ISSN 2181-3187

YUZNI TANISH ALGORITMLARINING QIYOSIY TAXLILI

Qodirov R (TUIT, AT kafedra, stajyor-o'qituvchi)

Annotatsiya: Har qanday yuzni aniqlash algoritmi uchun, ikki bosqich mavjud. Ulardan biri o'quv bosqichi, ikkinchisi - sinov bosqichi. O'quv bosqichida to'plamdagi barcha yuzlar topiladi va ma'lumotlar bazasida saqlanadi. Xususiyatlari standart Eigen yoki HOG bo'lishi mumkin. Sinov bosqichida probning xususiyatlari hisoblab chiqiladi. Ushbu xususiyatlar va galereyaning xususiyatlari har qanday tasniflagichga beriladi.

Kalit so'zlar: HoG, Facenet, Dlib, CNN, Cascade, pipeline,piksel, recognition.

Kirish

Agar biz tez yuzni aniqlash algoritmini xohlasak, Dlib-dan foydalanishimiz kerak. Boshqa tomondan, agar biz ko'p sonli yuzlarni aniqlash uchun algoritmni xohlasak, bizning tanlovimiz Facenet yoki Mtcnn bo'lishi mumkin.

Tezlik nuqtai nazaridan, HoG eng tezkor algoritm bo'lib ko'rinadi, undan keyin Haar Cascade tasniflagichi va CNN. Biroq, Dlibdagi CNN-lar eng aniq algoritm bo'lib qoladi. HoG juda yaxshi ishlaydi, lekin kichik yuzlarni aniqlashda ba'zi muammolar mavjud. HaarCascade tasniflagichlari umuman HoG kabi yaxshi ishlaydi.

Yuzni tanib olish bilan bog'liq bir nechta muammolar mavjud:

1. Birinchidan, rasmga qarash va undagi barcha yuzlarni topish
2. Ikkinchidan, har bir yuzga e'tibor qaratish va agar yuz g'alati tomonga burilsa yoki yomon yorug'likda bo'lsa ham, u baribir o'sha odam ekanligini tushunib olish.
3. Uchinchidan, yuzning o'ziga xos xususiyatlarini tanlay olish, uni boshqa odamlardan ajratish uchun foydalanish mumkin, masalan, ko'zlarning qanchalik kattaligi, yuzning uzunligi va hokazo.
4. Va nihoyat, odamning ismini aniqlash uchun o'sha yuzning o'ziga xosxususiyatlarini siz allaqachon bilgan barcha odamlar bilan solishtirish.

Inson sifatida sizning miyangiz bularning barchasini avtomatik ravishda va bir zumda bajarishga ulangan. Aslida, odamlar yuzlarni tanib olishda *juda yaxshi va oxir-oqibat kundalik narsalarda yuzlarni ko'rishadi.* Kompyuterlar bunday yuqori darajadagi umumlashtirishga qodir emas (*hech bo'lmaganda hali ...*), shuning uchun biz ularga ushbu jarayonning har bir bosqichini qanday qilishni alohida o'rgatishimiz kerak.

Biz pipeline ni qurishimiz kerak, unda biz yuzni tanib olishning har bir bosqichini alohida hal qilamiz va joriy qadamning natijasini keyingi bosqichga o'tkazamiz. Boshqacha qilib aytganda, biz bir nechta mashinani o'rganish algoritmlarini birlashtiramiz:

ISSN 2181-3187

	Facenet	Mtcnn	Dlib	OpenCV_DNN	OpenCV_Haar
Facenet	1812	1228	742	739	834
Mtcnn	1228	1800	858	766	1059
Dlib	742	858	1792	611	537
OpenCV_DNN	739	766	611	1770	584
OpenCV_Haar	834	1059	537	584	1605

Asosiy qism

Yuzni tanish - bosqichma-bosqich

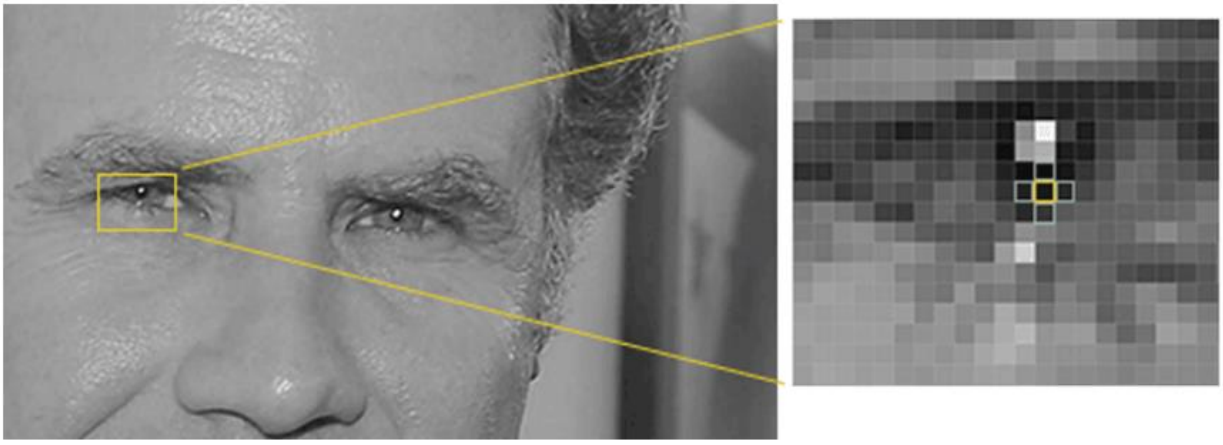
Keling, bu muammoni bir vaqtning o'zida hal qilaylik. Har bir qadam uchun biz boshqa mashinani o'rganish algoritmi haqida bilib olamiz. OpenFace va dlib dan Pythonda foydalangan holda o‘z yuzingizni aniqlash tizimini qanday yaratishni o‘rganasiz.

1-qadam: Barcha yuzlarni topish

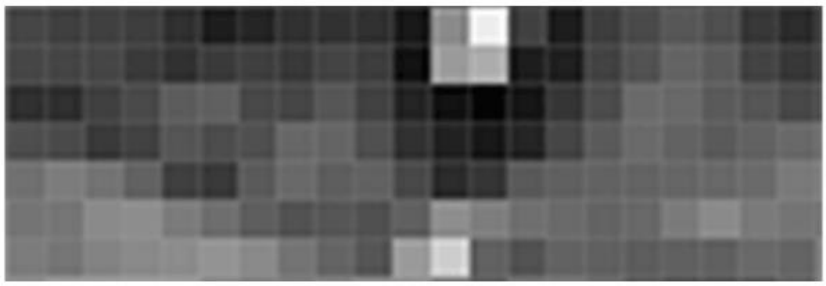
Bizning pipeline liniyasimizdagi birinchi qadam - bu *yuzni aniqlash*. Biz yo'naltirilgan gradientlarning histogrammasi yoki qisqacha ***HOG deb nomlangan algoritmdan foydalanmoqchimiz.*** Rasmdagi yuzlarni topish uchun biz rasmimizni oq-qora qilishdan boshlaymiz, chunki yuzlarni topish uchun rangli ma’lumotlar kerak emas:

Keyin rasmimizdagi har bir pikselni birma-bir ko'rib chiqamiz. Har bir piksel uchun biz uni to'g'ridan-to'g'ri o'rab turgan piksellarga qarashni xohlaymiz:

ISSN 2181-3187



Bizning maqsadimiz joriy pikselning to'g'ridan-to'g'ri atrofidagi piksellar bilan solishtirganda qanchalik qorong'i ekanligini aniqlashdir. Keyin biz tasvirning qaysi yo'nalishda qorayishini ko'rsatadigan o'qni chizmoqchimiz:

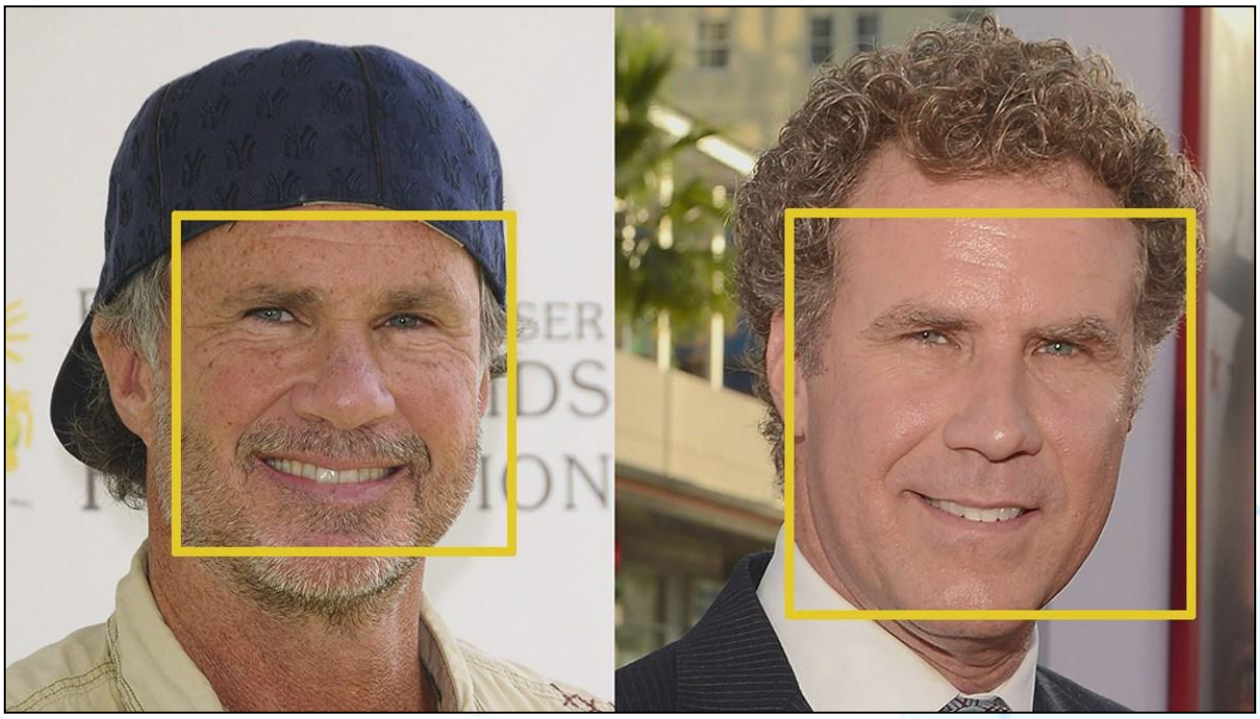


Bu bitta pikselga va unga tegib turgan piksellarga qarab, tasvir yuqori o'ng tomonga qarab qorong'ilashib

Agar siz ushbu jarayonni tasvirdagi har bir piksel uchun takrorlasangiz, har bir piksel o'q bilan almashtiriladi. Ushbu o'qlar *gradient* deb ataladi va ular butun tasvir bo'ylab yorug'likdan qorong'igacha oqimni ko'rsatadi. Bu tasodifiy ishdek tuyulishi mumkin, ammo piksellarni gradientlar bilan almashtirish uchun juda yaxshi sabab bor. Agar biz piksellarni to'g'ridan-to'g'ri tahlil qilsak, xuddi shu odamning haqiqatan ham qorong'i tasvirlari va haqiqatan ham ochiq tasvirlari butunlay boshqacha piksel qiymatlariga ega bo'ladi. Ammo yorug'lik o'zgarishi *yo'nalishini* hisobga olsak, haqiqatan ham qorong'i tasvirlar ham, haqiqatan ham yorqin tasvirlar bir xil aniq tasvir bilan yakunlanadi. Bu muammoni hal qilishni ancha osonlashtiradi! Yakuniy natija shundaki, biz asl tasvirni yuzning asosiy tuzilishini sodda tarzda aks ettiradigan juda oddiy tasvirga aylantiramiz:

