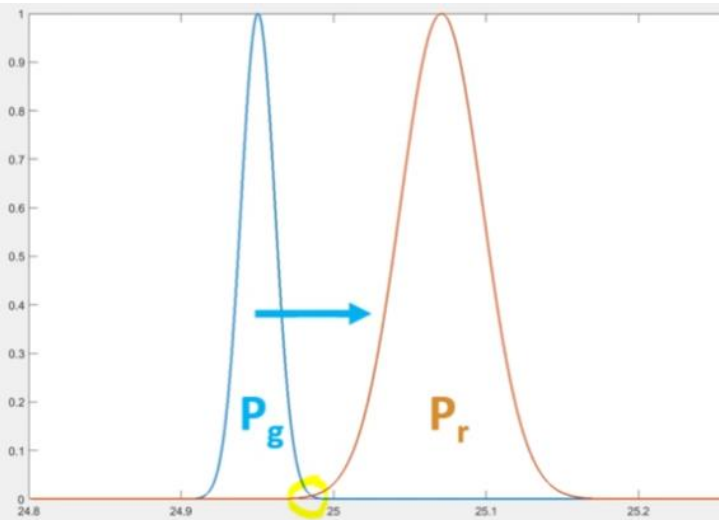
WGAN prevents mode collapse (prevalent classes)

High level idea in GANs:

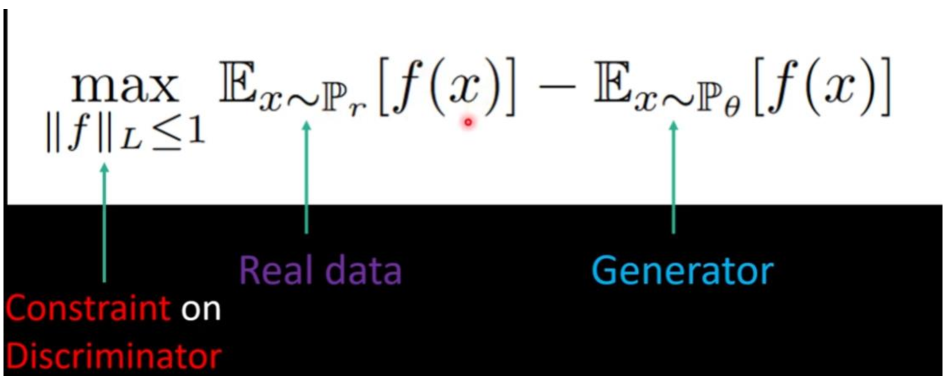
Two probability distributions, Pg and Pr (Distribution from generator and real).
We want Pg and Pr to be very similar to generate realistic looking images.



We want Pg and Pr to be as close as possible.

GAN Loss used is equivalent to Jensen-Shannon divergence.
JS divergence has gradient issues leading to unstable training.

WGAN uses Wasserstein Distance.



Discriminator wants to separate two distributions as much as possible (maximize)
Generator wants to put these as closer to each other as possible (minimize)

Lipshitz Constraint is applied to prevent gradient exploding and leading to instability.

To enforce Lipshitz constraint,

WGAN uses weight clipping, However,
Large weight clipping results in long time for any weight to reach its limit, making it harder to train the critic until optimality.
Smaller weight clipping leads to vanishing gradient when layers are deep or batch normalization is not used (such as in RNNs)

WGAN-GP uses softer Lipshitz constraint by penalizing large gradients.

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

Require: : α, the learning rate. c, the clipping parameter. m, the batch size. n_{critic}, the number of iterations of the critic per generator iteration.
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.
1: **while** θ has not converged **do**
2: **for** $t = 0, ..., n_{\text{critic}}$ **do**
3: Sample $\{x^{(i)}\}_{i=1}^{m} \sim \mathbb{P}_r$ a batch from the real data.
4: Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
5: $g_w \leftarrow \nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
6: $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$
7: $w \leftarrow \text{clip}(w, -c, c)$
8: **end for**
9: Sample $\{z^{(i)}\}_{i=1}^{m} \sim p(z)$ a batch of prior samples.
10: $g_\theta \leftarrow -\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$
11: $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$
12: **end while**

When critic's loss is near zero, it is metric for convergence.
It trains the critic more than the generator (5 to 1)
Weight clipping is done to adhere to the critic constraint ||f||L <= 1

Line 6 Maximize (+)
Line 11 Minimize (-)

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

Require: The gradient penalty coefficient λ, the number of critic iterations per generator iteration n_{critic}, the batch size m, Adam hyperparameters α, β_1, β_2.
Require: initial critic parameters w_0, initial generator parameters θ_0.

1: **while** θ has not converged **do**
2:     **for** $t = 1, ..., n_{\text{critic}}$ **do**
3:         **for** $i = 1, ..., m$ **do**
4:             Sample real data $\boldsymbol{x} \sim \mathbb{P}_r$, latent variable $\boldsymbol{z} \sim p(\boldsymbol{z})$, a random number $\epsilon \sim U[0, 1]$.
5:             $\tilde{\boldsymbol{x}} \leftarrow G_\theta(\boldsymbol{z})$
6:             $\hat{\boldsymbol{x}} \leftarrow \epsilon \boldsymbol{x} + (1 - \epsilon)\tilde{\boldsymbol{x}}$
7:             $L^{(i)} \leftarrow D_w(\tilde{\boldsymbol{x}}) - D_w(\boldsymbol{x}) + \lambda(\|\nabla_{\hat{\boldsymbol{x}}} D_w(\hat{\boldsymbol{x}})\|_2 - 1)^2$
8:         **end for**
9:         $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^{m} L^{(i)}, w, \alpha, \beta_1, \beta_2)$
10:     **end for**
11:     Sample a batch of latent variables $\{\boldsymbol{z}^{(i)}\}_{i=1}^{m} \sim p(\boldsymbol{z})$.
12:     $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^{m} -D_w(G_\theta(\boldsymbol{z})), \theta, \alpha, \beta_1, \beta_2)$
13: **end while**

Adam optimizer is used, but momentum is removed. (Beta1 = 0)

Line6 -> Interpolation between real image and generate image
The "percentage" of interpolation between two are randomly decided by the epsillon
Taking norm of the gradient of the interpolation guarantees Lipshitz constraints.

This way, the weight of the critics are not hard limited unlike WGAN with clipping.