Բովանդակություն

Ներածություն

Թաքնագրությունը[1] գաղտնի տեղեկատվությունը ոչ գաղտնի տեղեկատվության (կոնտեյներ, կամ կրիչ) մեջ թաքցման մեթոդների հավաքածու է: Իսկ թաքնավերլուծությունը[2] (Steganalysis), մի գործընթաց է, որն ուղղված է պարզելուն, թե արդյո՞ք հաղորդագրությունը պարունակում է թաքնված ինֆորմացիա, և հնարավորության դեպքում վերականգնել այն: Թաքնված ինֆորմացիայի ներկայությունը հայտնաբերելու համար սովորաբար օգտագործվում է երկուական դասակարգիչ (Binary classifier): Սույն ուսումնասիրության մեջ ներկայացվելու է մի մոդել, որը ստեղծում է նկար-կրիչներ, հիմնված՝ խորը փաթույթային ստեղծարար մրցակցող ցանցերի (Deep Convolutional Generative Adversarial Networks, կրճատ՝ DCGAN)[3,4] վրա: Այս մոտեցումը թույլ է տալիս ստեղծել ավելի թաքնակայուն կրիչ, ներդրված հաղորդագրությամբ, օգտագործելով ստանդարտ թաքնագրային ալգորիթմներ:

Այս թեմայի շուրջ 2016թ.-ին կատարվել է հետազոտություն[5], որի ընթացքում փորձել են գեներացնել մարդկանց դեմքեր: Մոդելը հաջողությամբ մոլորեցրել է թաքնագրային վերլուծիչին, սակայն որոշ դեպքերում մարդու աչքը գեներացված նկարները հեշտությամբ կարող էր տարբերել իրականից, քանզի մոդելին՝ ուսուցման ժամանակ, տրամադրվել էին տարբեր սեռի մարդկանց դեմքեր, սակայն չէին հաշվի առել այդ հանգամանքը:

Ուսուցմանը մասնակցելու են միանգամից 3 մոդել: Դրանք են՝

1. Գեներացնող մոդել (Գեներատոր - Generator) - G
2. Տարբերակող մոդել (Տարբերակիչ - Discriminator) - D
3. Թաքնավերլուծող մոդել (Թաքնավերլուծիչ - Steganalyser) - S

Առաջին մոդելը՝ գեներատորը, պատասխանատու է նկարներ գեներացնելու համար, այն պետք է այնպիսի նկարներ գեներացնի, որ հնարավոր չլինի տարբերել իրական նկարներից: Այս խնդրի լուծման համար օգտագործվելու է երկրորդ մոդելը՝

տարբերակիչը, որի խնդիրն է լինելու տարբերել իրական նկարը կեղծից (կեղծ են բոլոր այն նկարները որոնք ստեղծել է G գեներատորը)։ Այս ամենից հետո գործի է անցնում 3-րդ մոդելը՝ վերլուծիչը, որի խնդիրն է պարզել արդյո՞ք տրված նկարում առկա է թաքնագրված ինֆորմացիա, թե՞ ոչ։ D վերլուծիչին ուսուցման ընթացքում տրամադրվելու են գեներատորի նկարները, որոնք արդեն պարունակում են թաքնագրված ինֆորմացիա, ինչպես նաև սովորական նկարներ, որոնք չեն պարունակում թաքնագրված ինֆորմացիա։

Այսպիսով D տարբերակիչն ու S վերլուծիչը բարելավելու են իրենց արդյունքը՝ հիմնվելով G գեներատորի տրամադրած և սովորական նկարների վրա, իսկ G-ն բարելավելու է իր արդյունքը՝ հիմնվելով D-ի և S-ի արդյունքի վրա։ Հենց այստեղ էլ առաջ է գալիս մրցակցող ցանցերի գաղափարը, քանզի ստացվում է, որ ցանցերը մրցում են միմյանց հետ, թե ում արդյունքն ավելի լավը կլինի։

Վերջերս մշակված մրցակցող ցանցերը[3] հզոր գեներացնող մոդելներ են, որոնց հիմնական գաղափարը գեներատորի և տարբերակիչի ուսուցումն է մինիմաքս ալգորիթմի[6] միջոցով։ G մոդելը մուտքին ստանում է պատահական՝ այսպես ասած անիմաստ նկար, որի հիման վրա փորձում է ստեղծել իրականին հնարավորինս մոտ պատկեր, իսկ D-ն ձգտում է տարբերակել իրական պատկերները կեղծերից։

Գոյություն ունեն նմանատիպ ցանցերի տարբեր ձևափոխություններ՝

- Խորը փաթույթային ստեղծարար մրցակցող ցանցեր[4]
  - այս մոդելը ստեղծարար մրցակցող ցանցի (GAN) փոփոխություն է, որը մասնագիտացված է պատկերների առաջացման ուղղությամբ
- Պայմանական մրցակցող ցանցեր[7]
  - թույլ է տալիս ստեղծել որևէ դասի օբյեկտներ
- Պատկերների առաջացում՝ հիմնված տեքստային նկարագրության վրա[8]։

Թաքնագրվող գաղտնի ինֆորմացիան, ինչպես նաև կրիչը, կարող է ներկայացված լինել տարբեր տեսքով՝ նկարի, տեքստի, տեսահոլովակի, ձայնագրության և այլն։ Այս

ուսումնասիրության մեջ կատարվելու է տեքստի թաքնագրում նկարում և օգտագործվելու է DCGAN տեսակը։

Գլուխ 1. Գրականության վերլուծական ակնարկ

1.1 Մեքենայական ուսուցում

Նախքան անցնելը բուն թեմային, ծանոթանանք մեքենայական ուսուցման (Machine Learning[Error! Reference source not found.]) հետ: Արթուր Սամուելն այն նկարագրում է այսպես «մեքենայական ուսուցումը մի տեխնոլոգիա է, որը համակարգիչներին հնարավորություն է տալիս սովորելու, առանց բացահայտ ծրագրավորված լինելու»:

Մեքենայական ուսուցման խնդիրներից են.

- Վերահսկվող ուսուցում (Supervised learning)
- Չվերահսկվող ուսուցում (Unsupervised learning)
  - Մասնավոր դեպք է խորհրդատու համակարգը (Recommender system)
- Ուսուցում ամրապնդմամբ (Reinforcement learning)

Վերահսկվող ուսուցման դեպքում մեքենային տրվում է մուտքային տվյալների հավաքածու և այդ տվյալներին համապատասխան ելքային արժեքները: Այսպիսով այս ուսուցման դեպքում մեքենային հայտնի են ամեն մի մուտքային ինֆորմացիային համապատասխանող ելքային արժեքը կամ արժեքները:

1.1.1 Վերահսկվող ուսուցում

Վերահսկվող ուսուցման (Supervised Learning) խնդիրները դասակարգվում են հետևյալ 2 տիպերի՝ ռեգրսիայի խնդիրներ (Regression problems) և դասակարգման խնդիրներ (Classification problems):

Ռեգրեսիայի խնդրներում փորձում ենք կանխատեսել անընդհատ ֆունկցիայի արժեքներ, ինչը նշանակում է, որ մենք փորձում ենք մուտքային փոփոխականները համապատասխանեցնել ինչ-որ անընդհատ ֆունկցիայի ելքային արժեքներին: Դասակարգման հարցում մենք փոխարենը փորձում ենք կանխատեսել ընդհատ ելքային արժեքներ:

1.1.2 Չվերահսկվող ուսուցում

Չվերահսկվող ուսուցումը (Unsupervised Learning) հնարավորություն է տալիս լուծել այնպիսի խնդիրներ, որոնց ելքային արժեքների մասին կա՛մ քիչ ինֆորմացիա ունենք, կա՛մ ընդհանրապես չգիտենք, թե ինչ տեսքի պետք է լինեն: Մենք կարող ենք ստանալ մի այնպիսի ելքային տվյալի կառուցվածք, որի վրա մուտքային տվյալի ազդեցությունն անգամ չգիտենք: Այդ կառուցվածքը հնարավոր է ստանալ տվյալները համախմբելու արդյունքում՝ հիմնված մուտքային տվյալի փոփոխականների միջև կապերի վրա:

1.1.3 Որոշ նշանակումներ

Կատարենք մի քանի նշանակումներ, որոնք կօգտագործվեն հետագայում: $X_1, X_2, \dots X_n$-ով կնշանակենք մուտքային պարամետրերը, Y-ով՝ ելքայինները: $Input^{(i)}{}_1, Input^{(i)}{}_2, \dots Input^{(i)}{}_n$-ը մուտքային պարամետրերի արժեքներն են (տվյալի հատկություններ), իսկ $Output^{(i)}$-ն՝ ելքային պարամետրի արժեքն է, որտեղ՝ $i=1,2,\dots,m$: Հարմարավետության համար $Input^{(i)}{}_1, Input^{(i)}{}_2, \dots Input^{(i)}{}_n$-ը նշանակենք $x^{(i)}$-ով, իսկ $Output^{(i)}$-ն՝ $y^{(i)}$-ով, n-ը մուտքային պարամետրերի քանակն է: $(x^{(i)}, y^{(i)})$ զույգն կանվանենք ուսուցման օրինակ (training example), իսկ դրանց ցուցակը՝ ուսուցման տվյալներ (training set): Այսինքն m-ը՝ ուսուցման տվյալների քանակն է:

Կարող ենք ասել, որ «Վերահսկվող ուսուցման նպատակն է՝ տրված ուսուցման տվյալների հիման վրա ձևավորել մի այնպիսի $h : X \rightarrow Y$ հիպոթեզ ֆունկցիա, որ $h(x)$-ի ելքային արժեքը բավարար մոտ լինի համապատասխան y-ի արժեքին»: Ինչքան $h(x)$-ի արժեքը մոտ լինի համապատասխան y-ի արժեքին, այնքան ավելի ճիշտ արդյունքներ կտա մեր մեքենայական ուսուցման մոդելը: $\theta_0, \theta_1, \dots \theta_n$-ով նշանակենք $h(x)$-ի գործակիցները (որոշ դեպքերում $h(x)$-ը կնշանակենք $h_\theta(x)$): Մեքենայական ուսուցման խնդիրը հենց այդ θ-ների արժեքները գտնելու մեջ է կայանում, քանզի, հետագայում՝ երբ արդեն մեր մոդելը բավարար չափով ուսուցանված կլինի, նրան տրվելու են $X_1, X_2, \dots X_n$ արժեքները և քանզի այն ունի արդեն հաշվարկած $\theta_0, \theta_1, \dots \theta_n$ արժեքները, ընդամենը պետք է հաշվի $h_\theta(x)$-ի արժեքը:

1.2 Ուսուցման տարրեր

1.2.1 Արժեքի ֆունկցիա

h(x)-ի արժեքների ճշտությունը կարելի է գնահատել **արժեքի ֆունկցիայի (Cost Function)** միջոցով: Այն իրենից ներկայացնում է *h(x)-ի* բոլոր ելքային արժեքների և իրական *y-ների* արժեքների միջինացված տարբերություն (Բձ. 1):

$$J(\theta_0, \theta_1, \dots \theta_n) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_i) - y_i)^2 \qquad (1)$$

Այս ֆունկցիան նաև կոչվում է քառակուսային սխալի ֆունկցիա (Squared error function): Քառակուսային միջինը բաժանվել է 2-ի՝ հետագա հաշվարկների հարմարավետության համար, քանի որ դրա միջոցով $(h_\theta(x_i) - y_i)^2$-ի ածանցումից ստացված քառակուսի աստիճանը կվերանա:

Ստացվեց, որ մեր խնդիրը կայանում է *J(θ₀, θ₁, … θₙ)* - ը մինիմիզացնելու մեջ, որի ֆորմալ տեսքը ներկայացված է բանաձև 2-ում

$$\underset{\theta_0, \theta_1, \dots \theta_n}{minimize} J(\theta_0, \theta_1, \dots \theta_n) \qquad (2)$$

1.2.2 Նվազող գրադիենտ

Այսպիսով արդեն պարզաբանվեց, թե ինչ է հիպոթեզ ֆունկցիան և թե ինչպես կարելի է չափել նրա ճշտությունը: Այժմ անհրաժեշտ է որոշել հիպոթեզի պարամետրերը:

Քանզի արժեքի ֆունկցիան հիմնականում իրենից ներկայացնում է բարդ մաթեմատիկական բանաձև, այն դժվար է գծել, կամ գտնել, թե *θ-ի* որ արժեքների դեպքում է այն ընդունում մինիմալ արժեք: Հենց այս խնդիրը լուծելու համար օգտագործվում է նվազող գրադիենտը (Gradient Descent):

Կամայական ֆունկցիայի ածանցյալը ցույց է տալիս տվյալ կետում շոշափողի ուղղությունը, հետևաբար ամեն քայլին շարժվելով այն ուղղությամբ, որն ամենաշատն է նվազեցնում արժեքի ֆունկցիան ի վերջո կհասնենք որևէ մինիմում արժեքի: Յուրաքանչյուր քայլի չափը որոշվում է α պարամետրի միջոցով, որը կոչվում է ուսուցման

գործակից (learning rate): Քայլի ուղղությունը, որոշվում է $J(\theta_0, \theta_1, \dots \theta_n)$-ի մասնակի ածանցյալով: Կախված այն բանից, թե որտեղից ենք սկսում դիտարկել գրաֆիկը, հնարավոր է տարբեր մինիմումների հասնել:

