriously **difficult**
- Known problems
  - Convergence & Stability:
    - WGAN (Arjovsky et al., 2017)
    - WGAN-GP (Gulrajani et al., 2017)
    - Gradient-Based Regularization (Roth et al., 2017)
  - Mode collapse/dropping:
    - Mini-batch Discrimination (Salimans et al. 2016)
    - Unrolled GAN (Metz et al. 2016)
  - Overconfident discriminator:
    - One-side label smooth (Salimans et al. 2016)

Generalized Adversarial Methods

Implicit Distribution

Process

- [Step1] Z ~ P(Z), P(Z) can be any distribution
- [Step2] X = F(Z), F is a **deterministic** function

Result

- X is a random variable with an implicit distribution P(X), which decided by both P(Z) and F
- The process can produce any complicated distribution P(X) with a reasonable P(Z) and a powerful enough F

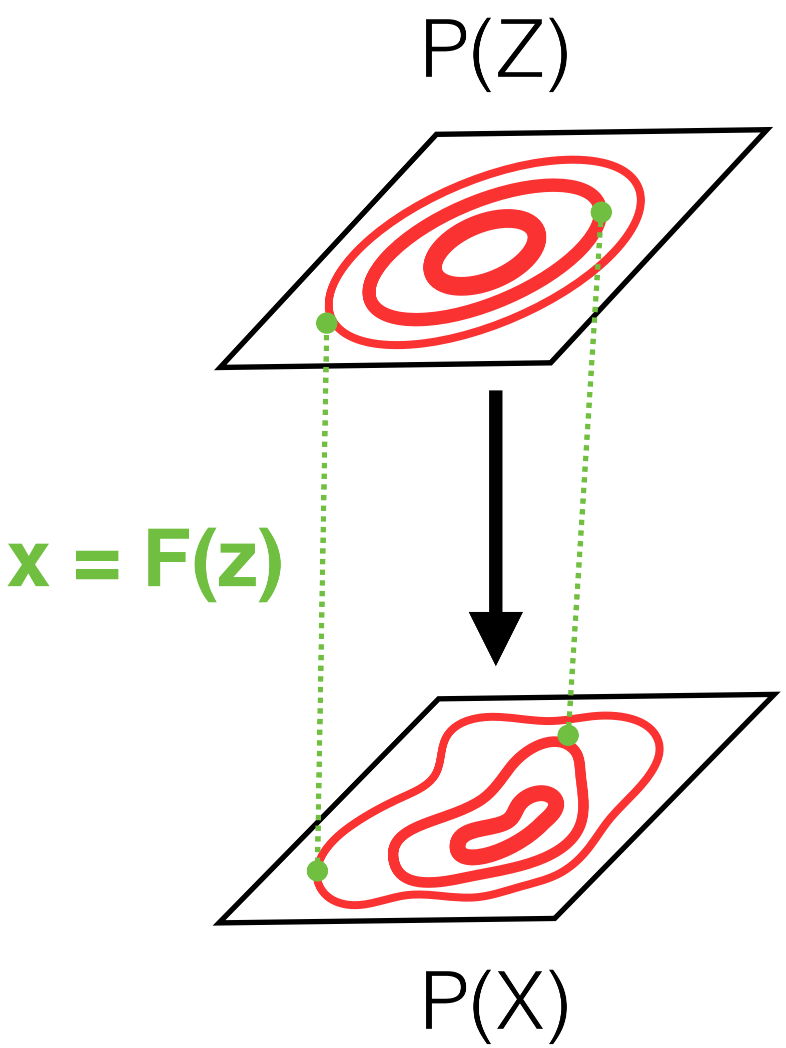


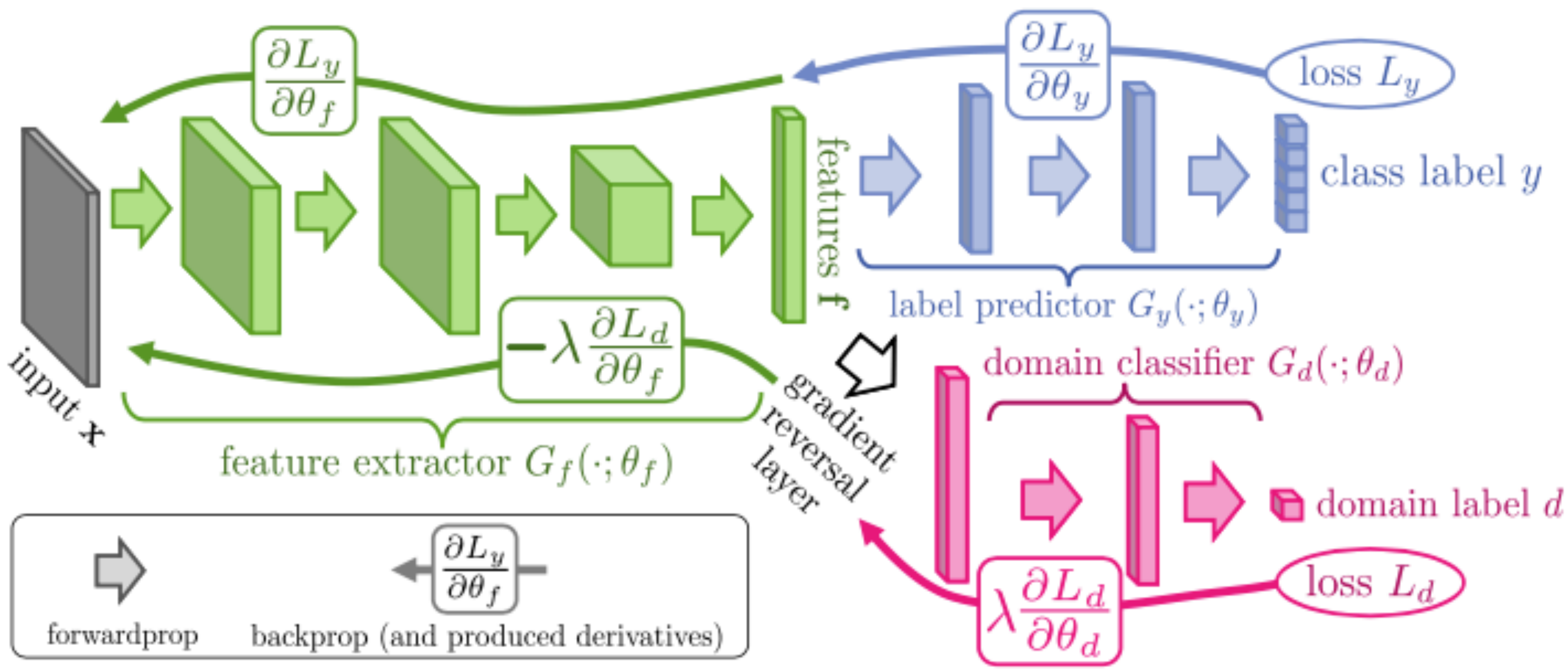
Image Credit: He et al. 2018

Distributional Matching via Samples

- Generator → Any model that produces “samples”
- Samples → **Anything** with an underlying distribution
  - hidden features, parameters, images/text
  - the distribution is often **implicit**
- Discriminator → Identify the <u>distributional differences</u>
  - as a **learned metric**
  - by checking real & fake **samples only**

Learning Domain-invariant Representations (Ganin et al. 2016)

- Learn features that cannot be distinguished by domain

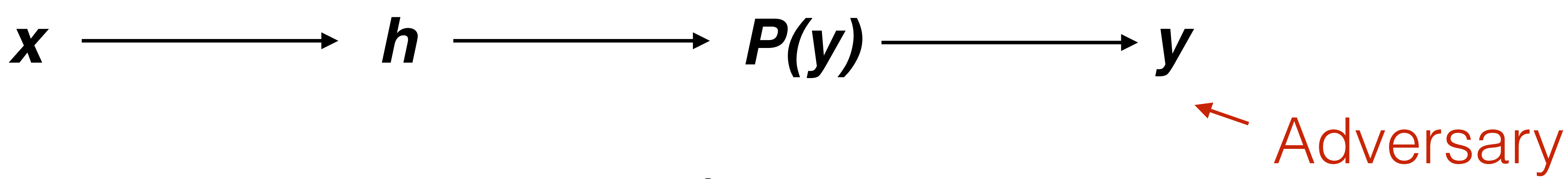


- Interesting application to synthetically generated or stale data (Kim et al. 2017)

Applying GANs to Text

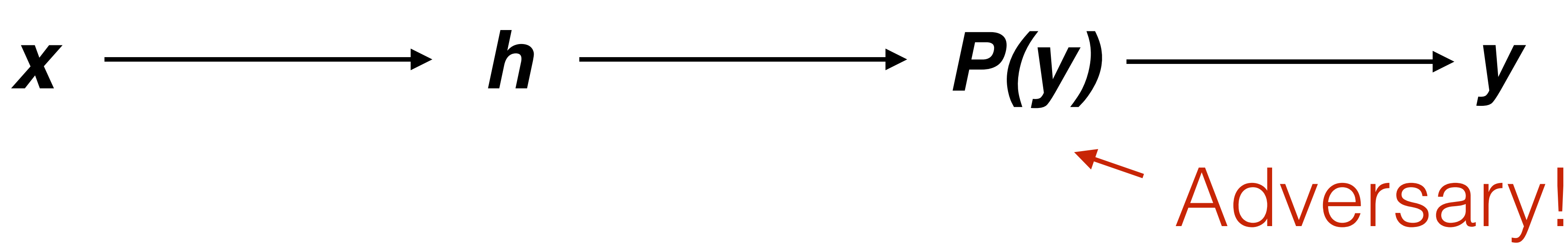
Adversarial Training Methods

- Generative adversarial networks



- Adversarial training over features

- Adversarial training over Softmax results

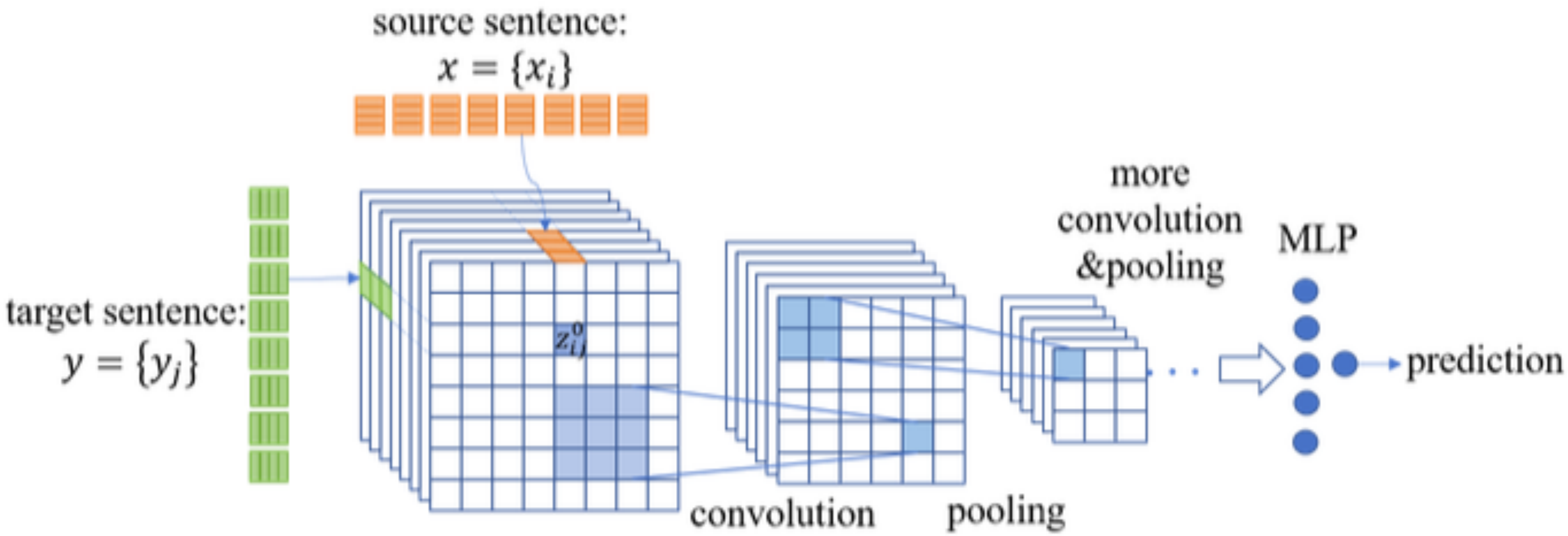


Applying GANs to Text

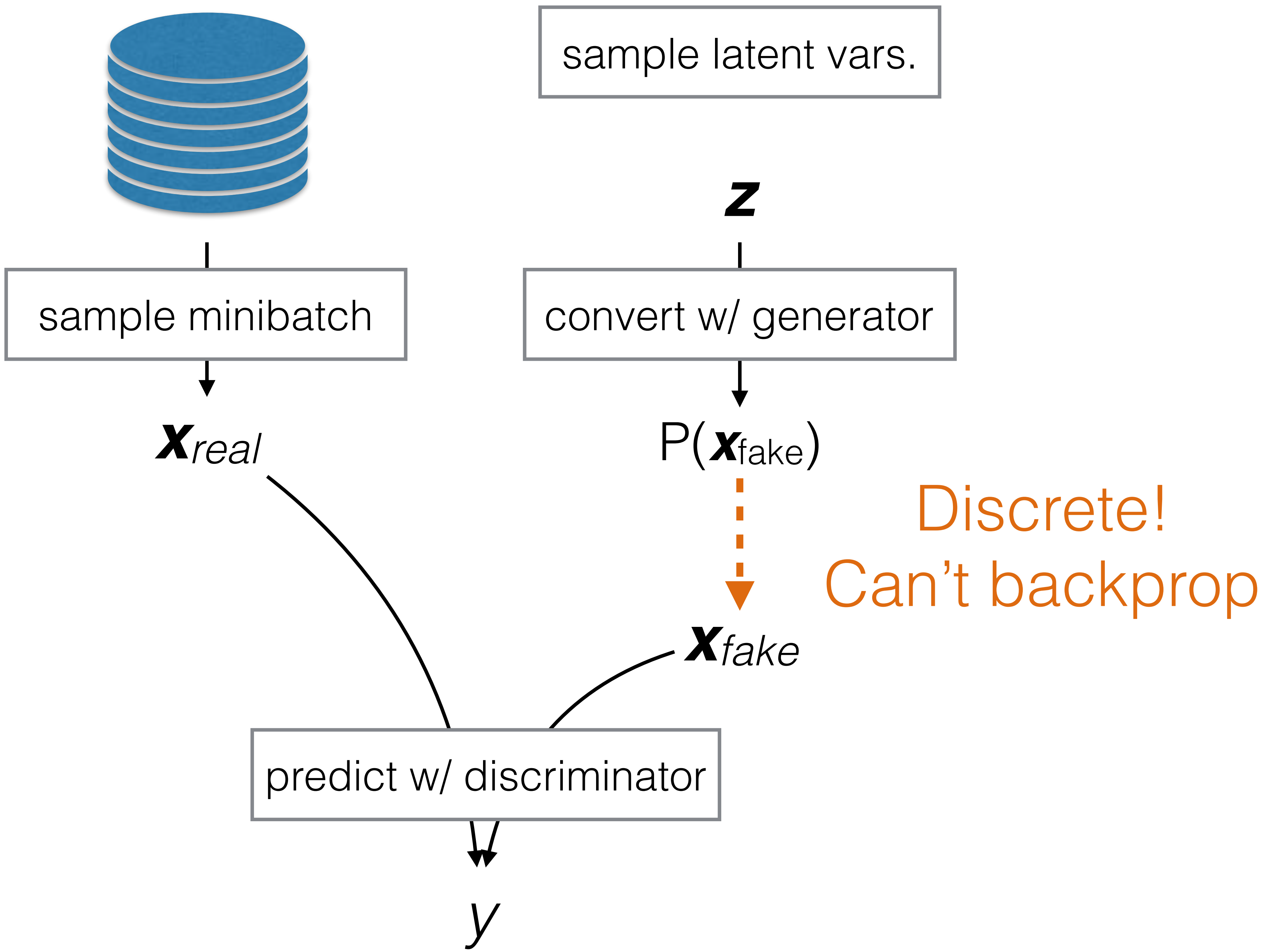
Adversarial Training over generated sentences (GAN)

Discriminators for Sequences

- Decide whether a particular generated output is true or not
- Commonly use CNNs as discriminators



Problem! Can't Backprop through Discrete Variables



Solution: Use Learning Methods for Discrete Latent Variables

- Policy gradient reinforcement learning methods (e.g. Yu et al. 2016)
- Reparameterization trick for latent variables using Straight-through Gumbel softmax (Gu et al. 2017)

Stabilization Trick: Assigning Reward to Specific Actions

- Getting a reward at the end of the sentence gives a credit assignment problem, leading to a high variance
- Solution: assign rewards for partial sequences (Yu et al. 2016, Li et al. 2017)

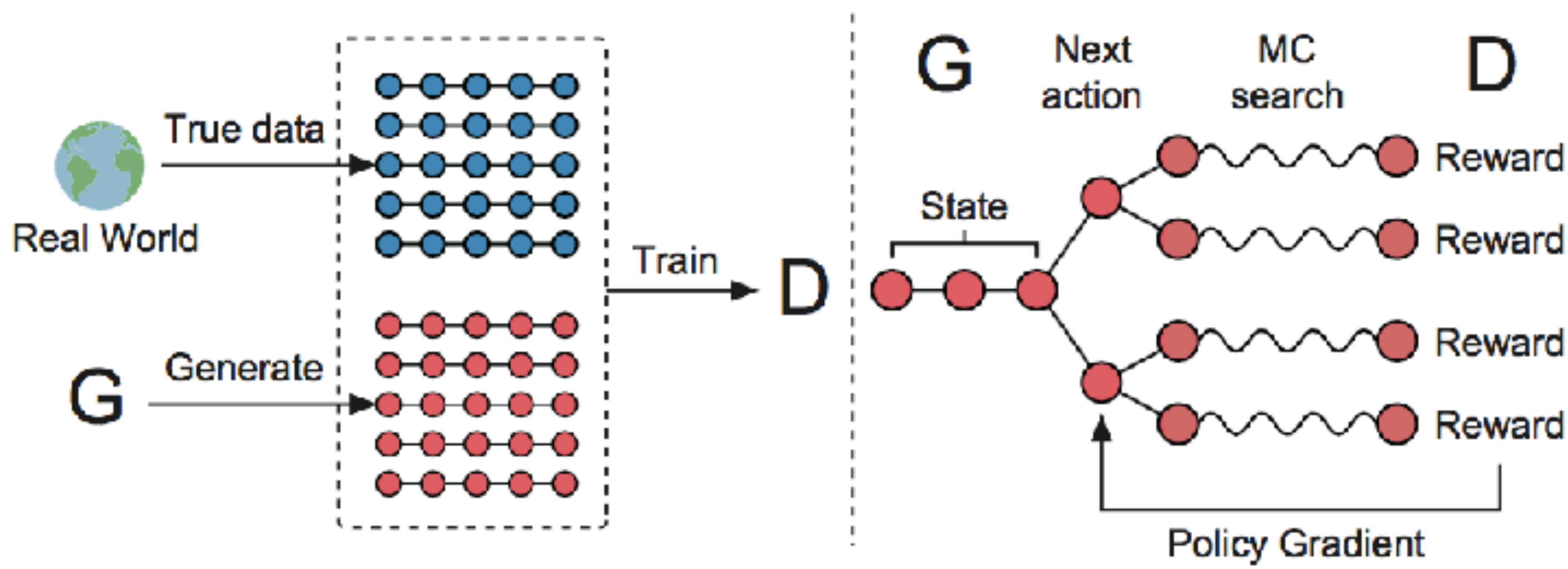
D(this)

D(this looks)

D(this looks do)

Stabilization Tricks: Performing Multiple Rollouts

- Instability is a severe problem
- High variance can be helped somewhat by doing multiple rollouts (Yu et al. 2016)
- Computationally heavy



Applications

- GANs for Language Generation (Yu et al. 2017)
- GANs for MT (Yang et al. 2017, Wu et al. 2017, Gu et al. 2017)
- GANs for Dialogue Generation (Li et al. 2016)

Strengths and Weaknesses

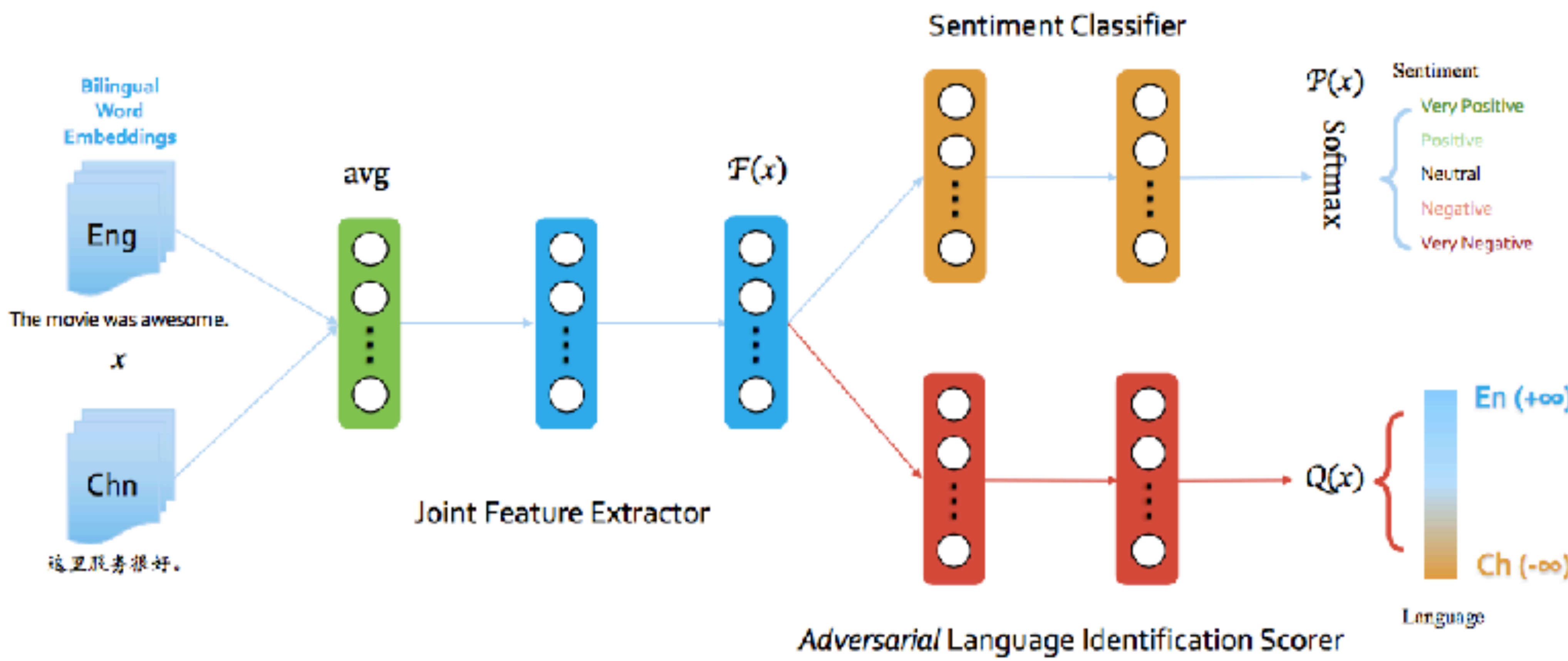
- Matching the distribution of generated sentences:
  - Pros: Unbiased (optimizing our final goal of generating natural sentences)
  - Cons: High variance (unstable), Sample inefficient (slow)
- Alternatives: Matching the distributions of features / Softmax results
  - Pros: Low variance, sample efficient
  - Cons: Biased (optimizing a surrogate objective)
  - Currently more widely used

Applying GANs to Text

Adversarial Training over features

Learning Language-invariant Representations

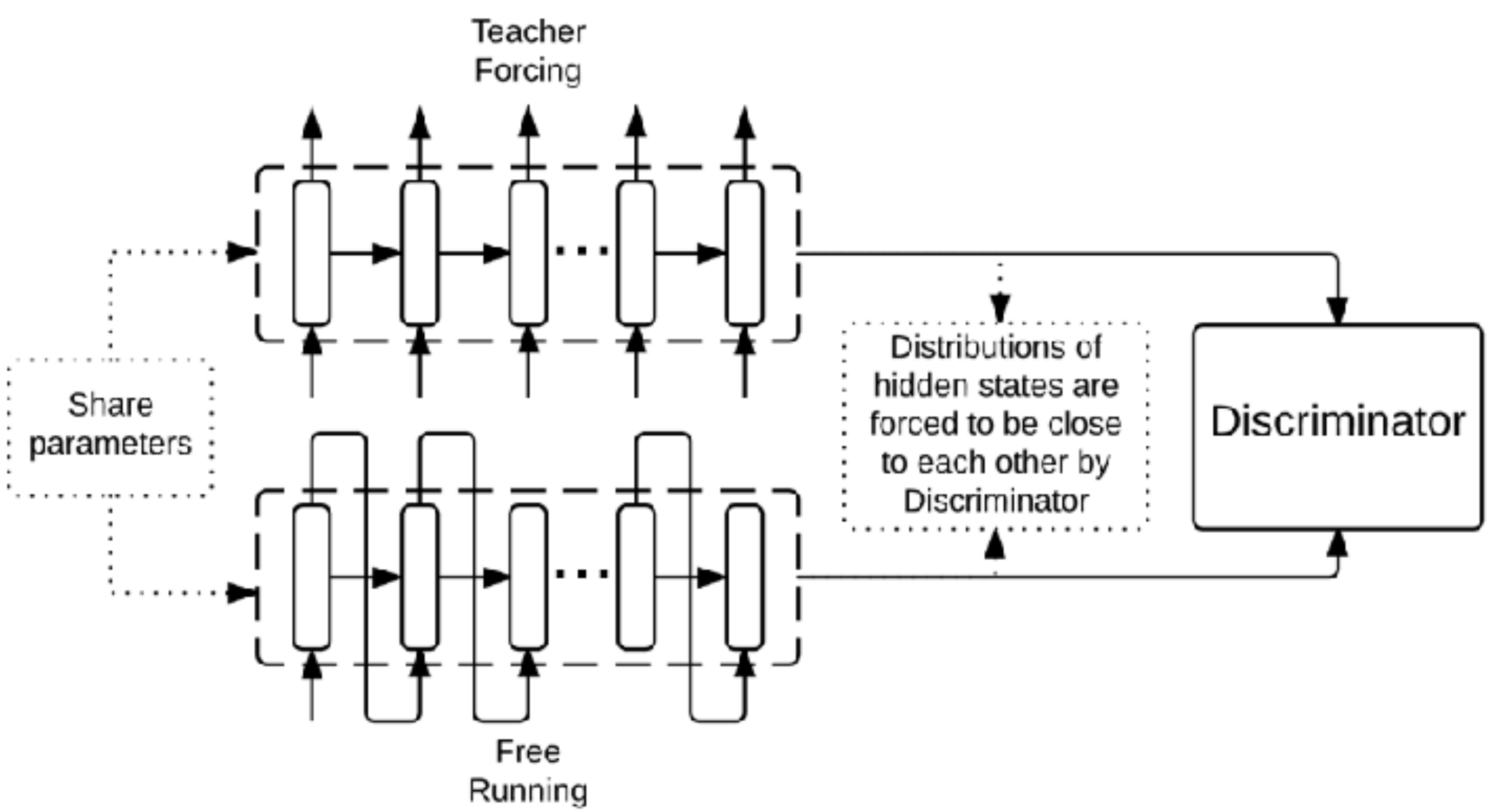
- Chen et al. (2016) learn language-invariant representations for text classification



- Also on multi-lingual machine translation (Xie et al. 2017)

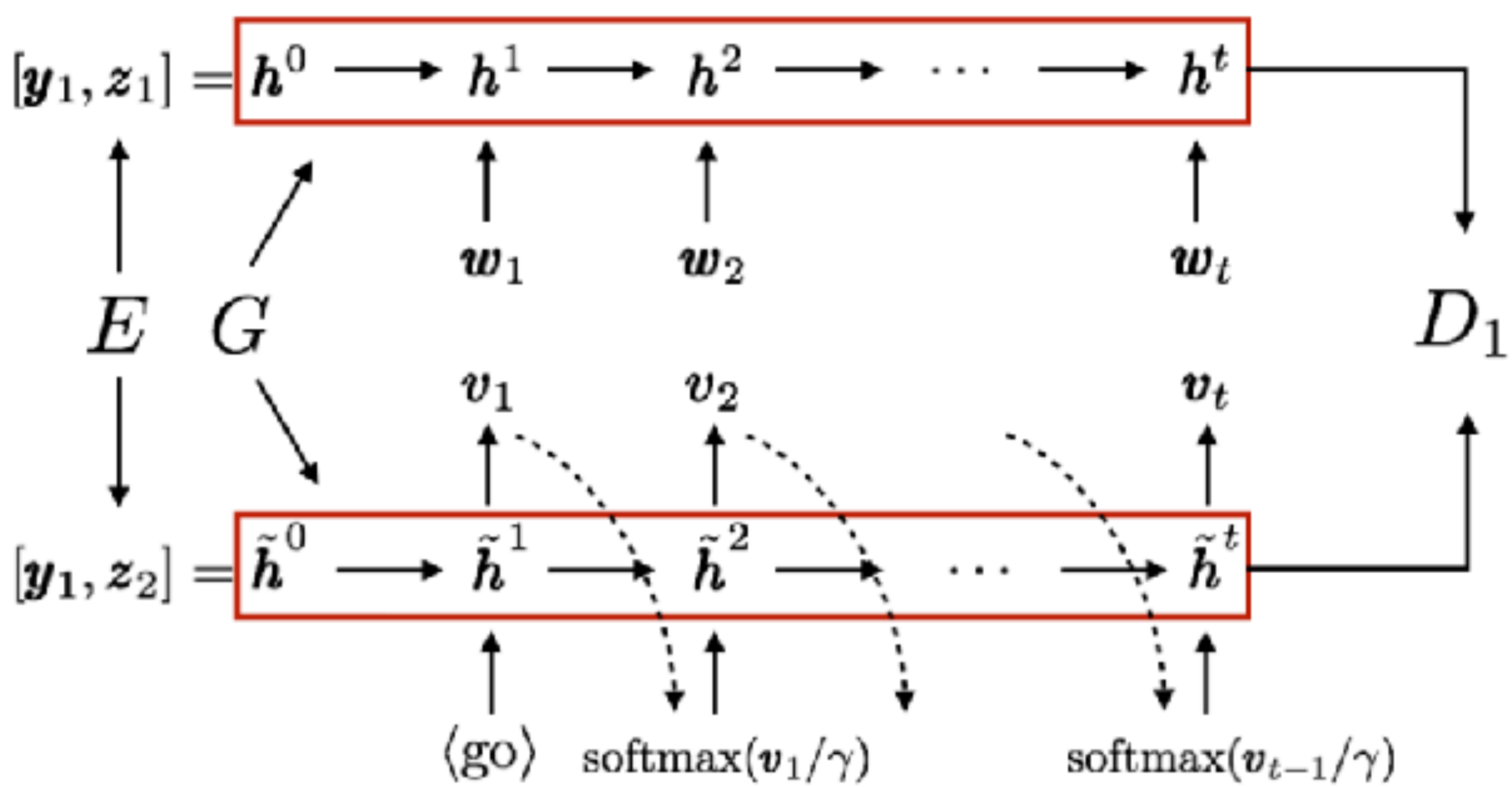
Professor Forcing
(Lamb et al. 2016)

