Գեներատիվ հակամարտող ցանցեր

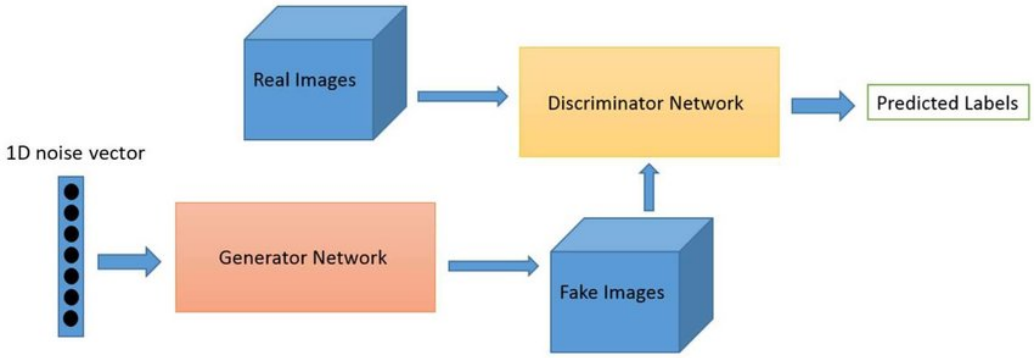
Հայկ Կարապետյան

Ունենք հետևյալ խնդիրը։ Հարկավոր է գեներացնել նոր նկարներ, մուտքում ստանալով աղմուկ վեկտոր (noisy vector)։ Variational autoencoder-ի միջոցով կարող ենք այս խնդիրը լուծել, երբ մուտքային վեկտորը գաուսյան բաշխումից է։ Այժմ ծանոթանանք ևս մեկ նեյրոնային ցանցերի կառուցվածքի հետ, որը կոչվում է գեներատիվ հակամարտող ցանց (generative adversarial network GAN)։ Այս ցանցը բաղկացած է երկու մասից՝ Generator և Discriminator։ Discriminator-ը մուտքում ստանում է նկար և պետք է տարբերակի այդ նկարը կեղծ է, թե իրական։ Օրինակ, երբ մուտքում ստանա մարդու դեմք, որտեղ աչքերի փոխարեն ականջներ են, ապա պետք է վերադարձնի, որ այդ նկարը կեղծ է։ Generator-ի նպատակն է աղմուկ վեկտորից, գեներացնել այնպիսի նկարներ, որ Discriminator-ը չհասկանա նկարը կեղծ է։ Այսինքն այն չպետք է աչքերի փոխարեն ականջներ գեներացնի։
Կան GAN-երի բազմաթիվ տեսակներ՝

1. Simple GAN
2. Deep Convolution GAN (DCGAN)
3. Conditional GAN
4. CycleGAN
5. Disco GAN
6. Wasserstein GAN
7. Drag GAN

Այս լեկցիա ընթացքում քննարկելու ենք GAN-երի տեսակներից առաջին երեքը։

1. Simple GAN



Նկար 1: Simple GAN ցանցի օրինակ

Դիտարկենք նկար 1-ը։ Generator-ի հատվածը մուտքում ստանում է աղմուկ վեկտոր և գեներացնում է նկար։ Discriminator-ի հատվածը մուտքում ստանում է և՛ իրական նկարներ, և՛ կեղծ նկարներ։ Իրականում այն չունի երկու մուտքային տվյալ, այն կա՛մ ստանում է իրական նկար, կա՛մ ստանում է կեղծ նկար։ Մենք կարող ենք Discriminator-ի հատվածը ուսուցանել հետևյալ կերպ։ Այն կլինի սովորական դասակարգման ցանց, որը մուտքում իրական նկար ստանալիս կվերադարձնի 1, իսկ կեղծ նկար ստանալիս կվերադարձնի 0։ Այդպիսի եղանակով կարող ենք թարմացնել Discriminator-ի կշիռները։ Իսկ Generator-ի կշիռները թարմացնելու ենք հետևյալ կերպ։ Կշիռներն այնպես ենք փոխելու (gradient descent), որպեսզի Discriminator-ը չհասկանա, որ նկարը կեղծ է։ Ֆիքսենք, որ Generator-ի գեներացրած ցանկացած նկար կեղծ է, նույնիսկ եթե մարդու դեմքը լիովին ճիշտ է պատկերված։