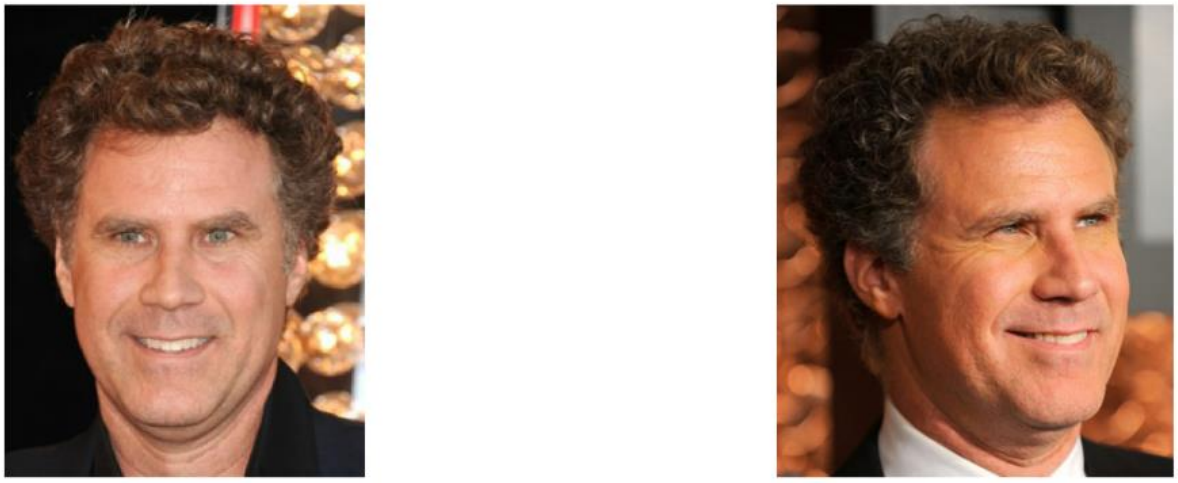
Ushbu texnikadan foydalanib, biz endi istalgan tasvirdagi yuzlarni osongina topishimiz mumkin:

ISSN 2181-3187



2-qadam: Yuzlarni suratga olish va proektsiyalash

Biz suratimizdagi yuzlarni ajratib oldik. Ammo endi biz turli yo'nalishlarga aylangan yuzlar kompyuterdan butunlay boshqacha ko'rinadigan muammoni hal qilishimiz kerak:



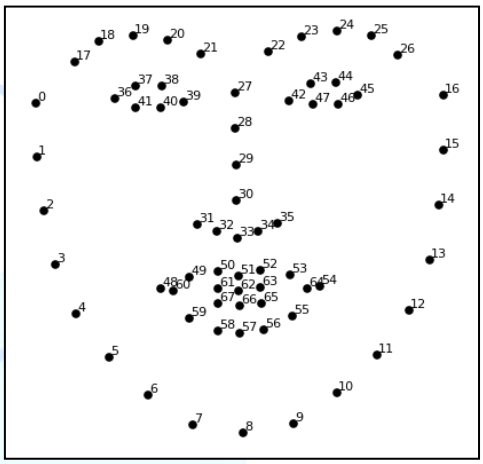
Odamlar ikkala rasm ham Uill Ferrellga tegishli ekanligini osongina tan olishadi, ammo kompyuterlar bu rasmlarni ikkita butunlay boshqa odamlar sifatida ko'rishadi.

Buni hisobga olish uchun biz har bir rasmni o'zgartirishga harakat qilamiz, shunda ko'zlar va lablar har doim tasvirdagi namuna joyida bo'ladi. Bu keyingi bosqichlarda yuzlarni solishtirishni ancha osonlashtiradi.

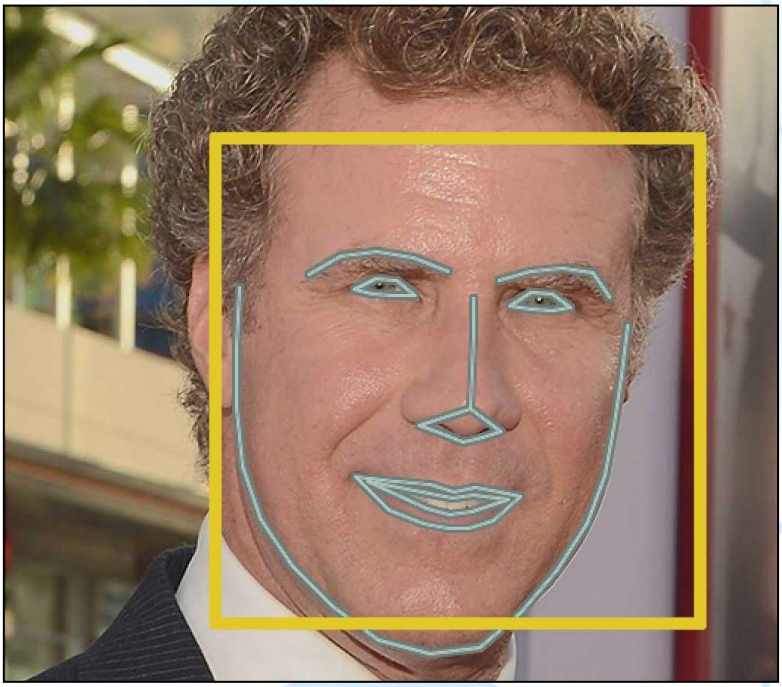
Buning uchun biz ***yuzning joylashuvini baholash (face landmark estimation)*** deb nomlangan algoritmdan foydalanamiz. Asosiy g'oya shundan iboratki, biz har bir yuzda mavjud bo'lgan 68 ta aniq nuqtani (orqaviy *belgilar* deb ataladi) - iyakning tepasi, har bir ko'zning tashqi cheti, har bir qoshning ichki cheti va hokazolarni o'ylab

ISSN 2181-3187

topamiz. Keyin biz mashinani o'rgatamiz. har qanday yuzda ushbu 68 aniq nuqtani topish uchun algoritmni o'rganish:

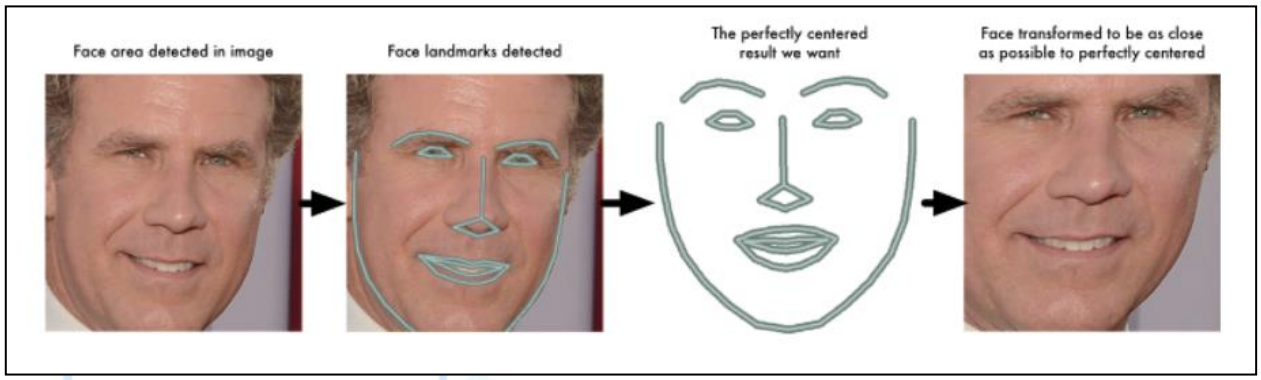


Sinov rasmimizdagi 68 ta yuz belgilarini topish natijasi:



Endi biz ko'zlar va og'iz borligini bilganimizdan so'ng, biz shunchaki tasvirni aylantiramiz, o'lchaymiz va qirqamiz , shunda ko'zlar va og'izlar iloji boricha markazlashtiriladi. Biz hech qanday ajoyib 3D burmalarni qilmaymiz, chunki bu tasvirga buzilishlarni keltirib chiqaradi. Biz faqat parallel chiziqlarni saqlaydigan aylanish va masshtab kabi asosiy tasvir o'zgarishlaridan foydalanamiz (affin o'zgarishlar deb ataladi):

ISSN 2181-3187



Endi yuz qanday o'girilgan bo'lishidan qat'i nazar, biz ko'z va og'izni markazga qo'yishimiz mumkin, tasvirda taxminan bir xil holatda. Bu bizning keyingi qadamimizni yanada aniqroq qiladi.

3-qadam: Yuzlarni kodlash

Endi biz muammoning mohiyatiga keldik - aslida yuzlarni bir-biridan ajratish. Yuzni tanib olishning eng oddiy usuli bu 2-bosqichda topilgan noma'lum yuzni bizda allaqachon teglangan odamlarning barcha rasmlari bilan to'g'ridan-to'g'ri solishtirishdir. Noma'lum yuzimizga juda o'xshab ko'rinadigan, avval belgilab qo'yilgan yuzni topganimizda, u bir xil odam bo'lishi kerak. Aslida bu yondashuvda katta muammo bor. Milliardlab foydalanuvchilari va trillionlab fotosuratlari bo'lgan Facebook kabi sayt har bir oldingi teglangan yuzni har bir yangi yuklangan rasm bilan taqqoslab bo'lmaydi. Bu juda uzoq davom etadi. Ular yuzlarni soatlarda emas, millisekundlarda taniy olishlari kerak.

Bizga kerak bo'lgan narsa har bir yuzdan bir nechta asosiy o'lchovlarni olish usulidir. Keyin biz noma'lum yuzimizni xuddi shunday o'lchab, eng yaqin o'lchovlar bilan ma'lum yuzni topishimiz mumkin edi. Misol uchun, biz har bir quloqning o'lchamini, ko'zlar orasidagi masofani, burun uzunligini va hokazolarni o'lchashimiz mumkin. Agar siz CSI kabi yomon jinoyat ko'rsatuvini ko'rgan bo'lsangiz, nima haqida gapirayotganimni bilasiz:

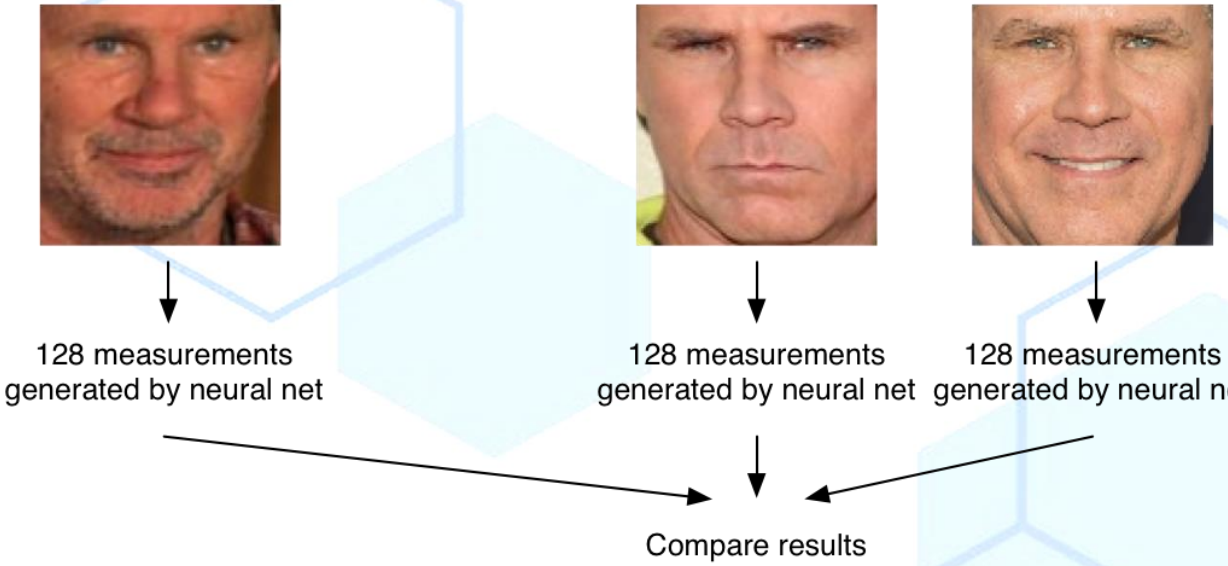
Ma'lum bo'lishicha, biz odamlarga aniq ko'rinadigan o'lchovlar (masalan, ko'z rangi) tasvirdagi alohida piksellarga qaraydigan kompyuter uchun haqiqatan ham mantiqiy emas. Tadqiqotchilar shuni aniqladilarki, eng to'g'ri yondashuv kompyuterga o'zini yig'ish uchun o'lchovlarni aniqlashga imkon berishdir. Deep learning yuzning qaysi qismlarini o'lchash muhimligini aniqlashda odamlarga qaraganda yaxshiroq ishlaydi. Ammo tarmoqni avvalgidek tasvir ob'ektlarini tanib olishga o'rgatish o'rniga, biz uni har bir yuz uchun 128 ta o'lchov yaratishga o'rgatamiz.

Tanib olish jarayoni bir vaqtning o'zida 3 ta yuz tasviriga qarash orqali ishlaydi:

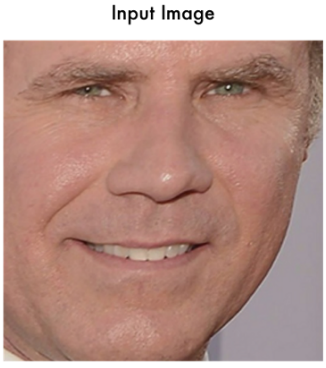
1. Ma'lum shaxsning mashg'ulot yuz tasvirini yuklash
2. Xuddi shu taniqli shaxsning boshqa rasmini yuklash
3. Mutlaqo boshqa odamning rasmini yuklash

ISSN 2181-3187

Keyin algoritm ushbu uchta tasvirning har biri uchun hozirda yaratilayotgan o'lchovlarni ko'rib chiqadi. Keyin u neyron tarmoqni biroz o'zgartiradi, shunda u №1 va №2 uchun yaratgan o'lchovlar bir oz yaqinroq ekanligiga ishonch hosil qiladi va №2 va №3 uchun o'lchovlar bir-biridan biroz uzoqroq ekanligiga ishonch hosil qiladi:



Minglab turli odamlarning millionlab tasvirlari uchun bu qadamni millionlab marta takrorlagandan so'ng, neyron tarmoq har bir kishi uchun 128 ta o'lchovni ishonchli tarzda yaratishni o'rganadi. Xuddi shu odamning har qanday o'n xil surati taxminan bir xil o'lchovlarni berishi kerak. Shunday qilib, biz o'zimiz qilishimiz kerak bo'lgan narsa, har bir yuz uchun 128 o'lchovni olish uchun oldindan o'rgatilgan tarmoq orqali yuzimiz tasvirlarini o'tkazishdir. Mana bizning sinov rasmimiz uchun o'lchovlar:



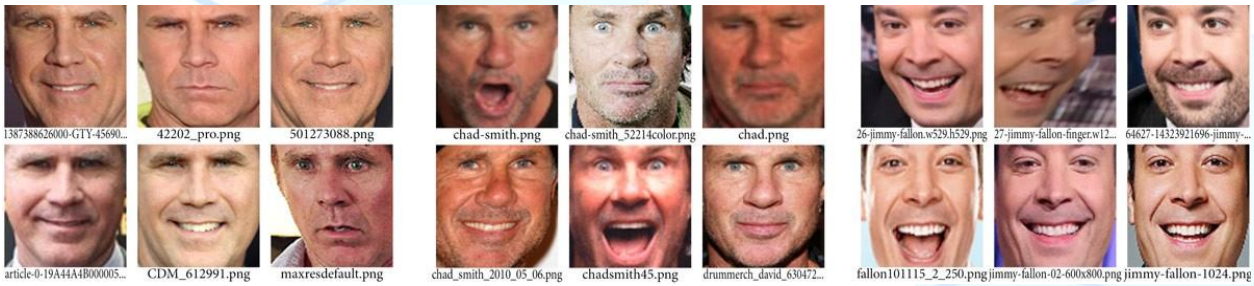
0.097496084868908	0.045223236083984	-0.1281466782093	0.032084941864014
0.12529824674129	0.060309179127216	0.17521631717682	0.020976085215807
0.030809439718723	-0.01981477253139	0.10801389068365	-0.00052163278451189
0.036050599068403	0.065554238855839	0.0731306001544	-0.1318951100111
-0.097486883401871	0.1226262897253	-0.029626874253154	-0.0059557510539889
-0.0066401711665094	0.036750309169292	-0.15958009660244	0.043374512344599
-0.14131525158882	0.14114324748516	-0.031351584941149	-0.053343612700701
-0.048540540039539	-0.061901587992907	-0.15042643249035	0.078198105096817
-0.12567175924778	-0.10568545013666	-0.12728653848171	-0.076289616525173
-0.061418771743774	-0.074287034571171	-0.065365232527256	0.12369467318058
0.046741496771574	0.0061761881224811	0.14746543765068	0.056418422609568
-0.12113650143147	-0.21055991947651	0.0041091227903962	0.089727647602558
0.061606746166945	0.11345765739679	0.021352224051952	-0.0085843298584223
0.061989940702915	0.19372203946114	-0.086726233363152	-0.022388197481632
0.10904195904732	0.084853030741215	0.09463594853878	0.020696049556136
-0.019414527341723	0.0064811296761036	0.21180312335491	-0.050584398210049
0.15245945751667	-0.16582328081131	-0.035577941685915	-0.072376452386379
-0.12216668576002	-0.0072777755558491	-0.036901291459799	-0.034365277737379
0.083934605121613	-0.059730969369411	-0.070026844739914	-0.045013956725597
0.087945111095905	0.11478432267904	-0.089621491730213	-0.013955107890069
-0.021407851949334	0.14841195940971	0.078333757817745	-0.17898085713387
-0.018298890441656	0.049525424838066	0.13227833807468	-0.072600327432156
-0.011014151386917	-0.051016297191381	-0.14132921397686	0.0050511928275228
0.0093679334968328	-0.062812767922878	-0.13407498598099	-0.014829395338893
0.058139257133007	0.0048638740554452	-0.039491076022387	-0.043765489012003
-0.024210374802351	-0.11443792283535	0.071997955441475	-0.012062266469002
-0.057223934680223	0.014683869667351	0.05228154733777	0.012774495407939
0.023535015061498	-0.081752359867096	-0.031709920614958	0.069833360612392
-0.0098039731383324	0.037022035568953	0.11009479314089	0.11638788878918
0.020220354199409	0.12788131833076	0.18632389605045	-0.015336792916059
0.0040337680839002	-0.094398014247417	-0.11768248677254	0.10281457751989
0.051597066223621	-0.10034311562777	-0.040977258235216	-0.082041338086128

Xo'sh, bu 128 raqam yuzning qaysi qismlarini aniq o'lchaydi? Ma'lum bo'lishicha, bizda hech qanday tasavvur yo'q. Bu biz uchun unchalik muhim emas. Bizni qiziqtirgan narsa shundaki, tarmoq bir odamning ikki xil suratini ko'rganda deyarli bir xil raqamlarni hosil qiladi.

ISSN 2181-3187

4-qadam: Kodlashdan odamning ismini topish

Bu oxirgi qadam aslida butun jarayondagi eng oson qadamdir. Biz qilishimiz kerak bo'lgan narsa bizning ma'lum odamlar ma'lumotlar bazasidan test tasvirimizga eng yaqin o'lchovlarga ega bo'lgan odamni topishdir. Buni har qanday asosiy mashinani o'rganish tasniflash algoritmi yordamida amalga oshirishingiz mumkin. Deep learning uchun ajoyib fokuslar kerak emas. Biz oddiy chiziqli SVM klassifikatoridan foydalanamiz, lekin ko'plab solishtirish algoritmlari ishlashi mumkin. Biz qilishimiz kerak bo'lgan narsa - yangi sinov tasviridan o'lchovlarni qabul qila oladigan va qaysi taniqli shaxs eng yaqin mos kelishini aytadigan klassifikatorni o'rgatishdir. Ushbu solishtirishni ishga tushirish millisekundlarni oladi. Klassifikatorning natijasi - bu shaxsning ismi.



Xulosa

Biz amal qilgan barcha qadamlarni qisqacha ko'rib chiqamiz:

1. Tasvirning soddalashtirilgan versiyasini yaratish uchun HOG algoritmi yordamida rasmni kodlash. Ushbu soddalashtirilgan tasvirdan foydalanib, tasvirning yuzning umumiy HOG kodlashiga o'xshash qismini topih.
2. Yuzdagi asosiy belgilarni topib, yuzning pozasini aniqlash. Ushbu belgilarni topganimizdan so'ng, ko'zlar va og'iz markazda bo'lishi uchun tasvirni burish uchun ulardan foydalanish.
3. Markazlangan yuz tasvirini yuz xususiyatlarini qanday o'lchashni biladigan neyron tarmoq orqali o'tkazish. Ushbu 128 o'lchovni saqlash.
4. O'tmishda biz o'lchagan barcha yuzlarni ko'rib chiqsak, qaysi odamning o'lchovlari bizning yuzimiz o'lchovlariga eng yaqin ekanligini bilib olish.

Foydalanilgan adabiyotlar

1. https://www.face-rec.org/algorithms/#TOP
2. https://www.researchgate.net/post/Which_is_the_best_algorithm_for_Face_Recognition
3. https://towardsdatascience.com/face-recognition-for-beginners-a7a9bd5eb5c2
4. https://www.researchgate.net/figure/Flow-Chart-of-K-means-Algorithm-32-PRINCIPAL-COMPONENT-ANALYSIS-Principal-component_fig1_323771551
5. https://www.researchgate.net/figure/The-PCA-algorithm-42-48-PCA-principal-component-analysis_fig3_258178406
6. http://fourier.eng.hmc.edu/e161/lectures/kernelPCA/node4.html
7. http://www.cs.haifa.ac.il/~rita/uml_course/lectures/KPCA.pdf