- Tackles the exposure bias problem
  - Encourage the dynamics of the model to be the same at training time and inference time



Unsupervised Style Transfer for Text (Shen et al. 2017)

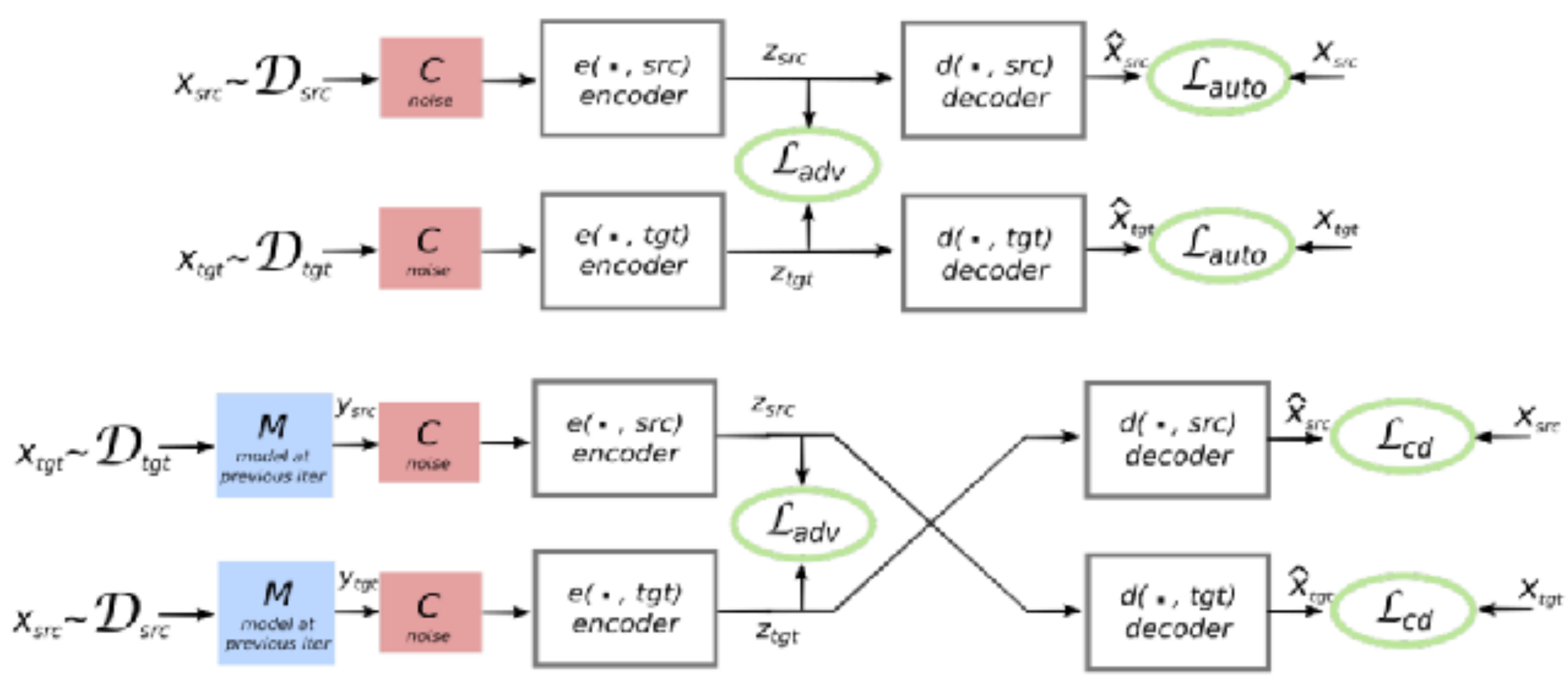
- Task: transfer sentences with one style to another style
  - Decipherment: Translate ciphered sentences to natural sentences (A simpler case of unsupervised MT)
  - Transfer sentences with positive sentiment to negative sentiment.
  - Word reordering
- Impressive performance on decipherment



Unsupervised Machine Translation

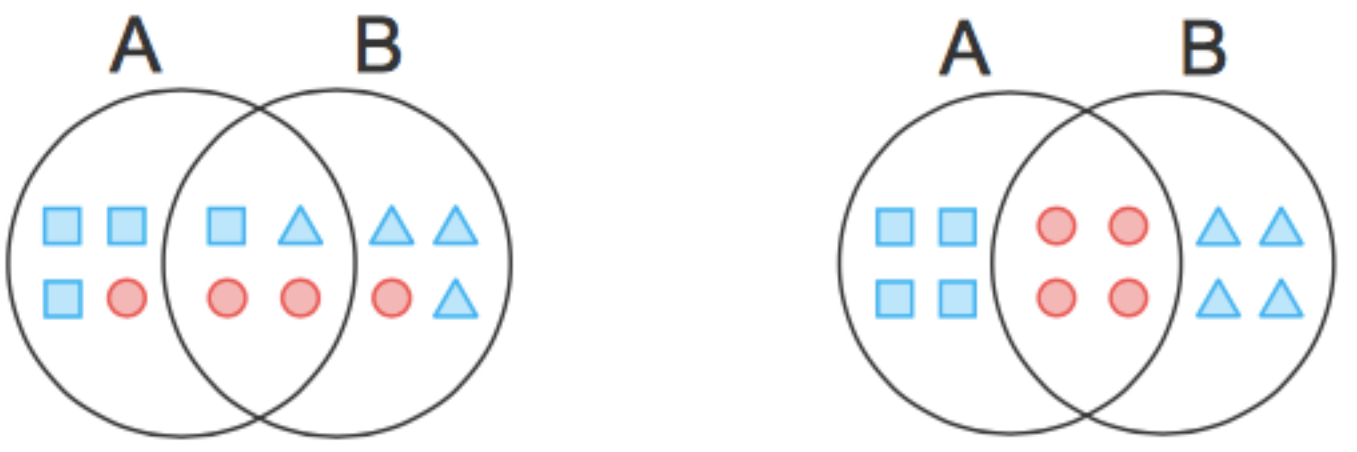
(Lample et al. 2017, Artetxe et al. 2017)

- Methods:
  - Cycle consistency (dual learning) (He et al. 2016, Zhu et al. 2017)
  - Employing denoising auto-encoder to refine translated sentence
- Performance on a par with supervised methods using 100k samples



Adversarial Multi-task Learning (Liu et al. 2017)

- Basic idea: want some features in a shared space across tasks, others separate



(a) Shared-Private Model (b) Adversarial Shared-Private Model

- Method: adversarial discriminator on shared features, orthogonality constraints on separate features

Applying GANs to Text

Adversarial Training over Softmax Results

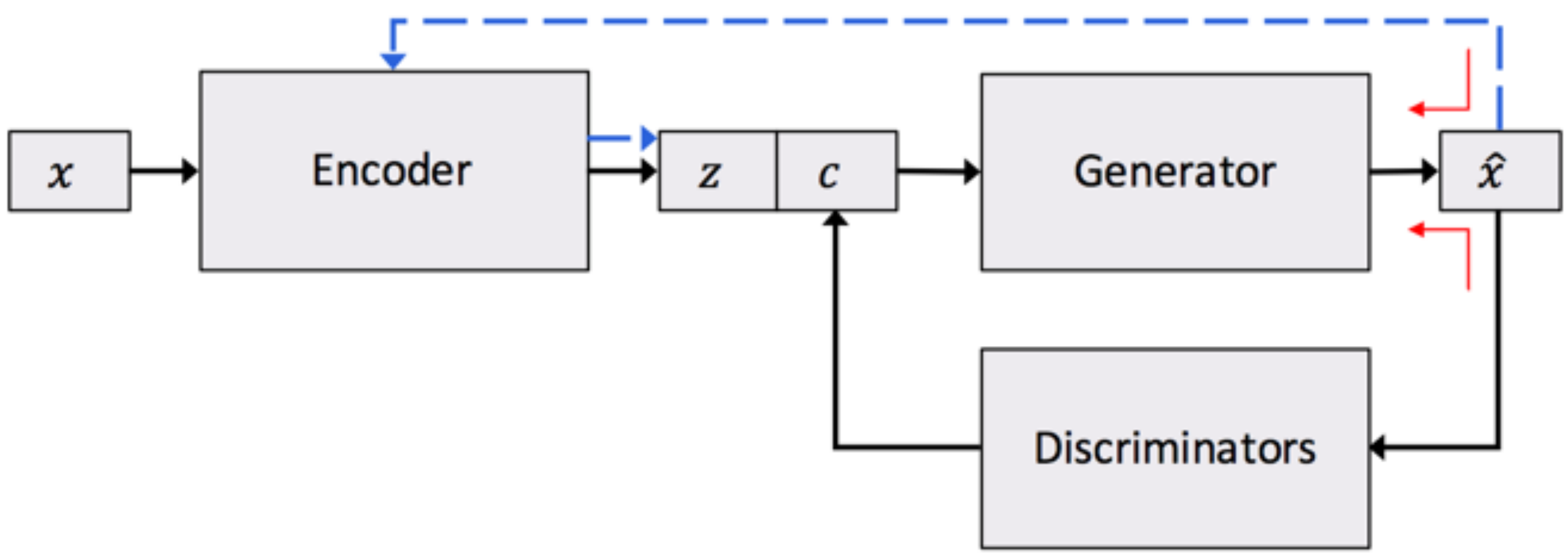
Adversarial Generation of Natural Language (Rajeswar et al. 2017)

- Unconditional generation of text with a **fixed length**
- Generator takes noise **Z** of shape [T x d] as input, and outputs the distribution **P(X)** of shape [T x V]
- Discriminator takes the **P(X)** of a fake generation or the **one-hot** representation of a real sample
- WGAN with GP regularization is crucial for training (Arjovsky et al., 2017, Gulrajani et al. 2017)
- Criticism: https://goo.gl/uNZtHm

Controlled Text Generation

(Hu et al. 2017)

- Separate the latent code of sentiment / tenses from the whole representation
- Propose to use the Softmax information
- Actually no adversarial training. Use cycle consistency to achieve latent code separation
- Great performance on modifying the sentiment / tenses of the sentence



Questions?