Կորստի ֆունկցիան սահմանենք հետևյալ կերպ։

$$\min_G \max_D (E_{x\sim p_{data}}[\log D(x)] + E_{z\sim p_z}[log(1 - D(G(z)))])$$

Մինչ այս անցած կորստի ֆունկցիաներում մենք միայն ունեինք մինիմիզացնելու հատված (descending gradient), բայց կորստում ունենք հատված, որը հարկավոր է մաքսիմիզացնել (ascending gradient)։ Ինչ է նշանակում հետևյալ կորուստը պետք է մաքսիմիզացնել Discriminator-ի նկատմամբ։ D(x)-ը մեզ վերադարձնում է հավանականություն, թե ինչ հավանականությամբ է նկարը կեղծ։ Հավանականության միջակայքը 0-ից 1 է։ Երբ լոգարիթմի մեջ գրված արժեքը փոփոխվում է 0-ից 1 միջակայքում, լոգարիթմի մեծագույն արժեքը կլինի 0, երբ իր մեջ գրվածը հավասար լինի 1-ի։ Առաջին գումարելին մեծագույն արժեքը կընդունի, երբ D(x)=1։ x$\sim p_{data}$, այսինքն x-ը իրական նկարն է, որին Discriminator-ը պետք է ասի իրական՝ 1։ Երկրորդ գումարելին ըստ Discriminator-ի նույնպես անհրաժեշտ է մեծացնել, այսինքն 1-D(G(z)) = 1 $\rightarrow$ D(G(z)) = 0։ Սա նշանակում է, որ Generator-ի գեներացրած նկարին, Discriminator-ը պետք է ասի կեղծ։ Այսինքն այս կորուստը մաքսիմիզացնելով բավարարվում են բոլոր պայմանները Discriminator-ի համար (կեղծ նկարին ասում է կեղծ, իրականին ասում է իրական)։ E-ն մաթսպասումմ է և նշանակում է, որ մենք ցանկանում ենք մեր Discriminator-ը աշխատի ոչ միայն մեր ունեցած նկարների համար, այլ նաև բոլոր գոյություն ունեցող նկարների համար։ Հիմա հասկանանք մինիմիզացման հատվածը ըստ Generator-ի։ Առաջին գումարելիի մեջ Generator-ը չի մասնակցում։ Երկրորդ գումարելին իր փոքրագույն արժեքը կընդունի, երբ ստանանք log 0-ին մոտ արժեք։ Այսինքն ցանկանում ենք 1-D(G(z))-ը մոտեցնել 0-ի $\rightarrow$ D(G(z))-ը մոտեցնել 1-ի։ Սա նշանակում է, որ ցանկանում ենք Discriminator-ը Generator-ի գեներացրած նկարներին նույնպես ասի իրական՝ 1։ Այսպիսով կորուստը մինիմիզացնելով Generator-ը գեներացնում է այնպիսի նկարներ, որ կարողանում է խաբել Discriminator-ին, իսկ կորուստը մաքսիմիզացնելով Discriminator-ը կարողանում է տարբերակել կեղծ և իրական նկարները։
Հիմա դիտարկենք ուսուցման ալգորիթմի քայլերի pseudocode-ը։

for ուսուցման քայլերի քանակ **do**
  for k քայլ **do**

- Ընտրել m հատ աղմուկ վեկտորներ p_z-ից և կազմել փունջ (batch) $\{z^{(1)}, \dots, z^{(m)}\}$
- Ընտրել m հատ իրական նկարներ p_{data}-ից և կազմել փունջ $\{x^{(1)}, \dots, x^{(m)}\}$
- Թարմացնել Discriminator-ը օգտագործելով գրադիենտային բարձրացման (ascending gradient) մեթոդը