Ընդհանուր դեպքի համար նվազող գրադիենտի ալգորիթմը կլինի. կրկնել հետևյալը մինչև զուգամիտում՝ $\theta_j := \theta_j - \alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1, \dots \theta_n)$, որտեղ՝ $j = 0,1, \dots n$ ներկայացնում է հատկության հերթական համարը: Այն անվանում են նաև թարմացման կանոն (update rule)*:* Յուրաքանչյուր իտերացիային պետք է միաժամանակ թարմացնել բոլոր $\theta_0, \theta_1, \dots \theta_n$ պարամետրերը:

Պետք է հաշվի առնել, որ կարևոր է α-ի ճիշտ ընտրությունը, քանզի դրանով է պայմանավորված ալգորիթմի զուգամիտման ժամանակը: Եթե ալգորիթմը չի զուգամիտում կամ շատ ժամանակ է պահանջում մինիմումին հասնելու համար ապա α քայլաչափը սխալ է ընտրված: α-ն հաստատուն պահելու դեպքում $\alpha \frac{\delta}{\delta \theta_j} J(\theta_0, \theta_1, \dots \theta_n)$ արտադրյալը ամեն քայլին կնվազի և հասնելով որևէ մինիմումի այն կհավասարվի *0*-ի (իրականում 0-ի չի հավասարվի, այլ կմոտենա ինչ-որ շատ փոքր թվի, որը մեր խնդրի համար համարվում է բավարար) և հետագա քայլերը ոչ մի կերպով չեն ազդի *θ*-ների արժեքների վրա: Հեշտությամբ կարելի է համոզվել, որ, եթե մեր հիպոթեզն ունի գծային տեսք՝

$$h_\theta(x) = \theta_0 + \theta_1 X_1 + \theta_2 X_2 + \cdots + \theta_n X_n \qquad (3)$$

ապա թարմացման կանոնի մեջ *J(θ)*-ի արժեքը տեղադրելուց հետո թարմացման կանոնի տեսքը կլինի՝

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right) \cdot x_j^{(i)}, \qquad (4)$$

որտեղ՝ j:=0…n: Այստեղ և հետագայում կընդունենք, որ $x_0^{(i)} = 1$, բոլոր *i*-երի համար: Սա արվում է բանաձևերը հարմար ներկայացնելու համար:

1.2.3 Ուսուցման գործակից

Նվազող գրադիենտն իրականացնելուց հետո անհրաժեշտ է հետևել ալգորիթմի աշխատանքին (մոդելի ուսուցման պրոցեսին) և հասկանալ արդյո՞ք այն ճիշտ է աշխատում: Պատկերացում կազմելու համար, թե ինչքան լավ է սովորում մոդելը, անհրաժեշտ է գծել արժեքի ֆունկցիայի՝ *J(θ)-ի,* կախումը *իտերացիաների քանակից:* Եթե ամեն ինչ ճիշտ է աշխատում, ապա ամեն իտերացիայից հետո *J(θ)-ի* արժեքը պետք է նվազի՝ ձգտելով *0-ի:* Հետևաբար, եթե գրաֆիկը աճում է, ապա ինչ որ բան այն չէ: Հիմնականում դրա պատճառը *α-ի* մեծ արժեքն է լինում: Հարկ է նշել՝ ապացուցված է, որ, եթե ուսուցման գործակից (Learning Rate) *α-ն* բավարար չափով փոքր է ընտրված, ապա *J(θ)-ն* նվազում է ամեն իտերացիայից: Սակայն, եթե այն շատ փոքր է ընտրված, ապա *J(θ)-ն* կարող է շատ դանդաղ նվազել:

Կարելի է համարել որ մոդելը բավարար չափով ուսուցանվել է միայն, երբ *J(θ)-ի* փոփոխությունն ինչ-որ իտերացիայից հետո փոքր է որևէ *E* արժեքից: *E-ն* կամայապես ընտրված փոքր թիվ է, օրինակ՝ 10^{-3}: Գործնականում դժվար է ընտրել *E-ի* օպտիմալ արժեք:

1.2.4 Մուտքային տվյալի հատկության մասշտաբավորում

Մենք կարող ենք արագացնել նվազող գրադիենտի աշխատանքը՝ բերելով բոլոր մուտքային պարամետրերը մոտավորապես նույն տիրույթի թվերի: Դա կապված է այն բանի հետ, որ որ *θ-ն* ավելի արագ է հասնում մինիմումին փոքր միջակայքերում և ավելի դանդաղ՝ մեծ միջակայքերում, հետևաբար այն տատանվելով է այն տատանվելով է ձգտում մինիմումին, երբ փոփոխականները շատ անհավասար են: Դա կանխելու համար կարող ենք այնպես փոփոխել հատկությունները, որ նրանք ընկնեն մոտավորապես միևնույն թվային տիրույթ: Իդեալական դեպքում՝ $-1 < x_i < 1$ կամ՝ $-0.5 < x_i < 0.5$: Եթե չկատարվի հատկությունների մասշտաբավորում, ապա որոշ դեպքերում հնարավոր է, որ ալգորիթմը երբեք չզուգամիտի:

Հատկության մասշտաբավորումն[Error! Reference source not found.] (Feature Scaling) ու միջինով նորմալացումը (Mean Normalization) այն երկու մեթոդներն են, որոնք կօգնեն լուծել այդ

խնդիրը: Առաջինը ենթադրում է մուտքային տվյալների բաժանում նրանց մեծագույն և փոքրագույն արժեքների տարբերության վրա: Միջինով նորմալացման դեպքում պետք է մուտքային փոփոխականից հանել մուտքային տվյալների միջին արժեքը, ապա նոր բաժանել մեծագույն և փոքրագույն արժեքների տարբերության վրա: Այս երկու մեթոդների իրականացման համար անհրաժեշտ է փոփոխել մուտքային պարամետրերը՝ համապատասխան ներքնի բանաձևի.

$$x_i := \frac{x_i - \mu_i}{s_i}, \tag{5}$$

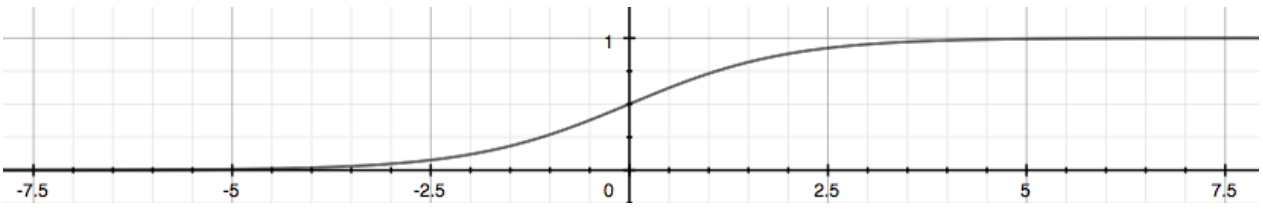
որտեղ s_i-ն i-րդ հատկության մեծագույն և փոքրագույն արժեքների տարբերությունն է, իսկ μ_i-ն՝ այդ հատկության բոլոր արժեքների միջինը:

1.3 Դասակարգում

Երկուական դասակարգման խնդիր (binary classification problem), որտեղ *y*-ը կարող է ընդունել միայն 2 արժեք՝ *0* և *1,* լուծելու համար համար կձևափոխենք $h_\theta(x)$-ն այնպես, որ այն բավարարի $0 \leq h_\theta(x) \leq 1$ պայմանին: Դա անելու համար կարելի է լոգիստիկ ֆունկցիային (Logistic Function) փոխանցել $\theta^T x$-ը.

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \tag{6}$$

Այստեղ *g*-ն հենց այն լոգիստիկ ֆունկցիան է, որը կամայական իրական թիվ համապատասխանեցնում է (0, 1) տիրույթի որևէ թվի, ինչը թույլ է տալիս կամայական տիրույթի ելքային արժեքներ ունեցող ֆունկցիան փոխակերպել դասակարգման խնդրին ավելի հարմար ֆունկցիայի: Լոգիստիկ ֆունկցիան նաև անվանում են Սիգմոիդ ֆունկցիա (Sigmoid Function): Այսպիսով $h_\theta(x)$-ը ելքային արժեքի 1 լինելու հավանականությունն է: Նկ. 1-ում պատկերված է այդպիսի ֆունկցիայի մի օրինակ:



Նկ. 1 Սիգմոիդ ֆունկցիայի օրինակ

1.3.1 Լոգիստիկ հիպոթեզի արժեքի ֆունկցիան

Ընդհանուր դեպքում արժեքի ֆունկցիան ունի հետևյալ տեսքը՝

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\theta(x_i), y_i), \tag{7}$$

որտեղ *Cost*-ը այն ֆունկցիան է, որը հաշվում է արժեքը *i*-րդ ուսուցման օրինակի համար: Լոգիստիկ ֆունկցիայի համար կարելի է օգտագործել հետևյալ ֆունկցիան՝

$$\begin{cases} Cost(h_\theta(x), y) = -\log\big(h_\theta(x)\big) & \text{երբ՝ } y = 1 \\ Cost(h_\theta(x), y) = -\log\big(1 - h_\theta(x)\big) & \text{երբ՝ } y = 0 \end{cases} \tag{8}$$

Այստեղից երևում է, որ, եթե արժեքի ֆունկցիան գրենք այս ձևով, ապա համոզված կարող ենք ասել, որ *J*-ն ունի ուռուցիկ տեսք լոգիստիկ ռեգրեսիայի համար: Ինչը շատ կարևոր է ավելի արագ ուսուցանվող և ճիշտ արդյունքներ գուշակող մոդել ստեղծելու համար: Այն կարելի է փոխարինել մեկ արտահայտությամբ հետևյալ կերպ՝

$$Cost(h_\theta(x), y) = -y\log\big(h_\theta(x)\big) - (1-y)\log\big(1 - h_\theta(x)\big) \tag{9}$$

Հետևաբար *J*-ն կունենա հետևյալ տեսքը՝

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[-y^{(i)}\log\left(h_\theta\left(x^{(i)}\right)\right) - \left(1-y^{(i)}\right)\log\left(1 - h_\theta\left(x^{(i)}\right)\right)\right] \tag{10}$$

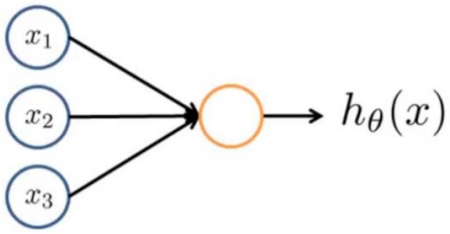
1.4 Նեյրոնային ցանցեր

Հասկանալու համար, թե ինչ անհրաժեշտություն կա ուսումնասիրել այլ ուսուցման ալգորիթմ՝ նեյրոնային ցանցեր[Error! Reference source not found.] (Neural Networks), պատկերացնենք մի դեպք, երբ դասակարգման խնդիր լուծելիս հատկությունները բավարար չեն ճշգրիտ մոդել ուսուցանելու համար, և անհրաժեշտություն է առաջացել ավելացնել նոր՝ քառակուսային, խորանարդային կամ այլ,հատկություններ: Այս դեպքում եթե ավելացնենք բոլոր քառակուսային հատկությունները՝

$$\begin{gathered} x_1^2, x_1x_2, x_1x_3, \dots x_1x_n \\ x_2^2, x_2x_3, \dots x_2x_n \\ \dots \\ x_{n-1}^2, x_{n-1}x_n \\ x_n^2 \end{gathered} \tag{11}$$

ապա կստանանք $\frac{n\cdot(n+1)}{2}+n$ քանակի հատկություն: Այսինքն ստացվում է, որ նոր հատկությունների քանակը նախկինից մոտավորապես քառակուսային $\left(\approx \frac{n^2}{2}\right)$ կախում ունի: Նույն ձևով խորանարդային հատկություններ ավելացնելիս կարելի է համոզվել որ կախումը խորանարդային է: Սա կարող է բերել գերհամապատասխանեցման (overfitting) խնդրին ինչպես նաև բավականաչափ մեծ հաշվողական ռեսուրսներ կպահանջվեն նման մեծ թվով հատկությունների հետ աշխատելու համար:

Մեքենայական ուսուցման մեջ օգտագործվող նեյրոնի մոդելը հնարավորինս մոտ է արված մարդու ուղեղի նեյրոնային կառուցվածքին: Յուրաքանչյուր նեյրոն ստանում է մուտքին որևէ պարամետրեր, կատարում է որոշակի հաշվարկներ, և արդյունների հիման վրա որոշում է թե ինչ ազդանշան ուղարկի հաջորդ նեյրոնին:

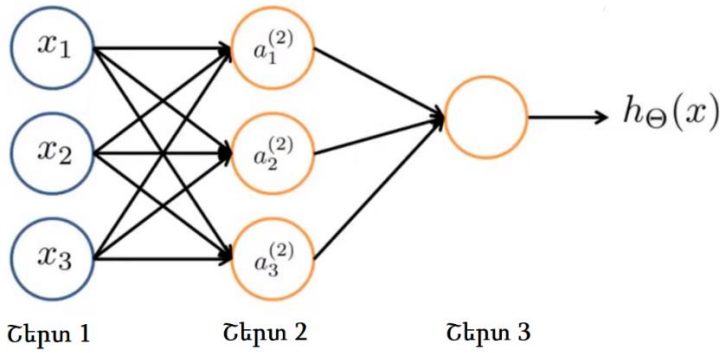


Նկ. 2 Նեյրոնի մոդելի օրինակ

Նկ. 2-ում պատկերված է նեյրոնի պարզեցված մոդելը, որն օգտագործվում է մեքենայական ուսուցման մեջ: x_1, x_2, x_3-ը մուտքային պարամետրերն են, իսկ դեղինով եզրագծվածը նեյրոնի «մարմինն» է: Այստեղ նույնպես կարող ենք ավելացնել $x_0 = 1$ պարամետրը, որը կոչվում է շեղում (bias): Նեյրոնի կատարած հաշվարկների արդյունքը հիպոթեզ ֆունկցիայի արժեքն է,որի բանաձևը լոգիստիկ ռեգրեսիայի բանաձևն է: Նեյրոնի հիպոթեզի ֆունկցիան այլ կերպ անվանում են նաև սիգմոիդ ակտիվացման ֆունկցիա: Նեյրոնի ակտիվացիան դա լոկ այն արժեքն է,որը հաշվարկում է այդ նեյրոնը,այլ կերպ ասած նրա ելքում ստացված արժեքն է: Բնականաբար այդ ֆունկցիան,ինչպես և նախորդ մեր դիտարկած սիգմոիդ ֆունկցիան ունի իր պարամետրերը՝ $\theta_0, \theta_1, \ldots \theta_n$ (մեր օրինակի դեպքում n = 3), որոնց անվանում են կշիռներ (weights):

Նեյրոնային ցանցը, ինչպես բխում է անունից, բազմաթիվ նեյրոններից բաղկացած ցանց է, որոնց ելքերն ու մուտքերը կապված են միմյանց հետ: Նկ. 3-ում պատկերված է նեյրոնային ցանցի մի պարզ օրինակ:

Ցանցը բաժանվում է շերտերի (layers): Առաջին շերտը անվանում են մուտքային շերտ, քանի որ սա այն շերտն է որտեղ ցանցի մուտքին տրվում են հատկությունները: Վերջին շերտը անվանում են ելքային շերտ, այն հաշվարկում է հիպոթեզ ֆունկցիայի վերջնական արժեքը: Առաջին և վերջին շերտերի միջև ընկած բոլոր մնացած շերտերն անվանում են թաքնված շերտեր: Վերջիններս թաքնված են, քանի որ ուսուցման ընթացքում նրանց արժեքներին չենք հետևում: $a_i^{(j)}$-ով նշանակված է j-րդ շերտի i-րդ նեյրոնի ակտիվացիան», իսկ $\theta^{(j)}$-ով կնշանակենք կշիռների այն մատրիցը, որը պարունակում է j-ից (j+1) շերտ անցնելու բոլոր ակտիվացիաների ֆունկցիաների պարամետրերը:



Նկ. 3 Նեյրոնային ցանցի պարզ օրինակ

Ընդհանուր դեպքում (j+1)-րդ շերտի նեյրոնների ակտիվացման ֆունկցիաների տեսքը բերված է ստորև.

$$a_1^{(j+1)} = g(\theta_{10}^{(j)} x_0 + \theta_{11}^{(j)} x_1 + \ldots + \theta_{1n}^{(j)} x_n) \quad (12)$$

$$a_{s_{j+1}}^{(j+1)} = g(\theta_{s_{j+1}0}^{(j)} x_0 + \theta_{s_{j+1}1}^{(j)} x_1 + \ldots + \theta_{s_{j+1}n}^{(j)} x_n) \quad (13)$$

որտեղ n-ը մուտքային պարամետրերի քանակն է, իսկ s_{j+1}-ը՝ (j+1)-րդ շերտում նեյրոնների քանակը: g-ֆունկցիան արդեն պարզաբանվել է 1.3 բաժնում: Նշված ֆունկցիաների օգնությամբ վերջին շերտի արժեքը հաշվելով կարող ենք ստանալ $h_\theta(x)$-ի արժեքը:

1.5 Խնդրի դրվածքը

Ուսումնասիրելով գրականությանը կարելի է եզրահանգել, որ ավարտական աշխատանքի շրջանակներում դրվում է խնդիր՝ մշակել գեներատիվ մրցակցային ցանցերի միջոցով նկարի տեսքով թաքնագրության կրիչ (կոնտեյներ) ստեղծող համակարգ։

Այդ նպատակով անհրաժեշտ է ուսուցանել միաժամանակ 3 մոդել՝ գեներատոր, տարբերակիչ, թաքնավերլուծիչ։ Այս մոդելները մրցակցելով միմյանց հետ փորձելու են լավորակել նախագծվող համակարգի արդյունքը, որից հետո համոզվելով, որ մոդելները բավարար չափով ուսուցանված են, գեներացնող մոդելը կկարողանա ստեղծել իրական մարդկանց դեմքերին մոտ այնպիսի նկար-կրիչներ, որոնք կապահովեն բարձր թաքնակայունություն։

2016 թվականին կատարված հետազոտությունները լավ արդյունքներ էին տվել, սակայն գեներացված նկարները մոտ չեին իրական նկարներին։ Սույն աշխատության մեջ գեներատորին ուսուցանման ժամանակ տրվելու են ինչ-որ հատկանիշներով (սեռ, տարիք, ռասսա) նման մարդկանց նկարներ, ինչը, ենթադրվում է, որ կհանգեցնի իրականին ավելի մոտ նկարների ստեղծմանը։

Գլուխ 2. Գեներատիվ մրցակցող ցանցերի կիրառումը նկարի տեսքով թաքնագրության կրիչ ստեղծելու համար

2.1 Մրցակցող ցանցեր

2.1.1 Մինիմաքս ալգորիթմ

Մինիմաքսը որոշումներ ընդունելու կանոն է, որն օգտագործվում է արհեստական բանականությունության, որոշումների տեսության, խաղերի տեսության, ստատիստիկայի և փիլիսոփայության մեջ, հնարավոր կորուստը (վատագույն դեպքում՝ մաքսիմալ կորուստը) քչացնելու համար։ Սկզբում կանոնները նախատեսված էին երկու խաղացողից բաղկացած զրոյական գումարով խաղի համար, որտեղ մի խաղացողի հաղթանակը բացառում է մյուսի հաղթանակը։ Հետագայում այն զարգացել է և օգտագործվում է ավելի բարդ խաղերում, ինչպես նաև անորոշության պայմաններում որոշումների ընդունման մեջ։

Խաղացողի մաքսիմալ վաստակած միավորը դա այն ամենամեծ թիվն է, որը խաղացողը կարող է հավաքել, առանց իմանալու հակառակորդների քայլերը։ Համապատասխանաբար այդ միավորը այն ամենափոքր թիվն է, որը հակառակորդները կարող են ստիպել խաղացողին հավաքել, երբ գիտեն նրա քայլերը։ Ասվածը մաթեմատիկորեն կարող ենք ներկայացնել հետևյալ կերպ՝

$$\underline{v_i} = \max_{a_i} \min_{a_{-i}} v_i(a_i, a_{-i}) \tag{14}$$

Որտեղ՝

- i-ն հերթական խաղացողի համարն է
- (-i)-ն բոլոր խաղացողներն են՝ բացառությամբ i-րդի
- a_i-ն i-րդ խաղացողի քայլն է
- a_{-i}-ն բոլոր խաղացողների քայլերն են՝ բացառությամբ i-րդի
- v_i i-րդ խաղացողի միավորների հաշվման (արժեքի) ֆունկցիան է

i-րդ խաղացողի մաքսիմալ միավորի հաշվարկը կատարվում է հաշվի առնելով վատագույն տարբերակը՝ նրա ամեն մի հնարավոր քայլի համար ստուգվում է մնացած խաղացողների հնարավոր բոլոր քայլերը և գտնվում է վատագույն կոմբինացիան, որը խաղացողին կբերի ամենափոքր միավորը։ Հետո պետք է հասկանալ, թե, ինչ քայլ պետք է անի i-րդ խաղացողը, որպեսզի համոզված լինի, որ այս ամենափոքր միավորը դա նրա ամենամեծ հնարավոր միավորն է։ Այսինքն i-րդ խաղացողն իր հերթական քայլով մաքսիմիզացնում է իր միավորը և մինիմիզացնում է հակառակորդների ազդեցությունը իր միավորի վրա։

Այն խաղը որում հնարավոր է կիրառել վերը նշված մոտեցումը, անվանում են «Մինիմաքս խաղ»։

2.1.2 Գեներատիվ մրցակցող ցանցեր

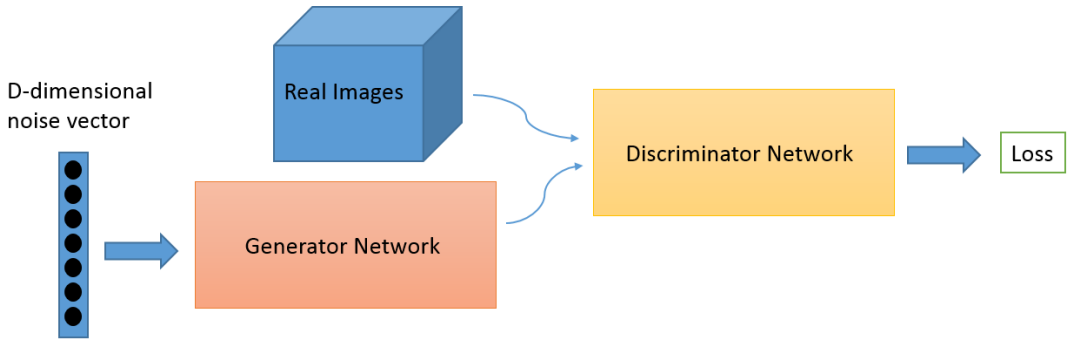
Մրցակցող ցանցերի առաջին հայտնագործողը Գուդֆելոուն էր։ Նա առաջինն էր, ով մտածեց հետևյալ հարցի շուրջ՝ «Ի՞նչ կլինի եթե երկու նեյրոնային ցանցի ստիպենք մրցակցել միմյանց դեմ կամայական տիպի տվյալներ գեներացնելու համար»։ Ընդ որում մրցակցող ցանցերից առաջինը տվյալների գեներատորն է, իսկ երկրորդը՝ տարբերակիչը։ Այստեղ տվյալ ասելով հասկանում ենք ամեն ինչ՝ նկար, երաժշտություն, վիդեո, տեքստ և այլն։ Գեներատորը, ինչպես հետևում է իր անվանումից, տվյալներ գեներացնող մոդելն է, իսկ տարբերակիչը իրական տվյալները գեներացվածներից տարբերակող մոդելն է։ Հետագա փորձերի արդյունքները բավականին լավ արդյունքներ ցույց տվեցին և այն, ինչը հայտնաբերեց Գուդֆելոուն կոչվեց «գեներատիվ մրցակցող ցանցեր» (ԳՄՑ)։ ԳՄՑ-ի միջոցով հնարավոր դարձավ ուսուցանել մի այնպիսի մոդել, որն ունակ է ստեղծել ուսուցման ընթացքում իրեն տրված տիպի տվյալներ։ Օրինակ՝ կենդանիների կամ մեքենաների նկարներ։ Վերջին տարիներին խորը ուսուցումը (Deep Learning) բավականին առաջընտաց է ապրել։ Դրա միջոցով հնարավոր է ստեղծել մոդելներ, որոնք ունակ են ճանաչել տարբեր տեսակի առարկաներ նկարների մեջ, սակայն հնարավոր չէ ստեղծել մի այնպիսի մոդել որը նկարներ կգեներացնի նույնքան լավ, ինչքան ԳՄՑ-երը։

ԳՄՑ-ի միջոցով կարելի է ստանալ նկարների գեներատիվ հզոր մոդելներ, սակայն ստացված մոդելներն ունակ չեն գեներացնել կամայական տիպի տվյալներ: Այդ մոդելները գեներացնում են միայն այնպիսի տվյալներ, ինչի վրա որ կատարվել է ուսուցումը: Այսպիսով, եթե ԳՄՑ-ի ուսուցման ժամանակ տրվել են միայն շան նկարներ, ապա գեներատիվ մոդելը կսովորի գեներացնել միայն շան նկարներ, և ունակ չի լինի գեներացնել բոլորովին այլ տիպի կենդանու նկարներ: Իսկ եթե մուտքային տվյալները բավականին տարբեր բնույթի լինեն, ապա հնարավոր է որ գեներատորի ուսուցումը անհաջող լինի և այն վերջիվերջո ունակ չլինի գեներացնել որևէ իմաստ արտահայտող տվյալներ:

Սույն աշխատությունում ուսուցման ժամանակ G գեներատորի մուտքին տրվելու է որևէ z բաշխումից աղմուկ՝ $p_z(z)$, որից G-ն ստանալու է նոր նկար, այդ ֆունկցիան նշանակենք՝ $G(z;\theta_g)$: Սահմանենք ևս մեկ ֆունկցիա տարբերակիչի համար՝ $D(x;\theta_d)$, որի ելքը մի սկալյար մեծություն է, որն արտահայտում է նրա մուտքին տրված x նկարի իրական լինելու հավանականությունը: Այսինքն նրա ելքը կլինի 0, եթե մուտքին տրված նկարը գեներացված է, և 1՝ հակառակ դեպքում: Ստացվում է, որ D-ն մեզ մոտ երկուական դասակարգիչ է: Ուսուցման ընթացքում մենք փորձում ենք մեծացնել D-ի ճշտությունը, նրա մուտքին տալով իրական և գեներատորի գեներացրած նկարներ: Միևնույն ժամանակ՝ զուգահեռաբար, մենք ուսուցանում ենք G-ն ստիպելով նրան փոքրացնել D-ի ճշտությունը: Այլ կերպ ասած այս երկու մոդելները մրցակցում են միմյանց հետ և խաղում են հետևյալ մինիմաքս խաղը $V(G,D)$ արժեքի ֆունկցիայով՝

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \qquad (15)$$

ԳՄՑ ուսուցանելիս պետք է միշտ հիշել, որ մենք ուսուցանում ենք գեներատորը և տարբերակիչը զուգահեռաբար: Այսինքն ուսուցման մեկ քայլի ընթացքում նեյրոնային ցանցի կշիռները թարմացնում է թե՛ գեներացնող և թե՛ տարբերակող մոդելը: Նկ. 4-ում պատկերված են ուսուցմանը մասնակցող նեյրոնային ցանցերը և նրանց միջև կապերն ու մուտքերը:



Նկ. 4 Գեներատիվ մրցակցող ցանցերի աշխատանքի սխեմա

2.2 Խորը փաթույթային գեներատիվ մրցակցող ցանցեր

Ներկա պահին գոյություն ունեն ԳՄՑ-երի տասնյակ տարբեր տեսակներ: Խորը փաթույթային գեներատիվ մրցակցող ցանցերը (ԽՓԳՄՑ) դրանցից մեկն է: ԽՓԳՄՑ-ն նախատեսված է նկարների գեներացիայի համար: Երկար տարիներ ուշադրության կենտրոնում են եղել վերահսկվող փաթույթային նեյրոնային ցանցերը, մինչդեռ ԽՓԳՄՑ-ն լավ օրինակ է չվերահսկվող փաթույթային նեյրոնային ցանցերի ոչ պակաս արդյունավետության: Ինչպես երևում է անվանումից, ԽՓԳՄՑ-երի հիմքում ընկած է փաթույթային ցանցերի գաղափարը, որոշակի փոփոխություններով, որոնք առաջ են քաշվել վերջերս: Դրանք են՝

1. Նեյրոնային ցանցի բոլոր ոչ-փաթույթային՝ միավորման (pooling), շերտերը փոխարինել փաթույթայինով, ինչը հնարավորություն կտա ցանցին սովորել մշակել սեփական downsampling-ը: Այս աշխատությունում այս մոտեցումը օգտագործվել է նաև գեներատորի մոդելավորման համար, ինչը թույլ է տալիս գեներատորին մշակել իր սեփական upsampling-ը:
2. Վերացնել ամբողջությամբ միացված շերտերը փաթույթային շերտերից առաջ:
3. Անհրաժեշտ է կիրառել խմբային նորմալիզացում (Batch Normalization), ինչը կկայունացնի ուսուցումը: Այն կնորմալիզացնի ամեն նեյրոնի մուտքը՝ բերելով միջին արժեքը զրոյի:
4. Տարբերակիչի բոլոր շերտերի համար օգտագործել բացվածքով ReLU (Leaky ReLU) ֆունկցիան ReLU-ի փոխարեն:

2.3 Թաքնավերլուծություն մեքենայական ուսուցմամբ

Թաքնագրության ժամանակ կոնտեյներում թաքցվում է գաղտնի տվյալը: Կոնտեյներների դասին են պատկանում նաև նկարները: Նկարներում՝ նրա պիկսելային ներկայացման մեջ ինֆորմացիայի թաքնագրման ժամանակ կատարվում է փոփոխություն նկարի տենզորի (Tensor) մեջ, որն ունի NxMxC չափողականություն, որտեղ՝

1. N-ը տողերի քանակն է
2. M-ը սյուների քանակն է
3. C-ը գույների խորությունն է կամ նկարի հոսքերի թիվը

Հաշվի առնելով այն, որ նկարներում հիմնականում օգտագործվում է 8 բիթ կոդավորում, կարելի է հաշվարկել նկարի տենզորի չափը բիթերով՝

$$S = N * M * C * 8 \tag{16}$$

Նկարի տենզորի մեջ թաքցվող ինֆորմացիայի քանակը համեմատական է նրա չափին՝

$$T = K_{\text{նորմ}} * S \tag{17}$$

$K_{\text{նորմ}}$ գործակիցը բնութագրում է նկարի կոնտեքստից, որն իր հերթին իրենից ներկայացնում է տենզորում պիկսելների բաշխման ֆունկցիա: $K_{\text{նորմ}}$-ը խիստ կախվածություն ունի պիկսելների բաշխումից և որպես հետևանք երկու նույն չափի նկարների թաքնագրման տարողունակությունը կարող է խիստ տարբերվել: $K_{\text{նորմ}}$-ն իրենից կրում է զուտ բնութագրական բնույթ և իհարկե հնարավոր է գերազանցել թույլատրելի նորման, սակայն նմանատիպ մոտեցումը կբերի թաքնագրային համակարգի վատթարացմանը և հետագա անվտանգության նվազեցմանը:

Ինչպես արդեն նշվեց $K_{\text{նորմ}}$-ը հանդիսանում է ֆունկցիա պիսկելների բաշխումից՝

$$K_{\text{նորմ}} = K(p) \tag{18}$$

Այստեղ p-ն հանդիսանում է պիկսելների բաշխման ֆունկցիան: K-ն բավականին դժվար է գնահատել և դժվար է այն ներկայացնել անալիտիկ տեսքով, հաշվի առնելով գոյություն ունեցող նկարների տարատեսակը:

Մեքենայական ուսուցման հիմնախնդիրներից է մոտարկել ֆունկցիան, ըստ մուտքային տվյալների: Այս մոտեցումը խոստումնալից է K-ի տեսքի որոնման հարցում, քանի որ կարելի է բավականին ճշգրիտ մոտարկել K ֆունկցիան: Առաջարկվող համակարգում K ֆունկցիայի մոտարկումը պարամետրիզացվում է նեյրոնային ցանցով:

$$K_{\text{նորմ,մոտ}} = K_w(p_{\text{տվյալ}}) \qquad (19)$$

Քանզի խնդիրը կայանում է գեներացնել նկարներ մաքսիմալ $K_{\text{նորմ}}$-ով, այս աշխատությունում ներկայացվող համակարգում ներդրվում է դասակարգիչ, որի հիմնական նպատակն է հասկանալ արդյոք առկա է նկարում թաքնագրված տվյալ, նույնն է թե արդյոք գերազանցվել է $K_{\text{նորմ}}$-ը տվյալ կոնտեյների համար: Քանի որ առաջարկվող մոտեցման մեջ խնդիրը մեծ $K_{\text{նորմ}}$-ով կոնտեյների գեներացումն է, ուսուցման ժամանակ եթե դասակարգիչը հայտանբերում է թաքնագրված տեքստ ապա $K_{\text{նորմ}}$ մոտարկող նեյոնային ցանցը, որը մոտարկումը կատարում է համապատասխան $K_{\text{նորմ}}$-ի համար նկարի գեներացմամբ ենթարկվում է պարամետերերի թարմացամն ըստ գրադիենտային անկման: Այսպիսի մոտեցումը թույլ է տալիս մաքսիմալացնել գեներացվող կոնտեյներների $K_{\text{նորմ}}$-ը և որպես հետևանք արդյունքում ստանալ մեծ տարողունակությամբ կոնտեյներ նկարներ:

Թաքնագված տվյալների դասակարգիչը ուսուցանվում է գեներատորի հետ միասին և ենթարվում է թարմացման՝ սխալ դասակարգման ժամանակ: Այս մոտեցումը թույլ է տալիս ստանալ տվյալ կոտեքստով նկարների համար բարձր ճշգրտության դասակարգիչ: Այդ դասակարգիչը չի հանդիսանում համապիտանի քանի որ նա կարող է գնահատել միայն ցածր պարամետրիզացիա ունեցող տվյալների բաշխման $K_{\text{նորմ}}$: Առաջարկվող մոտեցումը սահմանապակում է տվյալների բաշխումը, օգտագործելով միայն մարդկանց դեմքերի սահմանափակ խումբ:

2.4 Թաքնագրության կրիչի գներացիա

Սույն աշխատությունում ներկայցված համակարգում իրար են միացված 3 մոդել՝

- G գեներատորի ցանցը, որը գեներացնում է իրականին մոտ նկարներ
- D տարբերակիչը, որը որոշում է արդյո՞ք նկարը իրական է թե գեներացված
- S թաքնավերլուծիչը, որը որոշում է արդյո՞ք նկարը պարունակում է թաքնագրված ինֆորմացիա, թե ոչ

Ընդ որում գեներատորն ու տարբերակիչը իրենցից ներկայացնում են ԽՓԳՄՑ: Այս համակարգի հիմնական տարբերությունը սովորական ԳՄՑ-ներից այն է, որ ուսուցման ընթացքում գեներատորն իր կշիռները թարմացնում է միաժամանակ հիմնվելով երկու մոդելների՝ տարբերակիչի և թաքնավերլուծիչի, արդյունքների վրա: Գեներատորը փորձում է մեծացնել D-ի և S-ի սխալանքը, միևնույն ժամանակ վերջիններս փորձում են քչացնել այն:

Ներքևի հավասարումը վերը նշված օպտիմիզացիայի խնդրի մաթեմատիկական ներկայացումն է, որտեղից երևում է, որ գեներատորը խաղում է մինիմաքս խաղը միաժամանակ D-ի և S-ի հետ՝

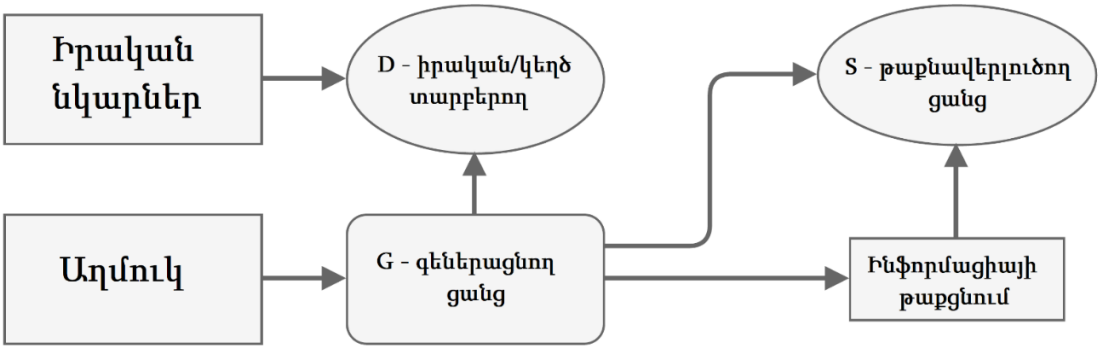
$$\alpha\Big(\mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_{noise}(z)}\left[\log(1-D(G(z)))\right]\Big) + \\ +(1-\alpha)\mathbb{E}_{z\sim p_{noise}(z)}\left[\log S(Stego(G(z))) + \log(1-S(G(z)))\right] \to \min_G \max_D \max_S \tag{20}$$

Որտեղ՝

- $p_{data}(x)$-ը x բաշխումից ստացված իրական նկարներն են, որի հիման վրա պետք է D-ն սովորի տարբերակել իրական նկարները գեներացվա
- $p_{noise}(z)$-ը z բաշխումից ստացված աղմուկն է, որի հիման վրա G-ն պետք է գեներացնի նոր նկարներ
- $Stego(x)$-ը մի ֆունկցիա է, որը x նկարի մեջ կատարում է թաքնագրում և վերադարձնում է արդեն թաքնագրված ինֆորմացիայով նկարը
- $S(x)$-ը մուտքին տրված x նկարի թաքնագրված ինֆորմացիա պարունակելու հավանականությունն է

Այս հավասարման մեջ օգտագործվել է D-ի և S-ի սխալանքների գծային գումարը α պարամետրով, որը որոշում է, թե ինչքանով է կարևոր G-ի գեներացրած նկարի ռեալիզմը թաքնակայուն կրիչ լինելու համեմատ: Այսինքն, եթե α-ն 1 է ապա մեծ հավանականությամբ գեներացված նկարները կլինեն ռեալիստիկ սակայն չեն լինի թաքնակայուն:

Այսինքն ստացվեց, որ ուսուցման ընթացքում գեներատորի մուտքին տրվելու է կամայական աղմուկ, որից այն գեներացնելու է նկար: Հետո այդ գեներացված նկարը տրվելու է տարբերակիչի մուտքին, որն իր հերթին որոշելու է արդյոք գեներացված նկարն իրական է, թե ոչ: Վերջինիս ուսուցումը վերահսկվող է, քանի որ հայտնի է, որ նկարը գեներացված է: Կախված տարբերակիչի տված արդյունքից իրենց կշիռները թարմացնելու են թե՛ գեներատորը և թե՛ տարբերակիչը: Հետո այդ նույն գեներացված նկարը տրվելու է թաքնավերլուծիչի մուտքին, որն էլ իր հերթին որոշելու է արդյոք կա թաքնագրված ինֆորմացիա թե ոչ: Թաքնավերլուծիչի ուսուցումը նույնպես վերահսկվող է, քանի որ այս փուլում մենք գիտենք, որ գեներատորի գեներացված նկարում ոչ մի թաքնագրված ինֆորմացիա չկա: Կախված թաքնավերլուծիչի տված արդյունքից վերջինս թարմացնելու է իր ցանցի կշիռները: Այս ամենից հետո նույն նկարի մեջ կատարվելու է թաքնագրում և այն նորից փոխանցվելու է թաքնավերլուծիչին: Այս դեպքում կախված թաքնավերլուծիչի արդյունքից իրենց ցանցի կշիռներն են թարմացնելու թե՛ գեներատորը և թե՛ թաքնավերլուծիչը: Այս ամենից հետո չպետք է մոռանալ, որ տարբերակիչի մուտքին անհրաժեշտ է փոխանցել իրական նկար և համապատասխանաբար թարմացնել նրա կշիռները: Մոդելները և նրանց միջև վերը նշված կապերը ներկայացված են Նկ. 5-ում



Նկ. 5 Գեներատիվ մրցակցող ցանցի մոդելներն ու նրանց կապերը

2017 թվականին կատարված հետազոտության ժամանակ օգտագործվել է տարբեր տարիքի, սեռի, մաշկի գույնի և այլ տարբեր հատկանիշներ ունեցող, հայտնի մարդկանց նկարներ: Դրա հետևանքով, $\alpha \leq 0.7$-ի դեպքում գեներատորի գեներացված նկարները այնքան ոչ-ռեալիստիկ էին ստացվել, որ ընդհանրապես պիտանի չէին: Հետևաբար աշխատության հեղինակները վերցրել էին $\alpha > 0.7$, ինչի արդյունքում բնականաբար գեներացված նկարներն ունեցել են ցածր թաքնակայունություն: Սույն աշխատությունում նախքան ուսուցումը, մարդկանց նկարները ֆիլտրվելու են, թողնելով միայն սպիտակամորթ տղամարդու նկարներ: Ինչն էլ իր հերթին կբարձրացնի G-ի գեներացված նկարների որակը: Միևնույն ժամանակ հնարավոր կլինի α-ին տալ այնպիսի արժեք՝ $\alpha = 0.5$, որը զգալիորեն կբարձրացնի նկարների թաքնակայունությունը:

2.5 Մոդելների մանրամասն նկարագրություն

2.5.1 Տարբերակիչ

Տարբերակիչ մոդելն իրենից ներկայացնում է երկուական դասակարգիչ: Նրա մուտքին տրվում է 128x128 չափի նկար, այսինքն ցանցի առաջին շերտը բաղկացած է 16384 նեյրոնից: Ելքը բաղկացած է 1 նեյրոնից, որի արժեքը ցույց է տալիս, թե որքան է հավանականությունն այն բանի, որ մուտքին տրված նկարն իրական է: Տարբերակիչի ցանցը բաղկացած է հետևյալ շերտերից.

1. Փաթույթային՝ 64 խորության, LeakyReLU ակտիվացիայով
2. Dropout՝ 40%
3. MaxPooling
4. Dropout՝ 40%
5. Փաթույթային՝ 256 խորության, LeakyReLU ակտիվացիայով
6. Dropout՝ 40%
7. MaxPooling
8. Dropout՝ 40%, ելքը՝ 8192 նեյրոն
9. Սովորական շերտ՝ 4096 նեյրոն, ելքի սիգմոիդ ակտիվացիայով
10. Սովորական շերտ՝ 1 նեյրոն, ելքի սիգմոիդ ակտիվացիայով

2.5.2 Թաքնավերլուծիչ

Տարբերակիչ մոդելը նույնպես երկուական դասակարգիչ է: Նրա մուտքին տրվում է 128x128 չափի նկար իսկ ելքը բաղկացած է 1 նեյրոնից, որի արժեքը ցույց է տալիս, թե որքան է հավանականությունն այն բանի, որ մուտքին տրված նկարում որևէ թաքնագրված ինֆորմացիա կա: Տարբերակիչի ցանցը բաղկացած է նույն շերտերից ինչ տարբերակիչը:

2.5.3 Գեներատոր

Գեներատորը մուտքին ստանում է 100 երկարության աղմուկ, իսկ ելքում տալիս է 128x128 չափի մի նկար: Այդ նկարը հանդիսանում է թաքնագրության կոնտեյներ: Գեներատորի մոդելը բաղկացած է հետևյալ շարտերից.

1. 100 նեյրոն
2. 12544 նեյրոն
3. BatchNormalization՝ տանգենցյալ ակտիվացիայով
4. Reshape(7x7x256)
5. Dropout՝ 40%
6. UpSampling
7. DeConvolution
8. BatchNormalization՝ ReLU ակտիվացիայով
9. UpSampling
10. DeConvolution
11. BatchNormalization՝ ReLU ակտիվացիայով
12. DeConvolution
13. BatchNormalization՝ ReLU ակտիվացիայով
14. DeConvolution

2.5.4 Մրցակցող մոդելներ

2 մրցակցող մոդելների զույգերն են՝ գեներատոր-տարբերակիչ և գեներատոր-թաքնավերլուծիչ այս 2 մոդելների ուսուցման ժամանակ մենք չենք թարմացնում տարբերակիչի և թաքնավերլուծիչի կշիռները, քանի որ այս դեպքում մեր խնդիրը գեներատորի որակը լավացնելն է: Տարբերակիչն ու թաքնավերլուծիչը թարմացնում են իրենց կշիռները ուսուցման այլ քայլերի ընթացքում:

Ուսուցումը բաղկացած է հետևյալ քայլերից՝

1. Տարբերակիչի ուսուցում
2. Թաքնավերլուծիչի ուսուցում
3. Գեներատոր-Տարբերակիչ մրցակցող մոդելների ուսուցում
4. Գեներատրո-Թաքնավերլուծիչ մրցակցող մոդելների ուսուցում

Ընդ որում առաջին երկու քայլերի ընթացքում թարմացվում են տարբերակիչի և թաքնավերլուծիչի կշիռները, իսկ վերջի երկու ուսուցումներն ուղղված են գեներատորի որակի լավացմանը, այսինքն միայն գեներատորի կշիռներն են թարմացվում:

Առաջին քայլի ընթացքում տարբերակիչի մուտքին տրվում են ինչպես գեներատորի գեներացրած նկարները, այնպես էլ իրական նկարներ: Երկրորդ քայլի ընթացքում թաքնավերլուծիչին տրվում են գեներատորի գեներացված նկարները՝ որպես առանց ինֆորմացիայի թաքնագրման և թաքնագրումով: Ընդ որում թաքնագրման համար օգտագործվում է պարզագույն «LSB-թաքնագրում» ալգորիթմը:

2.6 Ուսուցման տվյալներ

Ուսուցման ընթացքում օգտագործվում է հայտնի աստղերի նկարներ: Նկարները վերցված են հանրահայտ «Kaggle»[12] կայքից: Կայքում տրամադրված նկարների քանակը գերազանցում է 2 միլիոնը: Ընդ որում բոլոր նկարներն ունեն մոտ 40 հատկություններ որոնք հեշտացնում են նկարների ֆիլտրացիան: Ինչպես կամայական մոդելի ուսուցման դեպքում մեր դեպքում նույնպես շատ կարևոր է մուտքային տվյալների ֆիլտրացիան: Հենց այդ պատճառով էլ մոդելների ուսուցմանն անցնելուց առաջ նախ կատարվում է

նկարների ֆիլտրացիա: Հիմնվելով նկարների հատկությունների վրա ընտրվում են միայն այն նկարները, որոնք՝

- վնասված չեն
- պատկերված է տղամարդ
- չունեն բեղ
- չեն կրում վզնոց

Այս կրիտերիաները թույլ են տալիս ավելի ռեալիստիկ նկարների գեներատոր ուսուցանել:

2.7 Python լեզուն

Python-ը բարձր կարգի[14], ինտերպրիտացվող[16], դինամիկ[17], սակայն ուժեղ տիպիզացված[15] (Dynamically typed, Strong typed) լեզու է: Նշված հատկությունները Python-ին դարձնում են մեքենայական ուսուցման խնդիրների լուծման համար բավական հարմար լեզու: Բացի նշված հատկություններից, Python լեզվի բարձր տարածվածությունը տվյալագիտության ոլորտում կապված է նաև այնպիսի գործոնների հետ ինչպիսիք են՝ լեզվի պարզ սինտաքս (syntax), բազմաքանակ և զարգացած գրադարանների և հավելումների (plug-in) առկայություն, լավ դոկումենտացիա, մեծ և զարգացած համայնք (community):

Բարձր կարգի լեզվի առավելությունը կայանում է նրանում, որ այն անկախ է պլատֆորմից, ապարատային ապահովումից (hardware) և նրա ճարտարապետությունից: Ինչպես նաև այն ավելի հեշտ է սովորել քան ցածր մակարդակի լեզուները, ինչպես նաև այն շատ ավելի հեշտ է կարգաբերել (debug): Python-ը լինելով բարձր կարգի լեզու, բնականաբար ունի նշված առավելությունները:

Լեզվի պարզ սինտաքսն օգնում է Python-ի սկսնակ ծրագրավորողներին կարճ ժամանակում տիրապետել լեզվին, ինչն արագացնում է կամայական նախագծի ծրագրավորման գործընթացը: Ինչպես նաև այն, քիչ թե շատ, հեշտացնում է անծանոթ կոդի կարդալն ու հասկանալը:

Դինամիկ տիպիզացված լեզուն շատ հարմար է օգտագործման համար: Օրինակ հարկ եղած դեպքում կարելի է զանգվածի մեջ պահել տարբեր տիպի օբյեկտներ, ինչը

կարող է պետք գալ արագորեն ինչ-որ բան թեստավորելու համար: Սակայն պետք է զգույշ լինել կատարման ընթացքում բացառիկ իրավիճակներից (Runtime Exceptions) խուսափելու համար:

Վերջին 2 հատկությունների առավելություններն ավելի լավ պատկերացնելու համար դիտարկենք հետևյալ օրինակը՝

$$\text{arr} = [1, 20, -334, 44.44, -555.50005, \text{“str\_val\_1”}, \text{“str\_val\_2”}] \quad (21)$$

Այստեղ, arr զանգվածի առաջին երեք էլեմենտները ամբողջ թվեր են, հաջորդ երկուսը՝ կոտորակային, իսկ վերջին երկուսն ընդհանրապես թվեր չեն, այլ տողային օբյեկտներ: Օգտագործելով «+» օպերատորը, կարելի է իրար միացնել 2 կամ ավելի զանգվածներ և ստեղծել նոր զանգված: Ինչպես նաև կարելի է զանգվածի սկզբից կամ վերջից հեռացնել որոշ էլեմենտներ ընհամենը մեկ տողով: Օրինակ, քիչ 21 արտահայտությունում օգտագործված arr զանգվածի վերջին 2 էլեմենտները հեռացնելու համար կարող ենք կատարել հետևյալը՝

$$\text{arr\_no\_str} = \text{arr}[:-2] \quad (22)$$

որից հետո, arr_no_str-ի արդյունքը կլինի՝

$$[1, 20, -334, 44.44, -555.50005] \quad (23)$$

Python-ի սինտաքսային առանձնահատկություններից կարևոր է նշել նաև զանգվածների ձևափոխությունը: Օրինակ, 24 արտահայտությունով կարելի է ստանալ, նոր զանգված, որը կպարունակի 21 արտահայտությունում նշված arr զանգվածի էլեմենտների տիպերը, իսկ 25 արտահայությունով՝ մի զանգված, որը կպարունակի այդ նույն arr զանգվածի բոլոր այն աբողջ թվերի ½ մասը, որոնք բաժանվում են 2-ի:

$$\text{arr\_types} = [\text{type(x) for x in arr}] \quad (24)$$

$$[\text{x/2 for x in arr if isinstance(x, int) and x \% 2 == 0}] \quad (25)$$

Համապատասխանաբար 24 և 25 արտահայտությունների արդյունքը կլինեն 26 և 27 արտահայտությունները:

$$[<\text{class 'int'}>, <\text{class 'int'}>, <\text{class 'int'}>, <\text{class 'float'}>, \quad (26)$$

<class 'float'>, <class 'str'>, <class 'str'>]

[10.0, -167.0] (27)

Գրադարանների ու ֆրեյմվորկերի լայն ընտրություն. Մեքենայական ուսուցման ալգորիթմների իմպլեմենտացիան բարդ է և շատ ժամանակ կարող է պահանջել: Լավ թեստավորված ու կազմակերպված միջավայրի հասանելիությունը ծրագրավորողների համար կենսական նշանակություն ունի:

Որպեսզի կրճատվի ծրագրավորման ժամանակը, կարելի է օգտագործել Python-ի ֆրեյմվորկներին ու գրադարաններին: Ծրագրային գրադարանը նախապես գրված կոդ է, որն օգտագործվում է ծրագրավորողների կողմից հայտնի ծրագրավորման խնդիրներ լուծելու: Python-ն ունի մեքենայական ուսուցման գրադարանների լայն բազմություն: Դրանցից մի քանիսը նշված են ներքևում՝

- Keras, TensorFlow և Scikit-learn - մեքենայական ուսուցման համար,
- NumPy, SciPy - բարձր արագագործությամբ գիտական հաշվարկների ու տվյալների վերլուծության համար,
- Pandas - տվյալների վերլուծության և ձևափոխության համար,
- Seaborn - տվյալների վիզուալիզացիայի համար:

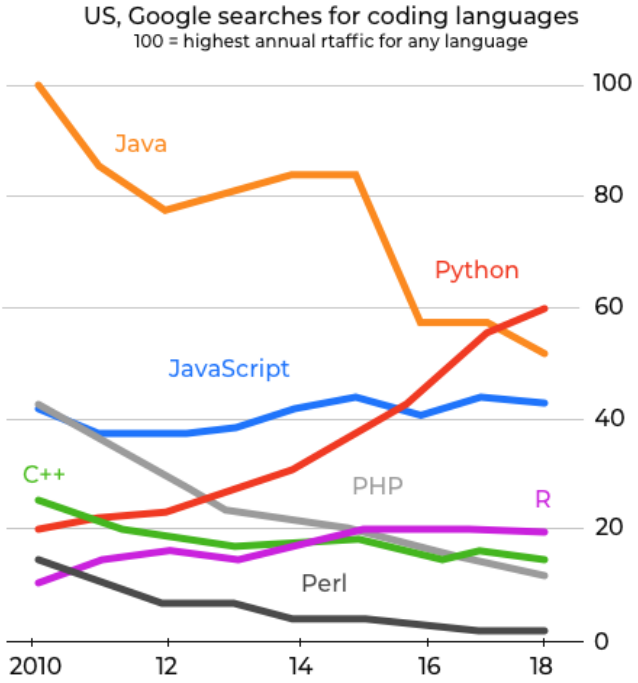
Scikit-learn-ը տրամադրում է բազմազան կլասիֆիկացիայի, րեգրեսիայի և կլաստերիզացիայի ալգորիթմներ՝ ներառյալ random forests, gradient boosting, k-means, և DBSCAN ալգորիթմները: Այն նախատեսված է աշխատելու Python-ի թվային և գիտական գրադարանների հետ՝ NumPy և SciPy :

Պլատֆորմից անկախություն. խոսքը ծրագրավորման լեզվի, ֆրեյմվորկի, գրադարանի մասին է, որը հնարավորություն է տալիս ծրագրավորողին աշխատել մեկ մեքենայի վրա, բայց օգտագործել ստեղծված ծրագիրն այլ մեքենաների վրա առանց ավելորդ փոփոխությունների: Python-ի առանձնահատկություններից մեկը պլատֆորմից, ապարատային ապահովումից (hardware) և մեքենայի ճարտարապետությունից անկախ լինելն է: Python-ով կոդը կարող է օգտագործվել

ստեղծելու կատարողական ֆայլեր բոլոր տարածված օպերացիոն համակարգերի համար:

Մեքենայական ուսուցման մոդելներ ուսուցանելու համար ծրագրավորողները հաճախ օգտվում են այնպիսի ծառայություններից, ինչպիսիք են՝ Google-ը կամ Amazon-ը: Այնուամենայնիվ կան շատ կազմակերպություններ ու անհատներ, որ օգտագործում են իրենց հզոր GPU-ով մեքենաները, և այն փաստը, որ Python-ն անկախ է պլատֆորմից, դարձնում է այդ ամենը շատ ավելի էժան և հեշտ:

Մեծ և զարգացած համայնք (community). Stack Overflow-ի կողմից 2018թ.-ին անցկացված հարցման համաձայն Python-ը 10 ամենատարածված ծրագրավորման լեզուների ցանկում է: Իսկ ներքևում բերված գրաֆիկից ակնհայտ է, որ Python-ը մարդկանց կողմից Google-ով ամենահաճախ փնտրված լեզուն է:



Օնլայն րեպոզիտորիաները պարունակում են ավելի քան 140,000 Python-ով ծրագրավորված նախագծեր: Այդ ծրագրային ապահովումները հնարավորություն է տալիս ծրագրավորողներին առանձնացնել փաթեթերները տվյալների մեծ բազմություններում:

2.8 Ծրագրային իրականացում

Սույն աշխատության ծրագրային իրականացման համար առաջին հերթին պետք է ընտրել համապատասխան ծրագրավորման լեզու: GitHub-ը, լինելով աշխարհում ամենատարածված կոդի կառավարման համակարգն ու դրա հոսթինգի հարթակը հրապարակել[13] է այդ հարթակում ամենատարածված ծրագրավորման լեզուները մեքենայական ուսուցման նախագծերում: Ներքևում բերված են այդ հրապարակման մեջ տեղ գտած լեզուները՝ տարածվածության նվազման կարգով.

1. Python
2. C++
3. JavaScript
4. Java
5. C#
6. Julia
7. Shell
8. R
9. TypeScript
10. Scala

Բնականաբար, չկա որևէ լեզու որն ամենալավն է մեքենայական ուսուցման համար: Այն պետք է ընտրել կախված՝ խնդրից, տվյալագիտության ոլորտում ունեցած փորձից և թե ինչ նպատակ է հետապնդում խնդրի լուծումը:

Սույն աշխատությունում բոլոր մեքենայական ուսուցման մոդելները նեյրոնային ցանցեր են: Այն իրականացնելու համար ընտրվել Python լեզուն՝ ելնելով Python լեզվի, նախորդ գլխում նշված, առավելություններից: Իսկ մոդելներ ստեղծելու համար օգտագործվել է Keras ֆրեյմվորկը:

Ծրագրային իրականացման հիմնական 2 դասերն են՝ **SGANModel** և **SGANTrainer**, ընդ որում առաջին դասը պարունակում նեյրոնային ցանցերը իսկ 2-րդն ուսուցանում է այդ ցանցերը: **SGANModel**-ի մեթոդներից 3-ը՝ **discriminator(), generator(), steganalyser()**, ստեղծում են այնպիսի ցանցեր, որոնք կարող ենք օգտագործել որպես միջանկյալ

շերտեր՝ այլ ցացներ կառուցելիս: Այս 3 մեթոդները առաջին կանչի ժամանակ ստեղծում են նոր ցանց և քեշավորում են այն: Մնացած բոլոր կանչերի ժամանակ նրանք վերադարձնում են նույն ցանցը: Այս քայլի հիմնավորումը կտանք քիչ հետո: **SGANModel**-ի 2 կարևոր մեթոդները գեներատոր-տարբերակիչ և գեներատոր-թաքնավերլուծիչ մրցակցող ցանցեր ստեղծող մեթոդներն են՝ **dcgan_adversarial_model()** և **stegano_adversarial_model()**: Այս մեթոդները օգտվում են նախորդ 3 մեոդներից, որպեսզի կառուցեն նոր ցանց բաղկացաց ենթացանցերից: Ընդ որում գեներատոր-տարբերակիչ ցանցի դեպքում տարբերակիչի, իսկ գեներատոր-թաքնավերլուծիչի դեպքում՝ թաքնավերլուծիչի, **trainable** պարամետրին տալիս ենք **False,** քանի որ մրցակցող ցանցերն ուսուցանելիս մեր նպատակը գեներատորի ուսուցանումն է, ոչ թե տարբերակիչի ու թաքնավերլուծիչի, հետևաբար միայն գեներատորը պետք է թարմացնի իր կշիռները: Դրանց ուսուցման համար անհրաժեշտ են 2 նոր մեթոդներ՝ **discriminator_model()** և **steganalyser_model()**: Վերջիններս ստեղծում են ցանցեր, օգտագործելով **discriminator()** և **steganalyser()** մեթոդները, սակայն այս դեպքում վերջիններիս վերադարձրած ցանցերի **trainable** պարամետրին տալիս ենք **True** արժեք, քանի որ հենց այս մոդելների (ցանցերի) ուսուցման ժամանակ է, որ տարբերակիչն ու թաքնավերլուծիչը պետք է թարմացնեն իրենց կշիռները: Հենց այս **trainable** պարամետրի արժեքի փոփոխման համար էր, որ **generator()** և **discriminator()** մեթոդները քեշավորում էին իրենց ստեղծած ցանցերը, քանզի մեզ անհրաժեշտ է, որ գեներատորի ու թաքնավերլուծիչի թարմացված կշիռները հասանելի լինեն մրցակցող ցանցերի ուսուցման ժամանակ:

SGANTrainer դասի կարևոր մեթոդը **train()** մեթոդն է: Այն օգտագործելով **SGANModel** դասը ուսուցանում է մոդելներ: Նկարների բեռնումը օպերացիոն հիշողության մեջ կատարվում է նրա կոնստրուկտորի մեջ, որից հետո այդ նկարներն օգտագործելով կատարվում է ուսուցումը: Հարկ է նշել, որ նկարները բեռնելուց հետ նրանց չափերը փոքրացրել ենք՝ բերելով 128x128-ի, քանի որ գեներատորի գեներացրած նկարներն այդ չափերի են և հետևաբար դիսկրիմինատորի ու թաքնավերլուծիչի մուտքին մենք նույնպես պետք է տանք 128x128 չափի նկարներ:

Գլուխ 3. Բնապահպանություն

3.1 Էկոլոգիական փորձաքննության նպատակները և խնդիրները

Բնապահպանական կազմակերպությունների, կոմիտեների և հասարակական կազմակերպությունների գործունեության հիմնական ուղղություններից մեկը էկոլոգիական փորձաքննությունն է:

Համաձայն Հայաստանի Հանրապետության «Մթնոլորտային օդի պահպանության մասին» «Շրջակա միջավայրի վիա ազդեցության փորձաքննության մասին» օրենքների՝ շրջակա միջավայրի վրա ազդեցության (էկոլոգիական) փորձաքննությունը պետության կողմից անցկացվող պարտադիր գործունեություն է, որի հիմնական նպատակն է կանխորոշել, կանխարգելել կամ նվազագույնի հասցնել հայեցակարգի ն. նախատեսվող գործունեության վնասակար ազդեցությունը մարդու առողջության, շրջակա միջավայրի, տնտեսական և սոցիալական բնական զարգացման վրա:

Շրջակա միջավայրի վրա ազդեցության փորձաքննությունը ելնում է՝

- մարդու առողջության, բնականոն ապրելու և ստեղծագործելու համար բարենպաստ շրջակա միջավայր ունենալու իրավունքից,
- բնական պաշարների արդյունավետ, համալիր և բանական օգտագործման պահանջներից,
- էկոլոգիական համակարգերի հավասարակշռության և բնության մեջ գոյություն ունեցող բույսերի ն կենդանիների բոլոր տեսակների պահպանման անհրաժեշտությունից՝ նկատի ունենալով ներկա և ապագա սերունդների շահերը:
- էկոլոգիական փորձաքննությունը հատուկ ստեղծված մարմինների, խմբերի առանձին փորձագետների փորձաքննական գործունեության տեսակ է՝ հիմնված փորձաքննման օբյեկտի միջառարկայական՝ էկոլոգա-տնտեսական-սոցիալական հետազոտման, ստուգման և գնահատման վրա, նպատակ ունենալով դրա իրականացման մասին որոշման կայացումը այն անձի կողմից,

ով իրավասու է կայացնելու այդպիսի որոշում: Պետք է նկատի ունենալ, որ Հայաստանի Հանրապետությունում իրականացվում է ինչպես պետական, այնպես էլ հասարակական էկոգիական փորձաքննություն:

Պետական էկոլոգիական փորձաքննությունը կանխարգելող հսկողության կազմակերպչական իրավական ձև է: Միաժամանակ, այն դուրս է գալիս «հսկողություն» հասկացության սահմաններից՝ հանդիսանալով կառավարչական գործունեության ինքնուրույն տեսակ: Պետական էկոլոգիական փորձաքննությունը պետական մարմինների և փորձաքննության հանձնախմբի հատուկ համալիր գործունեություն է: Պետական էկոլոգիական փորձաքննության նպատակը շրջակա բնական միջավայրի պաշտպանության և էկոլոգիական անվտանգության պահանջներին փորձաքննության օբյեկտների համապատասպանությունը ստուգելը և գնահատելն է:

Պետական էկոլոգիական փորձաքննության սկզբունքներն ամրագրված են օրենսդրորեն և նախատեսում են առաջին հերթին՝ փորձաքննության պարտադիր անցկացումը: Պետական էկոլոգիական փորձաքննությունը պետք է նախորդի տնտեսական որոշման կայացմանը՝ նպատակ ունենալով կանխարգելելու շրջակա միջավայրի վրա հնարավոր վնասակար ազդեցությունը: էկոլոգիական փորձաքննության անցկացումը պարտադիր է բոլոր նախագծերի և ծրագրերի համար: Որպես պարտադիր պետական էկոլոգիական փորձաքննության երաշխավոր նախատեսվում է այն հանգամանքը, որ նախագծերի և ծրագրերի աշխատանքների ֆինանսավորումը հնարավոր է միայն փորձաքննության դրական եզրակացության առկայության դեպքում: էկոլոգիական փորձաքննությունը հանդես է գալիս որպես շրջակա բնական միջավայրի պահպանության մեխանիզմի գործելու երաշխավոր:

Պետական էկոլոգիական փորձաքննության եզրակացությունների գիտական հիմնավորվածության և օրինականության սկզբունքն արտացոլում է դրա երկու ուղղությունները - գիտական և վարչաիրավական:

Փորձաքննությունը գիտահետազոտական գործընթաց է, հետևաբար, այն պետք է իրականացվի ժամանակակից գիտա-տեխնիկական մակարդակով, գիտական հետազոտությունների նոր ձևերի և մեթոդների օգտագործմամբ, որակյալ գիտնական-փորձագետների ընդգրկմամբ: Աշխատանքի արդյունքը պետք է լինի ոչ միայն թույլ

տրված էկոլոգիական նորմատիվների խախտումների արձանագրումը, այլ նաև դրանց հետևանքների գիտականորեն հիմնավորված գնահատումը թերությունների ուղղման և վերացման համար, որոշում կայացնող մարմիններին երաշխավորությունների տրամադրումը՝ ինչպես նաև փորձաքննվող նախագծերի և օբյեկտների ամենաարդյունավետ ձևով իրականացման պայմանների կանխատեսումը:

Պետական էկոլոգիական փորձաքննության անկախության, արտագերատեսչականության սկզբունքը նշանակում է, որ դրա արդյունավետության պարտադիր պայմանը փորձաքննություն կազմակերպող և իրականացնող մարմինների ֆինանսական անկախությունն է, փորձագետների արտահաստիքային կարգավիճակը:

Կազմակերպորեն պետական էկոլոգիական փորձաքննությունը այնպիսի համակարգ է, որի կառուցվածքն ուղղված է պետական էկոլոգիական փորձաքննության արտագերատեսչականության ապահովմանը: Փորձաքննության հանձնախմբերի, խմբերի ղեկավարությունը, ինչպես նաև փորձաքննության անցկացումը իրականացվում են հիմնականում արտահաստիքային փորձագետների կողմից:

Փորձաքննության ֆինանսական անկախությունն ապահովվում է նրանով, որ այն ֆինանսավորվում է Հայաստանի Հանրապետության բյուջեից և այն միջոցների հաշվին, որոնք ստացվում են պատվիրատուներից փորձաքննության անցկացման, այդ թվում՝ փորձաքննության կրկնակի անցկացման համար: Պատվիրատուների թվարկած ֆինանսական միջոցները ծախսվում են բացառապես պետական էկոլոգիական փորձաքննության վրա՝ դրա անցկացման համար կազմված նախահաշվին լիովին համապատասխան: Էկոլոգիական փորձաքննության ոլորտում հատուկ լիազորված պետական մարմինը պատասխանատվություն է կրում այդ միջոցների նպատակային օգտագործման համար:

Օբյեկտի փորձաքննության իրականացման դեպքում դրա անցկացման ընթացքի, ընդունված որոշումների և կառավարման մարմինների կողմից դրանք հաշվի առնելու վերաբերյալ տեղեկատվությունը պետք է հասանելի լինի բնակչության լայն զանգվածների համար: Կազմակերպորեն փորձաքննության վերաբերյալ աշխատանքը պետք է կառուցված լինի այնպես, որ հասարակական կազմակերպությունները և

քաղաքացիները տեղեկություն ստանան և կարողանան որոշում կայացնող մարմիններին ի գիտություն հասցնել իրենց դիրքորոշումը:

Պետական էկոլոգիական փորձաքննության առարկա են հանդիսանում օբյեկտների և միջոցառումների բերաբերյալ բոլոր նյութերը, որոնք նախատեսվում է իրականացնել Հայաստանի Հանրապետության տարածքում: Տարբերակում են պլանային աշխատանքները և նախապլանային փաստաթղթերը: Առաջինին են վերաբրում տնտեսության ճյուղերի զարգացման, շրջակա բնական միջավայրի վիճակի վերաբերյալ և այլ կանխատեսումները, երկրորդին՝ տեղաբաշխման սխեմաները, շրջանների հատակագծման և քաղաքների կառուցապատման սխեմաները և նախագծերը, առանձին բնական ռեսուրսների օգտագործման համալիր փոքր գետերի պահպանության սխեմաները այլն: Փորձաքննության օբյեկտների թվին են դասվում նաև բոլոր մինչնախագծային նյութերը՝ ըստ օբյեկտների և միջոցառումների, որոնք նախատեսվում է իրականացնել Հայաստանի Հանրապետությունում: Մինչնախագծային նյութերը տեխնիկա-տնտեսական հիմնավորումներն (ՏՏՀ) են, շինարարական նախագծման հիմնական դրույթները, շինարարական նախագծման հատուկ պայմանները, նախագծման առաջադրանքները, հարթակի (ուղեգծի) ընտրման նյութերը և այլն: «Նախագծային նյութեր» տերմինը վերաբերում է նաև օրենսդրական և այլ նորմատիվ իրավական ակտերի նախագծերին, որոնց ամրագրումը կարող է հանգեցնել բնական շրջակա միջավայրի վրա վնասակար ազդեցության:

Թվարկած օբյեկտները ենթակա են պետական էկոլոգիական փորձաքննության՝ անկախ դրանց նախահաշվային արժեքից և պատկանելությունից: Այս ճանապարհով վերացվում են գերատեսչական խոչընդոտները, այսինքն՝ պետական փորձաքննության ենթակա են ինչպես քաղաքացիական, այնպես էլ ռազմական պաշտպանական օբյեկտները:

Էկոլոգիական փորձաքննության օբյեկտներին են վերաբերում բնօգտագործման համար տրված լիցենզիաների էկոլոգիական, ինչպես նաև սերտիֆիկատների էկոլոգիական հիմնավորումները:

Պետական էկոլոգիական փորձաքննության եզրակացությունը փորձաքննող հանձնաժողովի կողմից պատրաստված փաստաթուղթ է, որը բովանդակում է

փորձաքննված գործունեության թույլատրելիության և պետական էկոլոգիական փորձաքննության օբյեկտի հնարավոր իրականացման վերաբերյալ հիմնավորված եզրակացություններ: Այդ փաստաթուղթը պետք է հավանության արժանանա փորձաքննական հանձնաժողովի ցուցակային կազմի որակյալ մեծամասնության կողմից:

Պատրաստված փաստաթուղթը պետական էկոլոգիական փորձաքննության եզրակացության կարգավիճակ ձեռք է բերում էկոլոգիական փորձաքննության ոլորտում հատուկ երաշխավորված պետական մարմնի կողմից հաստատ հետո: Պետական էկոլոգիական փորձաքննության դրական եզրակացությունը իրավաբանական ուժ ունի այն ժամանակահատվածում, որ որոշել է էկոլոգիական փորձաքննության ոլորտում պետական հատուկ երաշխավորված մարմինն, ը պետական էկոլոգիական փորձաքննության օբյեկտի ֆինանսավորման իրականացման պարտադիր պայմաններից մեկն է:

Պետական էկոլոգիական փորձաքննության բացասական եզրակացության իրավական հետևանքը պետական էկոլոգիական փորձաքննության օբյեկտի իրականացման արգելումն է: Բացասական եզրակացության դեպքում պատվիիատուին իրավունք է տրվում կրկին անգամ ներկայացնելու նյութերը պետական էկոլոգիական փորձաքննության: Այս դեպքում պարտադիր պայման է բացասական եզրակացությունում նշված դիտողությունների վերացումը: Բացի դրանից, պատվիրատուն իրավունք ունի եզրակացությունը վիճարկելու դատական կարգով:

Հասարակական էկոլոգիական փորձաքննությունը կազմակերպվում և անց է կացվում քաղաքացիների, հասարակական կազմակերպությունների (միավորումների), ինչպես նան տեղական ինքնակառավարման մարմինների նախաձեռնությամբ: Այդպիսի փորձաքննություն անցկացնում են գիտական կոլեկտիվները, հասարակական միավորումները: Գործնականում խոսքը առավելապես ժամանակավոր կոլեկտիվների, հանձնախմբերի և խմբերի մասին է: Հասարակական միավորումներ ասելով պետք է հասկանալ քաղաքացիների կամավոր միավորումները:

Հասարակական կազմակերպություններն իրավունք ունեն.

- պատվիրատուից ստանալու էկոլոգիական փորձաքննության ենթակա փաստաթղթերը,
- ծանոթանալու նորմատիվտեխնիկական փաստաքղթերին, որոնցով սահմանվում են պետական էկոլոգիական փորձաքննության անցկացման պահանջները,
- իրենց ներկայացուցիչների միջոցով, որպես դիտորդ, մասնակցելու պետական էկոլոգիական փորձաքննության փորձաքննական հանձնաժողովի նիստերին և դրանցում հասարակական էկոլոգիական փորձաքննության եզրակացության քննարկմանը:

Հասարակական էկոլոգիական փորձաքննություն կազմակերպող հասարակական կազմակերպությունները պարտավոր են տեղեկացնել բնակչությանը դրա սկզբի և արդյունքների վերաբերյալ:

Հասարակական էկոլոգիական փորձաքննության եզրակացությունը կրում է երաշխավորական, տեղեկատվական բնույթ: Սակայն այն դառնում է իրավաբանորեն պարտադիր դրա արդյունքների պետական էկոլոգիական փորձաքննության համապատասխան մարմինների կողմից հաստատվելուց հետո:

Հասարակական փորձաքննական կոլեկտիվների անդամները իրենց փորձաքննական գնահատականների ճշտության, հիմնավորվածության համար պատասխանատվություն են կրում՝ համաձայն Հայաստանի Հանրապետության օրենսդրության:

Չնայած հասարակական և պետական էկոլոգիական փորձաքննության նպատակները համընկնում են, սակայն դրանց խնդիրները տարբեր են: Որպես կանոն, հասարակական փորձաքննությունը անմիջական փորձաքննության խնդիրների հետ միասին նպատակ ունի պետական մարմինների ուշադրությունը սևեռելու կոնկրետ օբյեկտին, էկոլոգիական վտանգավորության վերաբերյալ գիտականորեն հիմնավորված տեղեկատվությունը հասու դարձնելու լայն հասարակայնությանը:

Գլուխ 4. Կենսագործունեության անվտանգություն

4.1 Կլաստերներից առաջացած աղմուկի ազդեցությունը մարդու վրա

Մեքենայական ուսուցման տարբեր տեսակի խնդրիներ լուծելիս առավել լավ արդյունքների հասնելու համար, միշտ էլ կարիք է լինում օգտագործել հզոր հաշվողական տեխնիկա: Քանի որ, ինչքան հզոր լինի հաշվողական տեխնիկան այնքան մոդելների ուսուցումն ավելի արագ կկատարվի, և հնարավոր կլինի ժամանակի անհամեմատ ավելի փոքր ինտերվալում ստանալ շատ ավելի մեծ ճշտություն ունեցող մոդելներ: Պահանջվող հաշվողական հզորությունը կարելի է ապահովել օգտագործելով մեծ կլաստերներ: Վերջիններս, ինչքան էլ որ նպատակահարմար լինեն տվյալ խնդրի լուծման համար, ունեն իրենց բացասական կողմերը, որոնցից ամենաազդեցիկը աղմուկն է: Անկախ այն բանից, թե ի՞նչ բարձրության է ձայնը, եթե այն չի դադարում կամ պարբերաբար կրկնվում է ժամանակի ընթացքում, ապա կարող է ազդել մարդու հոգևոր և ֆիզիկական վիճակի վրա: Իսկ կլաստերները երբեք չեն դադարում աշխատել, հետևաբար նրանց արձակած ձայնն անընդհատ է:

Ձայն հասկացությունը, որպես կանոն, ասոցացվում է մարդու լսողական զգացողությունների հետ, ով նորմալ լսողություն ունի: Լսողական զգացողությունները առաջ են գալիս առաձգական միջավայրի տատանումներից, որոնք իրենցից ներկայացնում են գազ, հեղուկ կամ պինդ միջավայրում տարածվող և մարդու լսողական ապարատի վրա ազդող մեխանիկական տատանումներ: Ընդ որում, միջավայրի այդ տատանումները որպես ձայն ընկալվում են միայն հաճախականությունների որոշակի միջակայքում:

Մարդն ընդունակ է որպես ձայն ընկալելու օդի 16-20000 Հց հաճախականությամբ տատանումները: 16 Հց-ից փոքր հաճախականությամբ տատանումները անվանում են ինֆրաձայն և ընկալվում են միայն որպես թրթռոցներ, իսկ 20000 Հց-ից բարձր հաճախականությամբ տատանումները անվանում են ուլտրաձայն և մարդու կողմից լսողությամբ չեն ընկալվում :

Լսելիության միջակայքից ցածր և բարձր տատանման հաճախականությունները կոչվում են, համապատասխանաբար, ինֆրաձայնային և ուլտրաձայնային, դրանք կապ չունեն մարդու լսողական զգացողությունների հետ և ընկալվում են որպես միջավայրի ֆիզիկական ազդեցություններ :

Եթե հոծ միջավայրում գրգռվեն տատանումներ, ապա նրանք կտարածվեն բոլոր ուղղություններով: Ակնառու օրինակ են հանդիսանում ալիքների տատանումները ջրի վրա: Ընդ որում պետք է տարբերել մեխանիկական տատանումների տարածման արագությունը և գրգռող ազդեցության տարածման արագությունը :

Ֆիզիկական տեսակետից տատանման տարածումը կայանում է մեկ մոլեկուլից մյուսին շարժման իմպուլսի փոխանցման մեջ: Առաձգական միջմոլեկուլային կապերի շնորհիվ յուրաքանչյուր մոլեկուլի շարժումը կրկնում է նախորդի շարժումը: Իմպուլսի փոխանցումը պահանջում է ժամանակի որոշակի ծախսում, ինչի արդյունքում դիտման կետերում մոլեկուլների շարժումը տեղի է ունենում տատանումների գրգռման գոտում մոլեկուլների շարժման համեմատ ուշացումով: Այսպիսով, տատանումները տարածվում են որոշակի արագությամբ:

Ձայնային ալիքի տարածման արագությունը միջավայրի ֆիզիկական հատկությունն է : Կախված տատանումների գոգռման եղանակից տարբերում են ալիքների մի քանի տեսակներ.

- հարթ, որը ստեղծվում է հարթ տատանվող մակերևույթի միջոցով
- գլանաձև, որը ստեղծվում է գլանի շառավղային տատանվող կողային մակերևույթի միջոցով
- գնդային, որը ստեղծվում է բաբախող գնդի տիպի տատանումների կետային աղբյուրի միջոցով

Ձայնային ալիքը բնութագրող հիմնական պարամետրերն են հանդիսանում ձայնային ալիքի երկարությունը, ալիքի տարածման արագությունը, տատանման հաճախությունը, ձայնային ճնշումը, ձայնի ինտենսիվությունը :

Մարդու հիմնական զգայարաններից լսողությունը շատ մեծ դեր է խաղում նրա կյանքում: Այն թույլ է տալիս մարդուն տիրապետել ձայնային ինֆորմացիոն դաշտերին:

Շրջակա միջավայրի հագեցումը բարձր ինտենսիվությամբ աղմուկներով բերում է ձայնային ինֆորմացիայի աղավաղման և մարդու լսողական ակտիվության խախտմանը: 135-140 դԲ արժեքի ձայնային գոգռիչների դեպքում, մարդու ներքին ականջի տարրերը առաջվա նորմալ տատանումների փոխարեն սկսում են տեղափոխվել մի կողմից մյուս կողմ, իջեցնելով խեցիում ճնշման և արտաքին միջավայրից ձայնային ճմշման տարբերությունը:

Ցանկացած պաշտպանողական համակարգ ունի իր սահմանափակումները: Այդ իսկ պատճառով ավելցուկային աղմուկները, որոնց ազդեցության ժամանակահատվածը նույնիսկ աննշան է, առաջացնում են ներքին ականջի վնասվածք, որը լավագույն դեպքում արտահայտվում է լսողության շեմի ժամանակավոր խախտմամբ: Վերականգնման ժամանակահատվածը կարող է տևել մի քանի րոպեից մինչև մի քանի օր՝ կախված վնասվածքի աստիճանից:

Արտադրական բնույթի աղմուկը փոփոխվում է ըստ ինտենսիվության և ըստ հաճախության՝ կախված այն մեքենաների, մեխանիզմների տիպերից և քանակությունից, որոնք օգտագործվում են տեխնոլոգիական գործընթացում:

Նույն ինտենսիվություն ունեցող տարբեր հաճախականությամբ ձայները մարդու կողմից ընկալվում են տարբեր բարձրությամբ: Միաժամանակ տարբեր հաճախականության և ինտենսիվության ձայները կարող են ընկալվել որպես նույն բարձրության ձայներ:

Շրջակա միջավայրի աղտոտումը աղմուկով և դրա ազդեցությունը մարդու վրա նպատակահարմար է հաշվարկել, օգտագործելով աղմուկի էներգիայի մակարդակին համարժեք մեծությունը՝ $E_{համ}$: Վերջինս կախված է E(t)-ից՝ ժամանակի ընթացքում աղմուկի էներգիայի փոփոխությունից, որտեղ t-ն աղմուկի ազդեցության ժամանակահատվածն է:

Համարժեք էներգիան պետք է փոքր լինի առավելագույն թույլատրելի էներգիայից, որի դեպքում ի հայտ են գալիս բացասական հետևանքներ: Ենթադրվում է, որ վնասվածքը, որը առաջացնում է փոփոխական աղմուկի E(t) ազդեցությամբ, հավասար է այն վնասվածքին, որը առաջանում է $E_{համ}$ էներգիայով հաստատուն աղմուկը: Այսպիսով,

եթե աղմուկի ազդման ժամանակամիջոցը նվազում է 2-ից 3 անգամ, ապա ձայնային էներգիայի թույլատրելի մաքսիմալ մակարդակը կարելի է ավելացնել նույնքան անգամ:

Ակուստիկ տատանումները, որոնք դուրս են գտնվում մարդու նորմալ ձայնաընկալման տիրույթից (16…20000 Հց), նույնպես կարող են բերել լսողության վատացման: Այսպես, ուլտրաձայնը (>20000 Հց), որը լայն տարածում ունի արդյունաբերության մեջ, հանդիսանում է լսողության վնասվածքների պատճառ, չնայած որ մարդու ականջը դրան նույնիսկ չի ընկալում: Հզոր ուլտրաձայնը ազդում է գլխուղեղի և ողնուղեղի նյարդային բջիջների վրա և առաջացնում է այրոց ականջի շրջանում և սրտխառնոց:

Ոչ պակաս վտանգավոր է ակուստիկ տատանումների ինֆրաձայնային ազդեցությունը (<16 Հց): Բավարար ինտենսիվության դեպքում դրանք ազդում են վեստիբյուլյար ապարատի վրա՝ իջեցնելով հավասարակշռությունը պահելու ունակությունը, լսելու ընկալունակությունը և առաջացնելով հոգնածություն, գրգռվածություն:

Յուրահատուկ դեր ունեն 7 Հց հաճախությամբ ինֆրաձայնային տատանումները: Եթե դրանք համընկնում են գլխուղեղի ռիթմի հետ, ապա նկատվում են ոչ միայն վերը թվարկված ախտանիշները, այլև կարող է առաջանալ ներքին արյունահոսություն: 6-ից 8 Հց հաճախությամբ ինֆրաձայնը կարող է առաջացնել արյան շրջանառության խանգարում:

Բարձր ինտենսիվությամբ աղմուկը հաճախությունների լայն տիրույթում(սկսած ինֆրաձայնից վերջացրած ուլտրաձայնով) կարող է առաջացնել գլխուղեղի և սրտի աշխատանքի խանգարումներ, շնչառական համակարգի արագության և շարժողական ակտիվության փոփոխություն: Առանձին դեպքերում աղմուկները կարող են առաջացնել վահանագեղձի չափերի փոփոխություն, արյունատար անոթների սեղմում, արյան ճնշման բարձրացում, անքնություն, հոգեկան խանգարումներ և այլն:

Աղմուկի պատճառով լսողության կորստի գնահատման համար (ISO 1999) ստանդարտների միջազգային կազմակերպությունը հաստատել է ստանդարտ: Այդ

փաստաթղթում բերվում է վնասված լսողությամբ աշխատողների սպասվող հարաբերական թիվը որպես ֆունկցիա աղմուկի էքսպոզիցիայի արժեքից:

Օրինակ աշխատողների 22%-ը հնարավոր է կկորցնեն լսողությունը, եթե նրանք 40 տարվա ընթացքում ենթարկվեն 90 Դբ մակարդակով աղմուկի ազդեցությանը (40 ժամ աշխատանքային շաբաթվա դեպքում): Գրաֆիկի կորերը կիրառելի չեն իմպուլսային կամ բարձր ինտենսիվությամբ կարճատև աղմուկների համար:

Մարդը, որը ենթարկվում է ինտենսիվ աղմուկի ազդեցությանը, միջին հաշվով ծախսում է 10-20% ֆիզիկական և նյարդահոգեբանական ջանքեր ավելին, քան ձայնամեկուսացված պայմաններում գտնվողը: Աղմկոտ արտադրություններում աշխատողների մոտ նկատվում է ընդհանուր բնույթի հիվանդությունների 10-15% աճ:

Գրականություն

1. Steganography An Art of Hiding Data, Shashikala Channalli et al /International Journal on Computer Science and Engineering Vol.1(3), 2009
2. https://en.wikipedia.org/wiki/Steganalysis
3. Generative adversarial nets. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. pp. 2672–2680, 2014.
4. Unsupervised representation learning with deep convolutional generative adversarial networks. Alec Radford, Luke Metz, and Soumith Chintala. arXiv preprint arXiv:1511.06434, 2015.
5. Generative adversarial networks for image steganography. Denis Volkhonskiy, Boris Borisenko and Evgeny Burnaev
6. https://en.wikipedia.org/wiki/Minimax
7. Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784, 2014.
8. Generative adversarial text to image synthesis. Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. arXiv preprint arXiv:1605.05396, 2016.
9. https://emerj.com/ai-glossary-terms/what-is-machine-learning/
10. https://sebastianraschka.com/Articles/2014_about_feature_scaling.html
11. https://skymind.ai/wiki/neural-network
12. https://www.kaggle.com/jessicali9530/celeba-dataset/version/2
13. https://www.techrepublic.com/article/github-the-top-10-programming-languages-for-machine-learning/
14. https://en.wikipedia.org/wiki/High-level_programming_language
15. https://en.wikipedia.org/wiki/Strong_and_weak_typing
16. https://en.wikipedia.org/wiki/Interpreted_language
17. https://en.wikipedia.org/wiki/Dynamic_programming_language

Հավելված

SGANModel-ի generator() մեթոդը

```
def generator(self):
    if self.G:
        return self.G
    self.G = Sequential()
    dropout = 0.4; depth = 128 * 4; dim = 7
    self.G.add(Dense(dim * dim * depth, input_dim=100))
    self.G.add(BatchNormalization(momentum=0.9))
    self.G.add(Activation('tanh'))
    self.G.add(Reshape((dim, dim, depth)))
    self.G.add(Dropout(dropout))

    self.G.add(UpSampling2D())
    self.G.add(Conv2DTranspose(int(depth / 2), 5, padding='same'))
    self.G.add(BatchNormalization(momentum=0.9))
    self.G.add(Activation('relu'))

    self.G.add(UpSampling2D())
    self.G.add(Conv2DTranspose(int(depth / 4), 5, padding='same'))
    self.G.add(BatchNormalization(momentum=0.9))
    self.G.add(Activation('relu'))

    self.G.add(Conv2DTranspose(int(depth / 8), 5, padding='same'))
    self.G.add(BatchNormalization(momentum=0.9))
    self.G.add(Activation('relu'))

    self.G.add(Conv2DTranspose(1, 5, padding='same'))
    self.G.add(Activation('sigmoid'))
    self.G.summary()
    return self.G
```

SGANModel-ի discriminator() մեթոդը

```
def discriminator(self):
    if self.D:
        return self.D
    self.D = Sequential(); depth = 64; dropout = 0.4
    input_shape = (self.img_rows, self.img_cols, self.channel)
    self.D.add(Conv2D(depth * 1, 5, strides=2,
                      input_shape=input_shape, padding='same'))
    self.D.add(LeakyReLU(alpha=0.2))
    self.D.add(Dropout(dropout))

    self.D.add(MaxPool2D(strides=2, padding='same'))
    self.D.add(Dropout(dropout))

    self.D.add(Conv2D(depth * 4, 5, strides=2, padding='same'))
    self.D.add(LeakyReLU(alpha=0.2))
    self.D.add(Dropout(dropout))

    self.D.add(MaxPool2D(strides=2, padding='same'))
    self.D.add(Dropout(dropout))

    self.D.add(Flatten())
    self.D.add(Dense(4096, activation="sigmoid"))
    self.D.add(Dense(1, activation="sigmoid"))
    self.D.summary()
    return self.D
```

SGANTrainer-ի train() մեթոդը

```
def train(self, initial_step, train_steps, batch_size, plot_interval):
    noise_input = None
    if plot_interval > 0:
        noise_input = np.random.uniform(-1.0, 1.0, size=[16, 100])
    for i in range(initial_step, train_steps):
        images_train = self.x_train[
                       np.random.randint(
                           0, self.x_train.shape[0],
                           size=(batch_size // 2)
                       ),:, :, :]

        noise = np.random.uniform(-1.0, 1.0, size=[batch_size // 2, 100])
        images_fake = self.generator.predict(noise)
        # images_train -> 1, images_fake -> 0
        x = np.concatenate((images_train, images_fake))
        y = np.concatenate((np.ones(batch_size // 2),
                            np.zeros(batch_size // 2)))
        d_loss = self.discriminator.train_on_batch(x, y)
```

```
        y_ones = np.ones(batch_size)
        noise = np.random.uniform(-1.0, 1.0, size=[batch_size, 100])
        gd_loss = self.gd_adversarial.train_on_batch(noise, y_ones)

        import stegano
        stegano_images = [stegano.encodeLSB(randomString(), x)
                          for x in images_train]
        # stegano_images -> 1, images_train -> 0
        x = np.concatenate((stegano_images, images_train))
        s_loss = self.steganalyser.train_on_batch(x, y)
        gs_loss = self.gs_adversarial.train_on_batch(noise, y_ones)

        log_msg = "%d: [D loss: %f, acc: %f]" \
                  % (i, d_loss[0], d_loss[1])
        log_msg = "%s  [D-G loss: %f, acc: %f]" \
                  % (log_msg, gd_loss[0], gd_loss[1])
        log_msg = "%s  [S loss: %f, acc: %f]" \
                  % (log_msg, s_loss[0], s_loss[1])
        log_msg = "%s  [S-G loss: %f, acc: %f]" \
                  % (log_msg, gs_loss[0], gs_loss[1])
        print(log_msg)
        if plot_interval > 0:
            if (i + 1) % plot_interval == 0:
                self.plot_images(save2file=True,
                                 samples=noise_input.shape[0],
                                 noise=noise_input, step=(i + 1))
```