Generative Cooperative Networks for Natural Language Generation

Sylvain Lamprier, Thomas Scialom, Antoine Chaffin, Vincent Claveau, Ewa Kijak, Benjamin Piwowarski, Jacopo Staiano

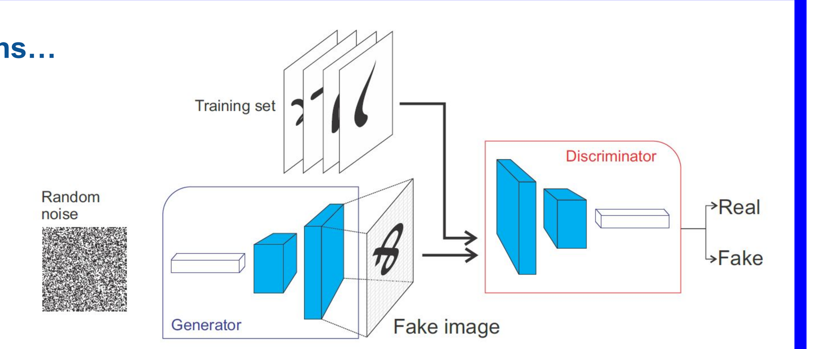
Sorbonne Université – Créateurs de futurs depuis 1257 · MLIA Machine Learning & Deep Learning for Information Access · reciTAL. · IMATAG · UMR IRISA

Language GANs fall short

GANs are good for approximating continuous data distributions…

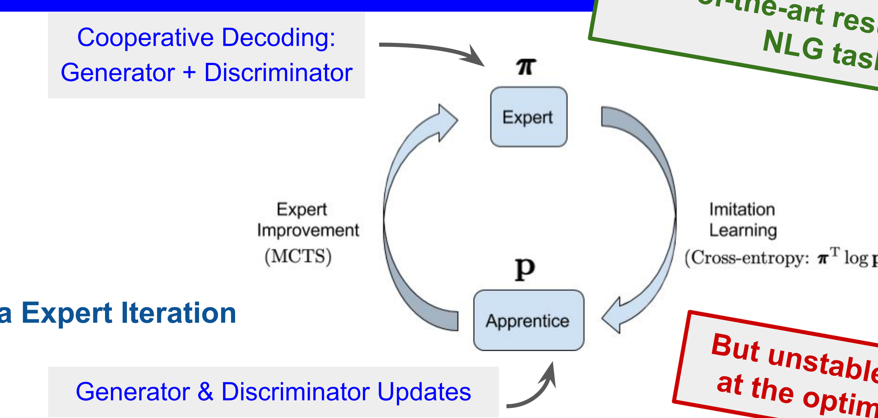
… but have hard time with discrete data (e.g., text):

- No backpropagation from the discriminator to the generator :
  - Reinforcement Learning with Discriminator scores as Rewards
    - Noisy and Moving Rewards
  - Existing language GANs are known to fall short (Caccia et al 2020)
  - Cautious Sampling is a key to stabilize the process (Scialom et al., 2020a)



Cooperative Decoding

- Use of the **discriminator D cooperatively with the generator p** for sampling texts
  - In Beam Search:
    - DAS (Scialom et. al, 2020b)
    - Discriminative EBM (Bakhtin et al., 2021)
  - In MCTS:
    - SelfGAN (Scialom et. al, 2021)
- **SelfGAN: Cooperative decoding at train time via Expert Iteration**
  - Denser Rewards
  - More Realistic Samples



State-of-the-art results on various NLG tasks !

But unstable even at the optimum !

Generative Cooperative Networks (GCN, this work)

- **Based on Reward-augmented Maximum Likelihood (RML)** (Norouzi et al., 2016):
  - considers a Boltzmann distribution $q(x) \propto \frac{exp(f(x))}{\tau}$ with $f(x)$ a reward dependent function and τ a temperature
  - updates the generator p via: $min_p KL(q||p)$
- **Our GCN considers** $f(x) = \log(p_{t-1}(x)) + \log(D_t(x))$
  - p_{t-1} is the generator at previous iteration
  - D_t is a discriminator trained with samples generated from p_{t-1}
- **Variance reduction via Cooperative Decoding with MCTS and Weighted Importance Sampling**

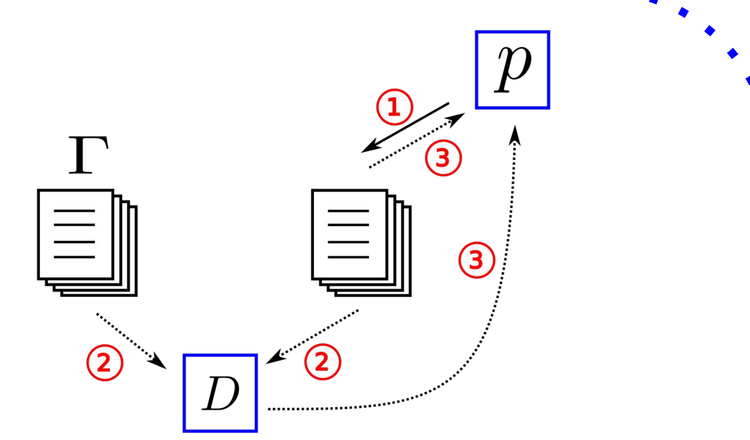
⇒ Ensures asymptotic convergence under usual assumptions of GANs !

⇒ Avoids Catastrophic Forgetting (e.g., SelfGAN)

Discrete-GAN

Algorithm *Discrete-GAN*

Input: a generator p_θ, a discriminator family $\mathcal{D}$.
for iteration t from 1 to T **do**
$D_t \leftarrow \arg\max_{D \in \mathcal{D}} \left[\mathbb{E}_{y \sim p_d(y)} [\log D(y)] + \mathbb{E}_{y \sim p_\theta(y)} [\log(1 - D(y))] \right]$
$\theta \leftarrow \theta + \epsilon \mathbb{E}_{y \sim p_\theta(y)} [D_t(y) \nabla_\theta \log p_\theta(y)]$
end for



① Sample M documents from generator p
$y^i \sim p_\theta(y^i)$
② Train the discriminator D_t
③ Train p with rewards from discriminator D on generated samples (policy gradient)

$$\theta \leftarrow \theta + \epsilon \frac{1}{M} \sum_{i=1}^{M} D_t(y^i) \nabla_\theta \log p_\theta(y^i)$$

- **Uses directly the Generator p to produce training samples**
  - Very unstable, sparse rewards
  - Very noisy discriminator at the mode of the distribution p (Scialom et al., 2020a)
- **Requires a scheduler to avoid divergence**

RML-GAN (this work)

Algorithm *RML-GAN*

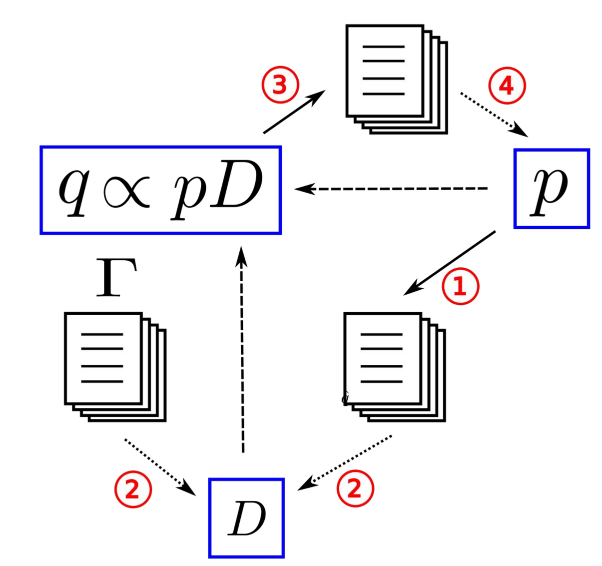
Input: a generator $p_0 \in \mathcal{G}$, a discriminator family $\mathcal{D}$.
for iteration t from 1 to T **do**
$D_t \leftarrow \arg\max_{D \in \mathcal{D}} \left[\mathbb{E}_{y \sim p_d(y)} [\log D(y)] + \mathbb{E}_{y \sim p_{t-1}(y)} [\log(1 - D(y))] \right]$
$p_t \leftarrow \arg\min_{p \in \mathcal{G}} KL(q_t \propto p_{t-1} D_t || p)$
end for

Guaranteed Convergence !!

$$KL(p_d||p_t) - KL(p_d||p_{t-1}) \leq \log(\tfrac{1}{\eta} - 1) < 0$$

with $\log \eta = \min \left(\mathbb{E}_{y \sim p_d(y)} [\log(D_t(y))], \mathbb{E}_{y \sim p_{t-1}(y)} [\log(1 - D_t(y))] \right)$

But how to sample from q ?



① Sample documents from generator p
② Train the discriminator D
③ Generate M samples from $q \propto pD$
$y^i \sim q(y^i)$
④ Train p using samples from q

$$\theta \leftarrow \theta + \epsilon \frac{1}{M} \sum_{i=1}^{M} \nabla_\theta \log p_\theta(y^i)$$

GCN (this work)

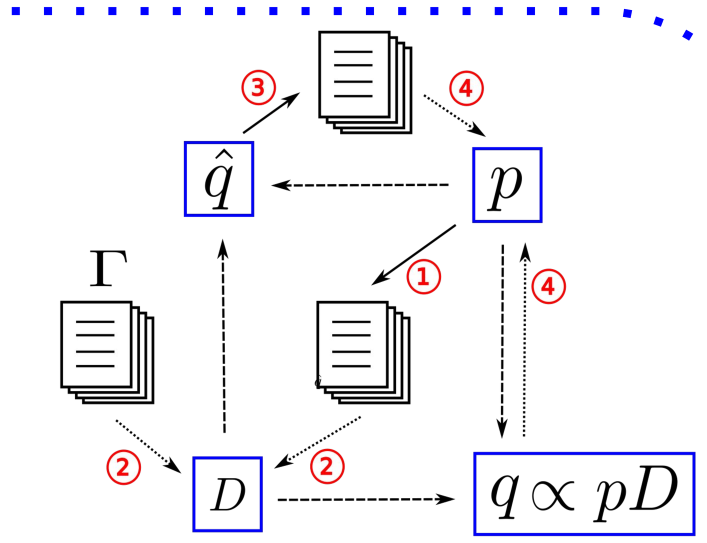
Algorithm 2 *Generative Cooperative Networks*

Input: generator p_θ with parameters θ, discriminator D_ϕ with parameters ϕ, training set Γ, sampling strategy $\hat{q}$, batch size m, max sequence length l.
for $t = 1, \dots, T$ **do**
Sample $\{(x^i, y^i)\}_{i=1}^m$ from Γ
$\forall i \in [\![1\,;m]\!]$: Sample $\hat{y}^i \sim p_\theta(\hat{y}^i|x^i)$;
$\phi \leftarrow \phi + \epsilon_\phi \sum_{i=1}^{m} \sum_{j=1}^{l} \left[\nabla_\phi \log D_\phi(x^i, y^i_{0:j-1})] + \nabla_\phi \log(1 - D_\phi(x^i, \hat{y}^i_{0:j-1})) \right]$
$\forall i \in [\![1\,;m]\!]$: Sample $\hat{y}^i \sim \hat{q}(\hat{y}^i|x^i)$;
$\theta \leftarrow \theta + \epsilon_\theta \left[\frac{1}{\sum_{i=1}^m w^i} \sum_{i=1}^m w^i \nabla_\theta \log p_\theta(\hat{y}^i|x^i) \right]$
with $w^i = \frac{p_\theta(\hat{y}^i|x^i) D_\phi(x^i, \hat{y}^i)}{\hat{q}(\hat{y}^i|x^i)}$
end for

Automatic Scheduling !

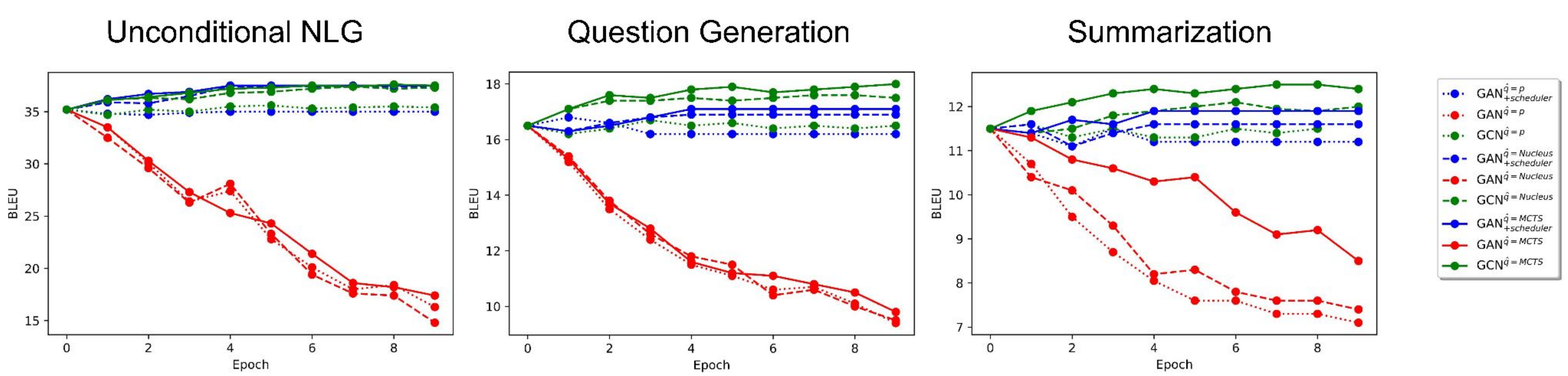
⇒ With $\hat{q} = p_\theta$, we have:

$$\theta \leftarrow \theta + \epsilon \frac{1}{Z_t} \sum_{i=1}^{M} D_t(y^i) \nabla_\theta \log p_\theta(y^i), \text{ where } Z_t = \sum_{i=1}^{M} p_\theta(y^i) D_t(y^i)$$



① Sample documents from generator p
② Train the discriminator D
③ Generate M samples from $\hat{q}$
$y^i \sim \hat{q}(y^i)$
④ Train p using weighted importance sampling

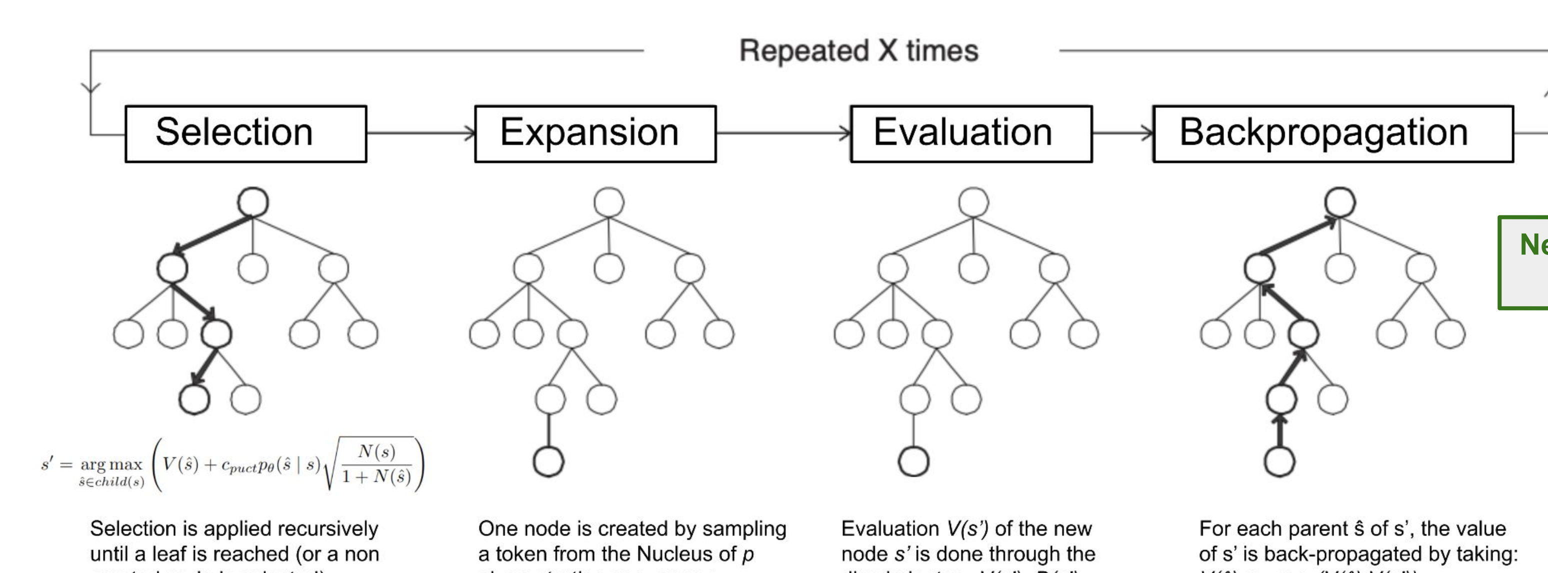
$$\theta \leftarrow \theta + \epsilon \frac{1}{\sum_{i=1}^{M} w^i} \sum_{i=1}^{M} w^i \nabla_\theta \log p_\theta(y^i) \quad \text{with: } w^i = \frac{q(y^i)}{\hat{q}(y^i)}$$



	QG			Summarization		
	B	R-1	R-L	B	R-1	R-L
MLE	19.7	45.2	41.1	15.9	42.3	40.4
ColdGAN	19.9	45.2	41.4	16.3	42.8	40.7
SelfGAN	20.5	46.6	42.6	17.0	42.8	41.5
$GAN^{\hat{q}=p}_{+scheduler}$	19.3	45.3	41.2	15.5	40.0	38.8
$GAN^{\hat{q}=p}$	11.2	26.3	23.9	9.8	23.3	22.5
$GCN^{\hat{q}=p}$	19.7	46.2	42.0	15.9	40.8	39.5
$GAN^{\hat{q}=Nucleus}_{+scheduler}$	20.1	47.3	43.0	16.0	41.8	40.4
$GAN^{\hat{q}=Nucleus}$	11.3	26.6	24.1	10.2	23.5	22.7
$GCN^{\hat{q}=Nucleus}$	20.9	47.7	44.5	16.6	43.2	41.8
$GAN^{\hat{q}=MCTS}_{+scheduler}$	20.4	47.9	43.5	16.4	42.2	40.9
$GAN^{\hat{q}=MCTS}$	11.7	27.5	25.0	11.7	24.3	23.4
$GCN^{\hat{q}=MCTS}$	**21.5**	**48.3**	**44.7**	**17.1**	**43.4**	**42.0**
$GCN^{\hat{q}=MCTS}_{decod=mcts}$	**21.6**	**48.7**	**45.2**	**17.6**	**43.7**	**42.3**
$GCN^{\hat{q}=MCTS}_{T5-3B}$	**21.8**	**49.8**	**45.9**	**19.2**	**44.2**	**43.8**

Table 1. Final results on QG and Summarization test sets, in terms of BLEU-4 (B), ROUGE-1 (R-1) and ROUGE-L (R-L). Scores in bold are significantly different from the best baseline ($GAN^{\hat{q}=MCTS}_{+scheduler}$) according to a 95%-Student-t-test.

- Z_t **acts as a scheduler**
  - It increases along with agreements between the generator and the discriminator about best sequences
- **GCN with** $\hat{q} = p$ **obtains good and stable results without the need of a scheduler**
  - But sampling closer to the target q allows to get better results !!
- **The use of Nucleus Sampling (Holtzman et al., 2019) allows to improve results by avoiding sampling from the tail of the distribution p**
- **Monte-Carlo Tree Search allows to go further by circumventing the left-to-right curse of myopic approaches**



New State-of-the-art Results !