$$\frac{1}{m}\sum_{i=1}^{m}[logD(x^{(i)}) + log(1 - D(G(z^{(i)})))]$$

  end for

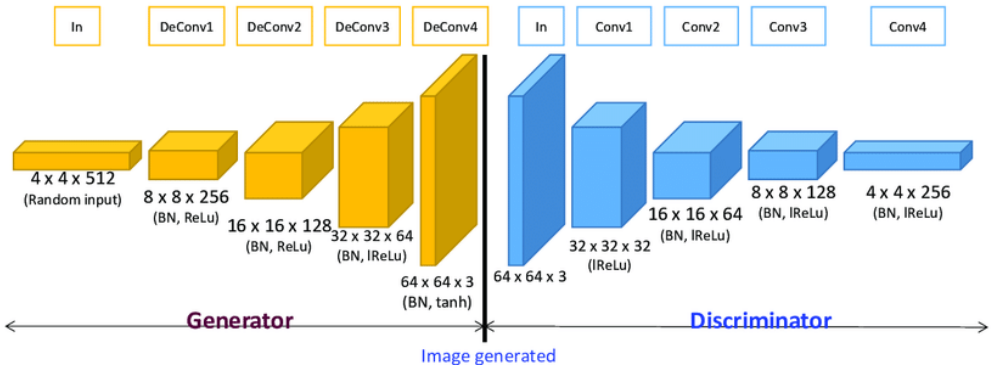
- Ընտրել m հատ աղմուկ վեկտորներ p_z-ից և կազմել փունջ (batch) $\{z^{(1)}, \dots, z^{(m)}\}$
- Թարմացնել Generator-ը օգտագործելով գրադիենտային վայրէջքի մեթոդը

$$\frac{1}{m}\sum_{i=1}^{m} log(1 - D(G(z^{(i)})))$$

end for

k-ն հիպերպարամետր է, որը ցույց է տալիս, թե քանի անգամ ենք թարմացնում Dsicriminator-ը։ Իսկ ինչի՞ համար ենք k քայլ թարմացնում Discriminator-ը, իսկ Generator-ը ընդամենը մեկ քայլ։ Դա արվում է այն պատճառով, որպեսզի Discriminator-ը ավելի ուժեղ լինի և կարողանա Generator-ի գեներացրած կեղծ նկարները տարբերակել։ Մեզ անհրաժեշտ է Generator-ի կեղծ նկարները տարբերակել և դրա շնորհիվ թարմացնել կշիռները։ Իսկ եթե Discriminator-ը թույլ եղավ և միշտ նկարներին ասեց իրական, այդ դեպքում Generator-ը չի սովորի լավ կեղծ նկարներ գեներացնել։

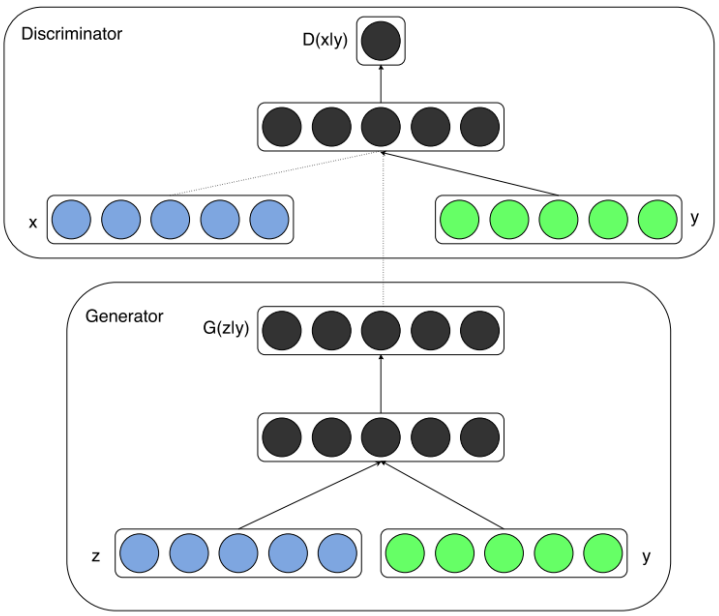
2. Երկրորդ տեսակը Deep Convolutional GAN-երն են (Նկար 2)



Նկար 2: DCGAN ցանցի օրինակ

Discriminator-ի output-ը կարող ենք փոխանցել dense շերտի, որը կասի նկարը կեղծ է, թե ոչ: Այս կառուցվածքը շատ նման է Autoencoder-ի Encoder և Decoder կառուցվածքին: Ի տարբերություն Autoencoder-ի այս կառուցվածքը սիմետրիկ չէ: Այսինքն Discriminator-ի երկրորդ շերտում feature map-ի խորությունը 32 է, իսկ Generator-ի նախավերջին շերտում՝ 64: DeConv գործողությունը նույն Transposed convolution գործողությունն է:

3. GAN-երը օգտագործելիս մեզ միայն անհրաժեշտ է լինելու օգտագործել Generator-ի հատվածը: Այսինքն մուտքում ցանցին փոխանցելու ենք աղմուկ վեկտոր և այն մեզ վերադարձնելու է գեներացված նկարը: Այժմ ունենք խնդիր, երբ մեզ անհրաժեշտ է գեներացնել դեմքի նկարներ: Ինչ որ ձև հարկավոր է ցանցին փոխանցել, որ ցանկանում ենք գեներացնել տղամարդու դեմքի նկար կամ աղջկա դեմքի նկար: Այս խնդիրը կարողանում են լուծել Conditional GAN-երը (Նկար 3):



Նկար 3: CGAN ցանցի օրինակ

Ցանցին փոխանցենք ոչ միայն աղմուկ վեկտորը, այլ նաև ինֆորմացիա, թե ինչ նկար ենք ուզում գեներացնել: Օրինակ դեմք գեներացնելիս փոխանցենք [1, 0] (y) վեկտորը, երբ տղամարդու դեմք ենք ցանկանում գեներացնել և [0, 1] վեկտորը, երբ աղջկա դեմքի

նկար ենք ցանկանում գեներացնել: Օգտագործումը պարզ է, մնում է հասկանալ, թե ինչպես ենք ուսուցանելու ցանցը: Ահա ցանցի կորստի ֆունկցիան:

$$\min_G \max_D (E_{x\sim p_{data}}[\log D(x|y)] + E_{z\sim p_z}[log(1 - D(G(z|y)))])$$

Կարող ենք տեսնել, որ կորստի ֆունկցիան տարբերվում է միայն y պայմանով: Generator-ը և Discriminator-ը մուտքում ստանում են նկար և y վեկտոր: Հիմա հասկանանք, թե ինչ դեպքերում, Discriminator-ը ինչ output պետք է վերադարձնի: Ֆիքսենք, երբ $output = 1$՝ իրական նկար, $output = 0$՝ կեղծ նկար, $y = [1, 0]$՝ տղամարդու դեմք, $y = [0, 1]$՝ աղջկա դեմք:

$$G(\text{Մուտքային տվյալներ}) \to output$$
$$x = \text{տղամարդու նկար},\ y = [1, 0] \to 1$$
$$x = \text{տղամարդու նկար},\ y = [0, 1] \to 0$$
$$x = \text{աղջկա նկար},\ y = [1, 0] \to 0$$
$$x = \text{աղջկա նկար},\ y = [0, 1] \to 1$$

Նույնիսկ եթե Generator-ը գեներացնի շատ լավ տղամարդու դեմք, բայց $y = [0, 1]$, այս դեպքում նույնպես Discriminator-ը պետք է դասակարգի, որպես կեղծ նկար:

Այսպիսով GAN-երի միջոցով, մուտքում փոխանցելով աղմուկ վեկտոր, կարող ենք գեներացնել նկարներ: Ընդ որում օգտագործելով CGAN, կարող ենք գեներացնել մեր ուզած նկարները: