Университет:	Satbayev University
Название:	Верификация лица человека по двум фотографиям на основе глубоких нейронных сетей
Автор:	Сляхан Кузаныш
Координатор:	Мақсат Қанатов
Дата отчета:	2019-05-06 11:18:16
Коэффициент подобия № 1:	5,3%
Коэффициент подобия № 2:	3,0%
Длина фразы для коэффициента подобия № 2:	25
Количество слов:	3 627
Число знаков:	29 486
Адреса пропущенные при проверке:	
Количество завершенных проверок:	33

Ғылыми жетекші _____ М. Қанат

Қ.И. Сәтбаев атындағы Қазақ ұлттық техникалық зерттеу университеті

Мамандығы 5В070400 – Есептеу техникасы және бағдарламалық қамтамасыз ету.

Студент Слямхан Қуаныш

Тақырыбы: «Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау»

ҒЫЛЫМИ ЖЕТЕКШІНІҢ СЫН-ПІКІРІ

Дипломдық жобада күнделікті кездесетін және адам өміріне тереңінен еніп жатқан компьютерлік техника, ақпараттық жүйе мен желінің дамуы ақпарат көлемінің жылдам өңделуіне жақсы әсер етуде.

Дипломдық жобаның мақсаты тұлғаны тану алгоритмдерін зерттеп, екі суреттің көмегімен бет әлпетті верификациялау. Осы алгоритмнің негізінде рұқсат беру жүйесін құру. Себебі қазіргі таңда рұқсат беру жүйелері өте жақсы дамып келе жатыр.

Дипломдық жұмыс кіріспеден, үш бөлімнен, қорытындыдан, қолданылған әдебиеттер тізімінен тұрады. Бірінші бөлімде мәселенің қазіргі замандағы жағдайына шолу және оны талдау жасалынды. Екінші бөлімде программалау тілдерін пайдалану ерекшеліктері жобаланған. Үшінші бөлімде рұқсат беру жүйесі құрылып, және оған арналған деректер қорын басқару жүйесі зерттеуден өтіп, жүзеге асырылған.

Дипломдық жобаны қарастыра келгенде жақсы жақтарын айта кетуге болады. Жұмыс материалы жоспарлы түрде, қойылған талаптарға сай, тақырып төңірегінде жазылған. студенттің жұмысты рәсімдеудегі біліктілігін, талдаушылық қабілетін көруге болады.

Қортындылай келгенде, осы зерттеу жұмысын орындау барысында Слямхан Қуаныш Дауренұлы айтарлықтай жақсы еңбектенгені және көптеген зерттеу жұмысы жүргізгені көрінеді.

Жоба жетекшісі ретінде бұл дипломдық жобаны өз деңгейіне сәйкес деп есептей отырып Слямхан Қуаныш Дауренұлының «Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау» атты тақырыптағы дипломдық жұмысын бағалай отырып, 5В040700 – «Есептеу техникасы және бағдарламалық қамтамасыз ету» мамандығы бойынша «Техника және технология бакалавры» академиялық дәрежесін тағайындауға болады деп есептеймін.

ҒЫЛЫМИ ЖЕТЕКШІ _____ **М. Қанат**
«Программалық инженерия»
кафедрасының лекторы «30» 04 2019 жыл

№_____
«03» 05 2019 г.

Акт о внедрении

Настоящим подтверждаем, что результаты дипломного проекта Слямхан К.Д. на тему: «Верификация лица человека по двум фотографиям на основе глубоких нейронных сетей» обладают актуальностью, представляют практический интерес и были использованы при разработке системы контроля доступа для АО «НУХ «Байтерек».

Председатель Правления
АО «Институт цифровой техники и технологий»

Баймуканов Б.Н.

050000. Қазақстан Республикасы. Алматы қ., 22 В Сатпаева к-сі. Тел. +7 (727) 255-82-53
050000. Республика Казахстан, г. Алматы, ул. Сатпаева 22 В, Тел. +7 (727) 255-82-53
22 V Satpaev Street, 050000, Almaty, Kazakhstan, Tel. +7 (727) 255-82-53

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

Кафедра «Программалық инженерия»

Слямхан Қуаныш Дауренұлы

«Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау» бағдарламалық қамтамасы

ТҮСІНДІРМЕ ЖАЗБА

дипломдық жобаға

5В070400 – «Есептеу техникасы және бағдарламалық қамтамасыз ету» мамандығы

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

Программалық инженерия кафедрасы

ҚОРҒАУҒА ЖІБЕРІЛДІ
Кафедра меңгерушісі
тех. ғыл. кандидаты, доцент,
ассистен-профессор
_____ Р. Юнусов
"__" _____ 2019 ж.

Дипломдық жобаға
ТҮСІНІКТЕМЕЛІК ЖАЗБА

Тақырыбы: «Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау» бағдарламалық қамтамасы

5В070400 – «Есептеу техникасы және бағдарламалық қамтамасыз ету»

Орындаған Қ.Д. Слямхан

Ғылыми жетекші
техн. ғыл. магистрі, лектор
_____ К. Мақсат
"30" 04 2019 ж.

Алматы 2019

ҚАЗАҚСТАН РЕСПУБЛИКАСЫНЫҢ БІЛІМ ЖӘНЕ ҒЫЛЫМ МИНИСТРЛІГІ

СӘТБАЕВ УНИВЕРСИТЕТІ

Ақпараттық және телекоммуникациялық технологиялар институты

Программалық инженерия кафедрасы

5В070400 – «Есептеу техникасы және бағдарламалық қамтамасыз ету»

БЕКІТЕМІН
ПИ кафедра меңгерушісі,
тех. ғыл. кандидаты, доцент,
ассистен-профессор
_____ Р. Юнусов
"__" _____ 2019ж.

Дипломдық жобаны орындауға
ТАПСЫРМА

Білім алушы: *Слямхан Қуаныш Дауренұлы*
Тақырыбы: *«Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау» бағдарламалық қамтамасы*
Университет академиялық мәселелер жөніндегі проректорының бұйрығымен *«14» наурыз 2018ж. № 1841-б шешімімен бекітілген*
Орындалған жобаның өткізу мерзімі: *13 " мамыр 2019 ж.*
Дипломдық жобаның бастапқы мәліметтері: *Терең нейрондық желілерге негізделген адамдардың бетін екі фотосурет бойынша верификациялау бағдарламалық қамтаманы, қосымшаны қолданушылардың, өтінімді орындау қызметін атқаратын әкімші-басқарушының деректер қорына енгізу, тіркеу жүйелерін ұйымдастыру.*
Есеп – түсініктеме жазбаның талқылауға берілген сұрақтардың тізімі:
а) бет әлпетті верификациялау бағдарламалық қамтамасын құруға ашық код алгоритмдерін зерттеу;
b) қарастырылған алгоритм негізінде бағдарламалық қамтама құру;
в) пайдаланушы интерфейсін және деректер қорын жобалау, бағдарламалық қамтаманы тестілеу;
г) бағдарламалық қамтама негізінде рұқсат беру жүйесін құру;
Графикалық материалдар тізімі (міндетті суреттердің нақты көрсетілуімен):
презентацияның 18 слайды ұсынылған.
Ұсынылған негізгі әдебиеттер: *12 әдебиеттер тізімінен*

Дипломдық жобаны орындау

КЕСТЕСІ

Бөлімдердің атаулары, зерттелген мәселелердің тізімі	Ғылыми жетекшіге және кеңесшілерге ұсыну мерзімі	Ескерту
1. Дипломдық жобаның жоспарын құру	14.01.2019	орын
2. Тапсырма қойылымы және бағдарламалау ортасын таңдау	18.01.2019	орын
3. Бағдарламалық қамтамасыз жобалау және талдау.	01.02.2019	орын
4. Бағдарламаны әзірлеу	15.02.2019	орын
5. Жоба бағдарламасын тестілеу	18.03.2019	орын
6. Дипломдық жоба түсіндірме жазбасын жазу	26.04.2019	орын

Дипломдық жұмыс бөлімдерінің кеңесшілерінің аяқталған жұмысқа қойған **қолтаңбалары**

Бөлімдер атауы	Кеңес берушілер (аты-жөні, тегі, ғылыми дәрежесі, атағы)	Қолтаңба қойылған мерзімі	Қолы
Нормалық бақылаушы	Таурбекова А. А. лектор	04.05.19	
Бағдарламалық бөлім	Қалдыбеков С. Б. сениор-лектор	23.04.19	

Ғылыми жетекші _____ М. Қанат

Тапсырманы орындауға қабылдап алған студент _____ Қ. Д. Слямхан

Күні «3» қазан 2018 ж.

АҢДАТПА

Жобаның мақсаты ашық код алгоритмдерін зерттеу негізінде екі суреттің көмегімен бет әлпетті верификациялау бағдарламалық қамтамасын жазу. Компьютер көзі бағдарламалаудың дамып келе жатқан саласы, үлкен сұранысқа ие және көптеген қосымшалары бар.

Бетті тану жүйесі қазіргі уақытта өзекті болып табылады. Бұл жобада бет әлпетті табу алгоритмдері зерттеліп, оның ішінде бағыт градиенттерінің гистограммасы әдісі қолданылды. Верицикациялау негізіне терең конвективті нейрондық желі қолданылды. Жүйенің деректер қорын басқару жүйесі – PostgreSQL объектілі реляциялық деректер қоры.

Жоба нәтижесінде қойылған мақсат, міндеттер орындалды. Екі суреттің көмегімен адамдардың бет әлпетін верификациялау бағдарламасы негізінде рұқсат беру жүйесі құрылды.

АННОТАЦИЯ

Целью проекта является написание программного обеспечения для верификации лица человека по двум фотографиям на основе изучения алгоритмов с открытым исходным кодом. Компьютерный зрение – растущая индустрия программирования с большим спросом и множеством приложений.

Система распознавания лиц в настоящее время актуальна. В этом проекте изучаются алгоритмы обнаружения лиц, в том числе гистограмма направленных градиентов. Глубокая конволюционная нейроная сеть была использована на основе проекта. Система управления базами данных - объектная реляционная база данных PostgreSQL.

В результате реализации проекта цели и задачи были выполнены. Была создана система управление контролем доступа на основе верификации лица человека по двум фотографиям.

ANNOTATION

The goal of the project is to write software for verification of a person's face using two photos based on the study of open source algorithms. Computer vision is a growing programming industry with great demand and a multitude of applications.

The facial recognition system is currently relevant. This project studies face detection algorithms, including a histogram oriented gradients. A deep convolutional neural network was used based on the project. Database management system - PostgreSQL object relational database.

As a result of the project, the goals and objectives were fulfilled. An access control management system was created based on the facial verification from two photos.

МАЗМҰНЫ

КІРІСПЕ

Бұл жобаның мақсаты тұлғаны тану алгоритмдерін зерттеп, екі суреттің көмегімен бет әлпетті верификациялау. Осы алгоритмнің негізінде рұқсат беру жүйесін құру. Себебі қазіргі таңда рұқсат беру жүйелері өте жақсы дамып келе жатыр.

Мақсатқа жету міндеттері:

- коммерциялық емес ашық код алгоритмдерін зерттеу;
- дәлдігі жоғары алгоритм негізінде программалық қамтама жазу;

Зерттеу тәсілдері ретінде статистикалық модельдеу, компьютерлік кескінді өңдеу және компьютерлік көру пайдаланылды.

Нейрондық желі машиналық оқытудың әдістерінің бірі. Қазіргі таңда нейрондық желілер негізінен деректерді талдау, математика мамандарының назарында және әртүрлі салада көптеген жақсы нәтижелер көрсетуде.

Бет әлпетті тануды көптеген салада тұлғаны анықтау үшін қолдануға болады:

- рұқсат беру жүйелерінде;
- банк жүйелерінде несие алуға;
- шекаралық бақылауда;
- идентификация жүргізуге;

Соңғы жылдары рұқсат беру жүйелерінде бет әлпетті тану алгоритмдерін қолдану қарқынды дамып келе жатыр. Сол себепті мен бұл жобаны рұқсат беру жүйесінде қолдандым. Жүйенің негізі Python бағдарламалау тілінде Keras, Tensorflow, OpenCV қосымшаларының көмегімен жазылды. Жүйенің веб беті AngularJS – JavaScript бағдарламалау тілінің фреймворкінде жазылды.

1 Негізгі бөлім

1.1 Жобаның өзектілігі

Терроризмнің бүкіл әлемде жиі кездесетініне байланысты, қауіпсіздікті қамтамасыз ету үшін адамның бет әлпетін тану маңызды болып табылады. Бет әлпетті тану тапсырмасына сурет алу, алдын-ала өңдеу, бетті табу және сәйкестендіру қадамдары кіреді. Бұл жобада бет әлпетті адамдардың екі суретімен верификациялау қарастырылған.

Дербес компьютерлер мен мобильді құрылғылардың есептеу күшінің өсуіне байланысты, тұлғаны тану танымалдылыққа ие. Facebook әлеуметтік желісі пайдаланушы жүктеп салған фотосуреттердегі тұлғаларды анықтап, оларды желідегі пайдаланушымен байланыстыра алады.

Бүгінгі таңда, тұлға табу алгоритмдерінің қолдану саласы динамикалық артуда. Осындай жүйелер әуежайларға, сауда орталықтарына, тіпті үлкен қалалардың көшелерінде де орналасқан. Олардың басым бөлігі жүйені Amazon сияқты үлкен компаниялардан сатып алуда.

Суреттің көмегімен тұлға тану басқа жүйелерден бірнеше артықшылығы бар:

– арнайы қымбат құрылғыларды қажет етпейді;

– құрылғымен физикалық байланыста болудың қажеті жоқ;

Дегенмен бұл жүйелердің кемшіліктері дер бар:

– бұл әдістердің дәлдігі 100%-ды көрсетпейді;

– бет әлпетті табу алгоритмдері толығымен зерттелмеген. Яғни бет әлпетті табу барысында жүйе қателесуі мүмкін;

– жүйе өзгеру белгілеріне, жарықтың өзгеруіне, көру бұрышына байланысты қателесуі мүмкін.

Қазіргі компьютерлердің есептеу күшінің зор екендігін ескере отырып, тұлға тану алгоритмдерін жетілдіру керек. Бүгінгі таңда бірнеше ондаған бет әлпетті тану әдістері бар:

– нейрондық желілердің көмегімен;

– негізгі компонеттер әдісі;

– жергілікті екілік жүйелер;

– серпінді графикалық әдіс;

– жасырын Марков моделіне негізделген әдіс;

– сыртқы белгілердің белсенді моделі;

– белсенді пішін үлгілері.

Мен өз жобамда нейрондық желілерді қолдандым. Себебі нейрондық желілер, оның ішінде конволюциялық нейрондық желілер суреттерді өңдеуге мүмкіндік береді.

1.2 Жобаның мақсаты

Бұл жобаның мақсаты тұлғаны тану алгоритмдерін зерттеп, екі суреттің көмегімен бет әлпетті верификациялау. Осы алгоритмнің негізінде рұқсат беру жүйесін құру. Себебі қазіргі таңда рұқсат беру жүйелері өте жақсы дамып келе жатыр.

Мақсатқа жету міндеттері:

– коммерциялық емес ашық код алгоритмдерін зерттеу;

– дәлдігі жоғары алгоритм негізінде программалық қамтама жазу;

Зерттеу тәсілдері ретінде статистикалық модельдеу, компьютерлік кескінді өңдеу және компьютерлік көру пайдаланылды.

1.3 Ұқсас жобаларды шолу

Соңғы жылдары компьютер көзі үлкен танымалдылыққа ие болды. Интернет алыптары жаңа технологиялар мен инновацияларды барлығымен бөлісуде. Сол технологиялардың бірі – бет әлпетті тану. Facebook әлеуметтік желісі суреттегі сіздің достарыңызды тани алады. Facebook платформаға жүктелген әр сурет үшін автоматты түрде жасалынған тегтер ұсыныстары бар кескіндерді қолмен белгілеуге ауыстырған. Facebook суреттегі пикселдерді талдау үшін қарапайым тұлғаларды тану алгоритмін пайдаланады және оны сәйкес пайдаланушылармен салыстырады.

Тұлғаны тану қауіпсіздік саласында қолданылуда. Жеке деректерді қорғау үшін тұлғаны тану технологиясын пайдаланудың қарапайым мысалы ол қазіргі смартфондарда бет әлпеттің көмегімен телефонды ашу. Дәл осындай технологияны рұқсат беру жүйесінде қолдануға болады: адам камераға қарайды ал ол оған кіруге болады немесе болмайтынын анықтайды.

Адамдардың санын санау үшін бет әлпетті тану қолданылуы. Бет тану технологиясын кез-келген мерекеге қатысатын адамдардың санын (мысалы, конференция немесе концерт) есептеу кезінде пайдалануға болады. Қатысушыларды қолмен санаудың орнына біз қатысушылардың беттерін суретке түсіріп, келушілердің жалпы санын қайтара алатын жүйені орнатамыз. Бұл процесті автоматтандыруға және уақытты үнемдеуге көмектеседі.

Стэнфорд университетінде Microsoft компаниясының президенті Брэд Смит лекция барысында құқық қорғау органына өздерінің тұлға тану жүйесін орнатудан бас тартқандығын айтты. Оған себеп технологиялардың дамуы мемлекеттік органдарға адамдарды жаппай қадағалауға көмектесуде. Алайда бұл адамдардың құқығын бұзады. Бұл тек бастамасы, бұндай технологиялар үлкен қалалардың барлық жерлерінде орнатылып, адамдардың құқығын бұзуға дейін баруы мүмкін.

Тұлға тану бағытында дамыған елдердің бірі Қытай. Тұлға тану жүйесі Қытай елінде мектептерде орнатылған. Ол жүйе әрбір 30 секунд сайын оқушылардың көңіл күйін тексеріп отырады. Сонымен қатар бірнеше миллион халыққа тестілеу негізінде әлеуметтік рейтинг жүйесі жұмыс істейді [9].

2 Жобалау бөлімі

2.1 Унифицирленген моделдеу тілі

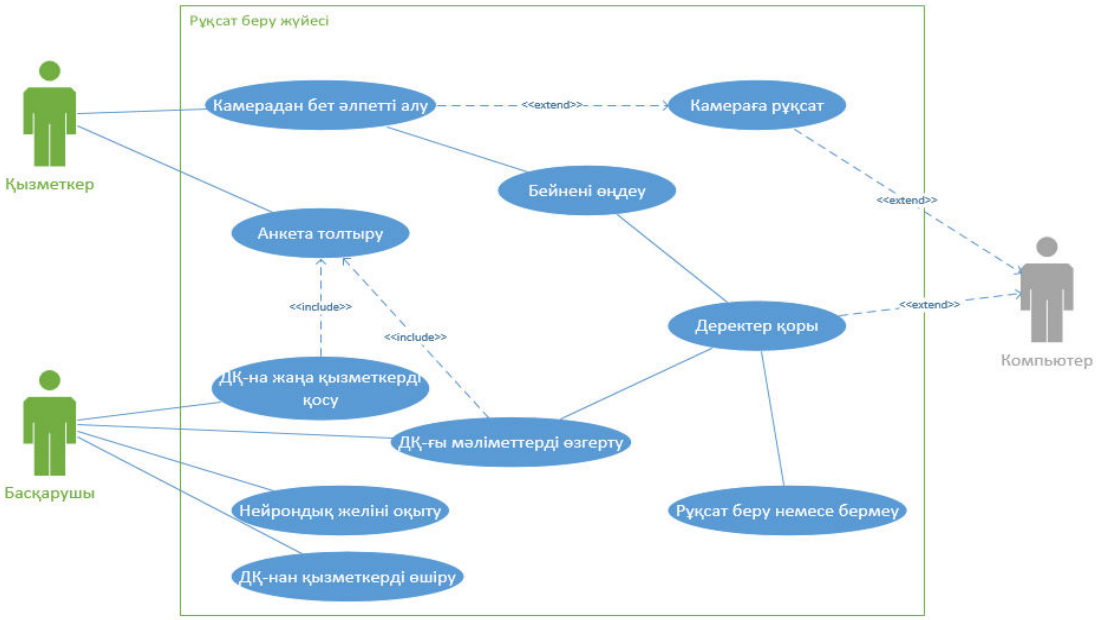
UML (ағылшын Unified Modeling Language – Unified Modeling Language) – бағдарламалық дамытуға графикалық объект моделдеу сипаттау үшін тілді, бизнес-процесстерді моделдеу, жүйелер инженерлік және ұйымдастырушылық құрылымдарды салыстыру.

2.1.1 Прецеденттер диаграммасы (use case)

Диаграммалардың бұл түрі жүйе орындайтын операциялардың тізімін жасауға мүмкіндік береді. Көп жағдайда мұндай диаграмма функционалдық диаграмма деп аталады, өйткені мұндай диаграммалардың жиынтығына негізделген жүйе талаптарының тізімі жасалады және жүйенің көптеген функциялары анықталады.

Әрбір осындай диаграмма немесе, әдетте, әрбір Пайдалану жағдайы - бұл актерлердің (актерлердің) кейінгі әрекеттер сценариінің сипаттамасы.

Диаграммалардың бұл түрі автоматтандырылған доменнің бизнес-процестерін сипаттағанда, болашақ бағдарламалық жасақтама жүйесіне қойылатын талаптарды анықтағанда пайдаланылады. Жүйенің екеуі де, тақырыптық аймақ та, орындаған тапсырмалары да көрсетіледі (2.1-сурет).

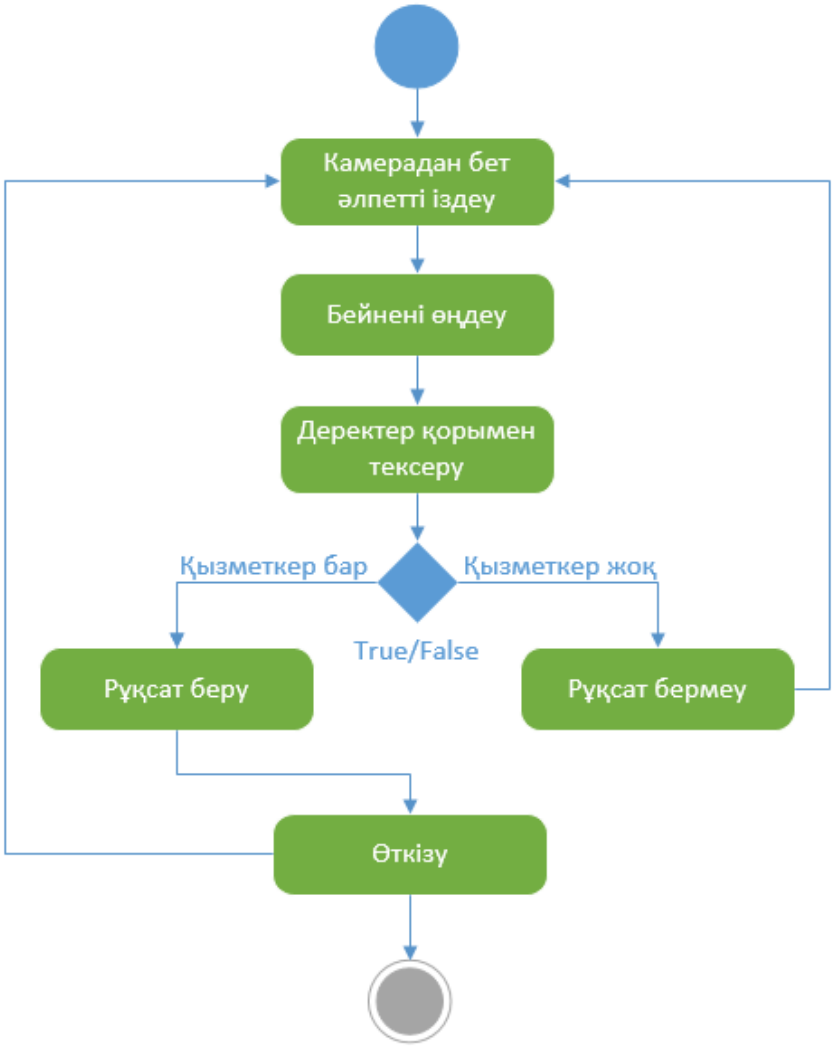


2.1-сурет – Use case диаграммасы

2.1.2 Күй диаграммасы (state machine diagram)

Белгілі бір мінез-құлыққа ие жүйенің әрбір нысаны белгілі бір мемлекеттерде болуы мүмкін, мемлекет тарапынан жүріп, объектінің мінез-құлқының сценарийін іске асыру барысында белгілі бір әрекеттерді жүзеге асырады. Нақты жүйелердің көптеген объектілерінің мінез-құлқы түпкілікті автомат теориясы тұрғысынан ұсынылуы мүмкін, яғни объектінің мінез-құлқы оның күйлерінде көрініс табады, және бұл диаграммалар осы графикті көрсетуге мүмкіндік береді. Бұл үшін диаграммалардың екі түрі қолданылады: Мемлекеттік диаграмма (мемлекеттік диаграмма) және қызмет диаграммасы (әрекет диаграммасы) Мемлекеттік схема (мемлекеттік диаграмма)

Statechart мінез-құлықтың күрделі үлгісіне ие жүйедегі объектілердің жай-күйін көрсету үшін пайдаланылады. Бұл бір мәзір элементінен қол жеткізілетін «Мемқұрылыс» машинасының екі диаграммасының бірі (2.2-сурет).

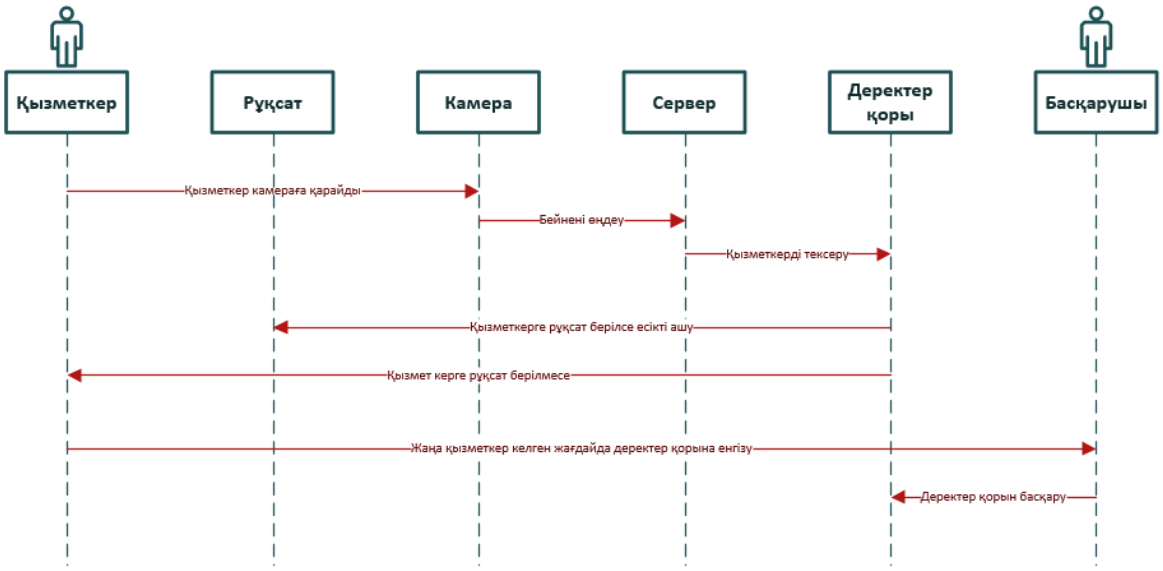


2.2-сурет – Күй диаграммасы

2.1.3 Тізбек диаграммамысы (sequence)

Жүйедегі объектілердің өзара әрекеті клиенттің объектілерінің хабарламаларын қабылдау және беру және осы хабарламаларды серверлік нысандар арқылы өңдеу арқылы жүзеге асырылады. Сонымен қатар, әртүрлі жағдайларда бірдей нысандар клиенттер ретінде де, серверлер ретінде де жұмыс істей алады. Диаграммалардың бұл түрі нысандар арасында хабарларды беру кезектілігін көрсетуге мүмкіндік береді.

Диаграмманың бұл түрі өзара әрекеттесуге бағытталмайды, басты назар формадағы хабарламаларды қабылдау/беру ретіне жатады. Объектілердің өзара байланысын көру үшін Бірлескен жұмыс диаграммасы қолданылады (2.3-сурет).



2.3-сурет – Тізбек диаграммасы

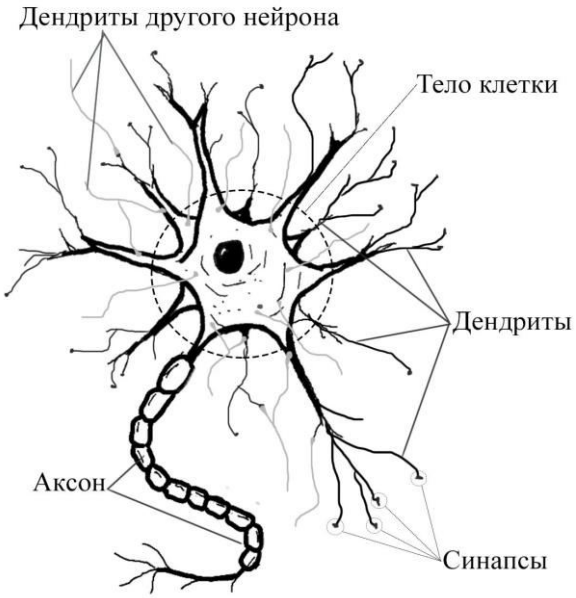
2.2 Нейрондық желілер

Жасанды нейрондық желілер оқуға қабілетті. Көптеген мысалдарды қабылдай отырып, олар өз бетінше осы заңдылықтарды таба алады және ондағы жасырын белгілерді анықтай алады. Жасанды нейрондық желілер көптеген тапсырмаларда өте жақсы нәтижелер көрсетуде.

Нейрондық желі қарапайым есептеуіш элементтерден тұрады - жасанды нейрондар, адам миының нейрондармен ұқсастығы бар. Жасанды нейрондар өзара жасанды нейрондық желі құрады.

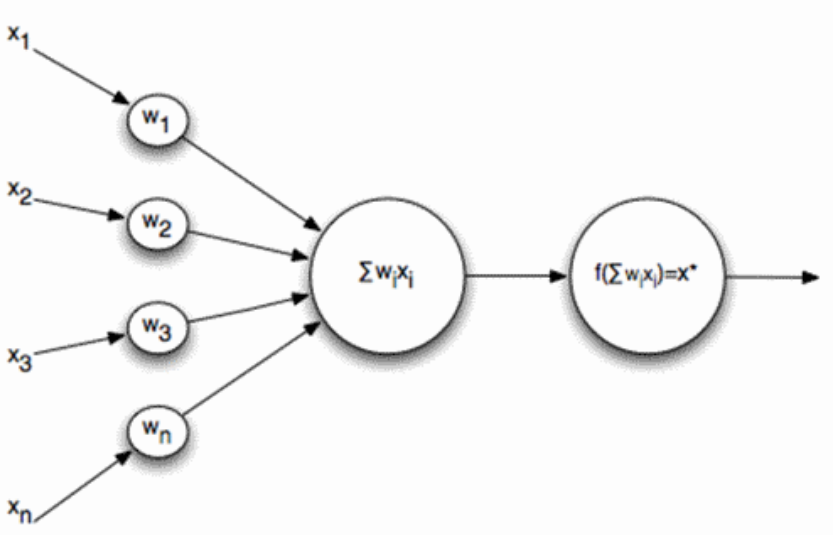
Адам миының нейрондарын сипаттайтын болсақ ол өте қарапайым көрінеді (2.4-сурет). Дендридтер синапстардан тұрады. Синапстар басқа

нейрондармен байланысады. Басқа нейрондардан синапстар арқылы сигналдар нейронның денесіне түседі. Егер сома белгілі бір шектен асса, онда нейронның жеке сигналы-спайк пайда болады, ол әрекет потенциалы. Спайк аксонға таралады және басқа нейрондарға түседі. Синапстар өз сезімталдығын өзгерте алады. Осылайша, нейрон басқа нейрондардың белсенділігінің белгілі бір комбинацияларына әрекет етуі мүмкін [2].



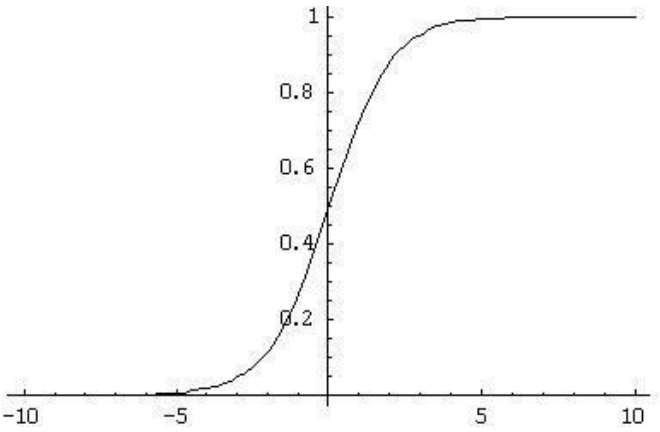
2.4-сурет – Адам миының нейроны

Жоғарыда көрсетілген нейроннан формальды жасанды нейрон ала аламыз. Нейронның бірінші болып формальды моделін 1940-ы жылдардың басында Маккалока – Питтс әзірлеген болатын (2.5-сурет).



2.5-сурет – Маккалок – Питтс формальды нейроны

Мұндай нейронның кірісіне сигналдар беріледі. Бұл сигналдар өлшеніп өзара қосылады. Бұдан әрі осы сызықтық операцияға бір активациялау функциясы орындалады, мысалы сигмоид тәріздесфукция. Көбінесе сигмоид тәріздес функция ретінде логистикалық функция қолданылады (2.6-сурет).



2.6-сурет – Логистикалық функция

Нәтижесінде бұндай нейрон шекті сумматорға айналады. Нейронның шығу сигналы 0 немесе 1-ге тең. Яғни нейрон кіріс сигналы оның синапстарына жазылған бейнеге қаншалықты ұқсас екендігін анықтайды. Өлшенген соманың мәні белгілі бір деңгейден асқанда және шекті функция бірге ауысқанда, оны нейронның ұсынылған суретті анықтағаны туралы батыл мәлімдеме айтуға болады.

2.3 Терең нейрондық желілер

Терең нейрондық желілер (deep neural networks) бүгінгі күні ең танымал машиналық оқытудың әдістерінің бірі болып табылады.

Терең нейрондық желілер альтернативті тәсілдерге қарағанда қол жетімді деректердің барлық жиынтығымен жұмыс істей алады. Оқу үрдісінде нейрондық желінің өзі деректердегі қандай белгілер маңызды және қандай белгілер керек болмайтынын анықтайды. Жасанды нейрондық желілер адамдардың болжай алмайтын белгілерін болжай алады. Сондықтан терең нейрондық желілердің көмегімен машиналық оқытудың дәстүрлі алгоритмдері орындай алмайтын міндеттерін шеше алады .

Дегенмен терең нейрондық желілерді оқыту үлкен есептеу ресурстарын талап етіледі, себебі деректердің үлкен көлемімен жұмыс істеуге тура келеді [11].

Терең нейрондық желілер қазір әртүрлі түрдегі көптеген есептерді шешу үшін қолданылады:

- белгілі суретшілердің стилінде фото суреттерді рәсімдеу (Prisma және Artisto мобильді қосымшалары);
- музыка жазу;
- әртүрлі тілдерде мәтінді автоматты түрде аудару (Google Translate, Skype Translator, Yandex Translate);
- өздігінен жүретін автокөліктер (Waymo, Google Self-Driving Car, Yandex Taxi, Uber);
- робототехника және электронды көмекшілер;
- медициналық бейнелерді талдау.

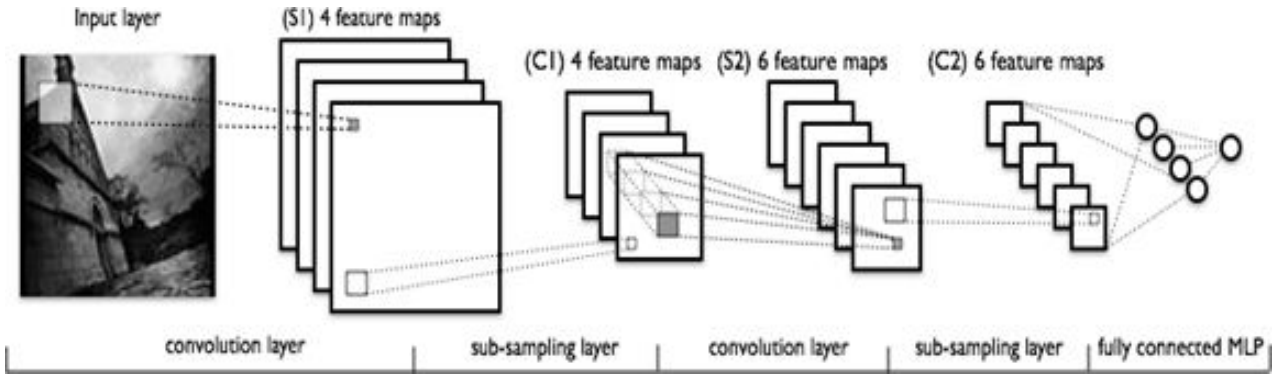
Нейрондық желілер оқыту үлгісінде оқытылады. Оқудың мәні – градиентті түсу әдісімен оңтайландыру мәселесін шешуде интернорональдық қосылыстардың салмағын түзету. Нейрондық желіні меңгеру барысында автоматты түрде экстракция, негізгі ерекшеліктер арасындағы өзара байланыстың маңыздылығын және құрылысын анықтау. Оқытылған жүйке желісі жалпылама қабілеттерге байланысты белгісіз бейнелер бойынша жаттығу кезінде алынған тәжірибені қолдануға мүмкіндік береді деп болжануда [3].

2.4 Конвективті нейрондық желілер

Адамды танудағы ең жақсы нәтижелерді немесе конволюциялық нейрондық желі (CNN) көрсетті. Табыс көп қабатты перцепронға қарағанда кескіннің екі өлшемді топологиясын ескеру мүмкіндігімен байланысты.

Қазіргі уақытта бейнелерден объектілерді табудың дәлдігі мен жылдамдығы жағынан әдістердің ішінде конвективті нейрондық желі және оның модификациялары үздік болып саналады. Сол себепті менің жобамда конвективті нейрондық желі қолданылды.

Конвективті нейрондық желілер әр түрлі қабаттардан тұрады: конвективті қабаттар, subsampling қабаттарынан, нейрондық желі қабаты – перцептроннан тұрады.



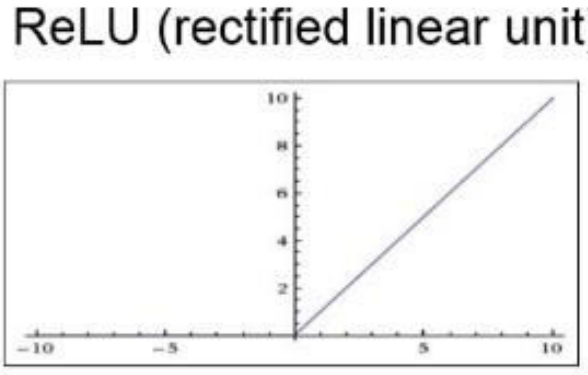
2.7-сурет – Конвективті нейрондық желілер топологиясы

Алғашқы екі қабат (convolutional, subsampling) өзара кезектесіп, Көп қабатты перцептронға кіріс векторларын қалыптастырады (2.7-сурет).

Конвективті нейрондық желілер биологялық тұрғыдағы қарапайым желілер мен көп қабатты перцептрондар арасында жақсы орта. Бүгінгі таңда бейнелерді танудың ең жақсы нәтижесі конвективті нейрондық желілердің көмегі арқылы алынада. Орташа алғанда мұндай желілер дәлдігі жасанды нейрондық желілерден 10-15%-ға асады. Конвективті нейрондық желілер терең оқытудың негізгі технлогиясы болып табылады.

Нейрондық желіні дамыту сатыларының бірі нейрондардың белсендіру функциясын таңдау болып табылады. Белсендіру функциясының түрі нейрондық желінің функционалдығын және осы желіні оқыту әдісін анықтайды. Классикалық Backpropagation (қателерді қайта тарату) алгоритмі екі қабатты және үш қабатты нейрондық желілерде жақсы жұмыс істейді, бірақ нейрондық желінің тереңдігі одан әрі артуымен ол қиындықтарға ұшырайды. Оған бір себеп болатын градиенттердің азаюы. Қате шығыс қабатының кіріс қабатына таралатындықтан, ағымдық нәтиже әр қабатта белсендіру функциясының туындысы бойынша көбейтіледі. Дәстүрлі сигмойдты белсендіру функциясының туындысы бүкіл аумақтағы бірден аз, сондықтан бірнеше қабаттан кейін қате нөлге жақын болады. Егер, керісінше, белсендіру функциясы шектелмеген туынды болса (мысалы, гиперболалық тангенс), онда ол таратылатын кезде қатенің көбеюі мүмкін, бұл оқу процесінің тұрақсыздығына әкеледі.

Бұл жұмыста белсендіру функциясы ретінде ReLU және шығыс қабаттарында maxpooling және соңғы қабат шығыс функциясы ретінде softmax қолданылады. Қазіргі уақытта ReLU – әлемдегі ең көп қолданылатын белсендіру функциясы. Өйткені ол барлық конвективті нейрондық желілерде немесе терең оқыту үшін қолданылады (2.8-сурет).

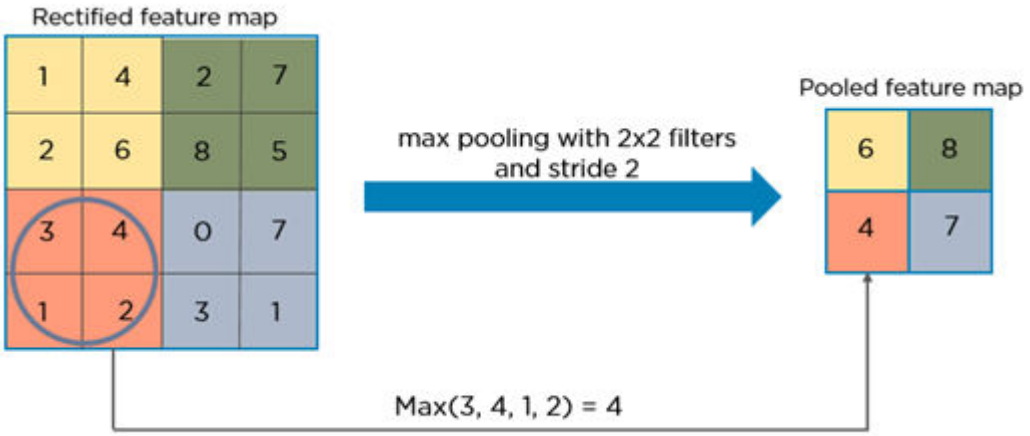


$$f(s) = \max(0, s)$$

$$f'(s) = \begin{cases} 1, & s > 0 \\ rand(0.01, 0.05), & s \le 0 \end{cases}$$

2.8-сурет – ReLU белсендіру функциясы

Maxpooling бұл 2D кіріс кеңістігінде қозғалатын терезе қолданысы. Негізгі міндеті орам материалының кеңістіктік өлшемін азайту. Терезедегі ең үлкен мән сол терезенің шығыс мәні болады. Яғни бір қабаттағы терезенің улкен мәні келесі қабаттағы бір нейронның кіріс мәні болады (2.9-сурет).



2.9-сурет – Max pooling функциясы

Softmax белсендіру функциясы мультикласстық жіктеу үшін қолданылатын логистикалық белсендіру функциясының жалпы түрі [10].

Кейінірек нейрондық желілер Facebook ұсынған DeepFace-тің дамуына байланысты танымал болды. Сәулет ерекшеліктері туралы ақпарат ашылмады.

“Oxford visual geometry group” зерттеу тобы өздерінің бет әлпетті танудың архитектурасын жариялады. Біз трансферлік оқытуды қолдана отырып, жүздеген бейнелерді тануға осы архитектураны қолдана аламыз.

Негізінде, біз беру оқыту қолдануға және vgg бет моделінің алдын ала оқытылған салмағын пайдалануға болады. Imagenet VGG нұсқасы VGGG бет моделі сияқты дерлік бірдей болса да, зерттеушілер адамдарды тану үшін таразы орнату үшін арнайы оқыту суреттерін береді.

2.5 Бет әлпетті табу алгоритмдері

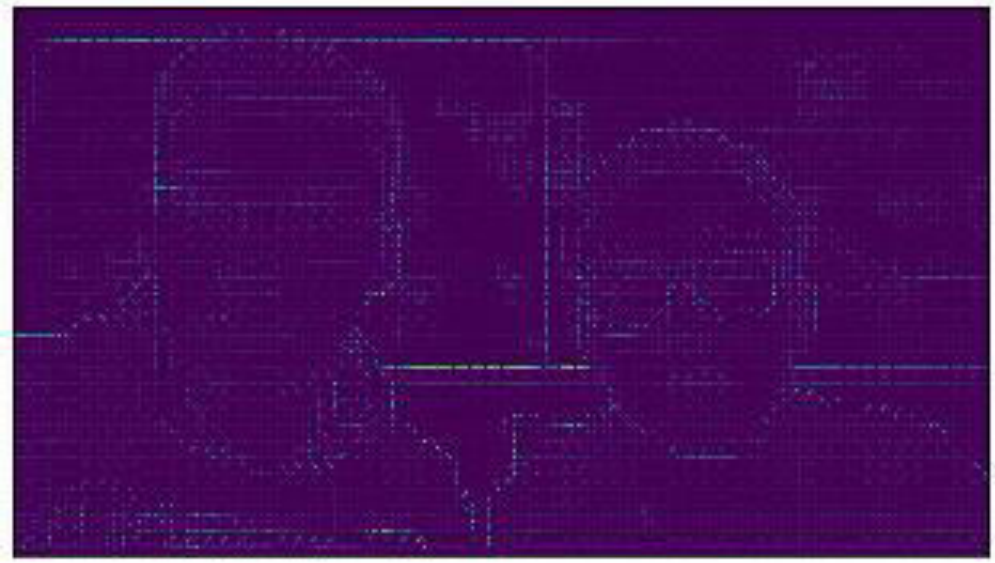
Қазіргі таңда суреттен бет әлпетті табудың алгоритмдері өте көп. Олардың кейбіреулері математикалық модельдермен түсіндірілген, мақала ретінде белгілі болса, кейбіреулері бағдарламалық қамтама ретінде белгілі (OpenCV кітапханасы, dlib кітапханасы және т.б.) [5].

Бұл бөлімде бет әлпетті табу алгоритмдерінің классификациясы қарастырылған. 2000 жылдардың. Басында Пол Виола мен Михаэль Джонс бет әлпетті жылдам табудың әдісін ойлап тапты. Алайда, қазір әлдеқайда сенімді шешімдер бар. Мен бұл жобада 2005 жылы құрылған бағыт градиенттерінің гистограммасы әдісін қолдандым (Histogram of Oriented Gradients) [12].

Бағыт градиенттерінің гистограммасы әдісі енгізілген суреттегі барлық пиксельді рет-ретімен қабылдайды. Әрбір пиксельдің жанындағы тігінен қосылатын көрші төрт писельдері қарастырылады. Негізгі мақсат әрбір пиксельдің көрші тік қосылатын пиксельдерге қарағанда қаншалықты қара екенін анықтау. Содан кейін қай бағытқа қарай пиксельдер қарайып бара жатқаны бойынша бағыт қоямыз. (2.10-сурет).

2.10-сурет – Бағыт градиенттерінің гистограмма әдісі

Егер осы процессті суреттегі әрбір пиксель үшін қайталайтын болсақ, онда шығыс мәнінде суреттегі әрбір пиксель бағытқа айналады. Бұл бағыттар градиент деп аталады және ол бағыттар суреттегі жарықтың қараңғыға қарай ағынын көрсетеді (2.11-сурет).



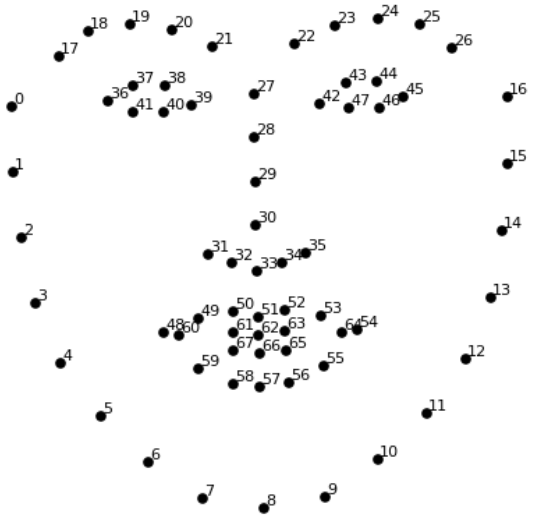
2.11-сурет – Бағыт градиенттерінің гистограмма әдісі

Нәтиже кездейсоқ нәрсе болып көрінуі мүмкін, бірақ пикселдерді градиентпен ауыстыруға өте жақсы себеп бар. Пиксельді тікелей талдау кезінде, сол адамның қараңғы және жарқын бейнелері пиксель қарқындылығының мәндеріне өте ұқсас болады. Бірақ егер біз жарықтың өзгеру бағытын ғана қарастыратын болсақ, онда қара және жарқын суреттер де бірдей бағыт ағынында ие болады. Нәтижеден беттің кескінін алу үшін оқытуға арналған адамдардың бетінен алынған бізге белгілі HOG кескініне ұқсас бөлігін тауып аламыз (2.12-сурет).

2.12-сурет – HOG кескіні

Осы әдісті қолдана отырып кез келген бейнеден адам бетін таба аламыз. Бұл әдісті қолдана отырып бейнеде адам бері әртүрлі бағытқа бұрылып тұру жағдайын қарастыру керек. Бұл мәселені шешу үшін кез келген бейнедегі адам көздері мен ерні бір орында тұратын жағдайды қарастыру керек. Ол үшін әртүрлі әдістер бар, солардың бірі 2014 жылы Вахид Кэземи мен Джозефин Салливан ұсынған [4].

Бұл әдістің түпкі идеясы әрбір адам бетінде бар 68 белгілі нүктені анықтау арқылы жүзеге асады (2.13-сурет).



2.13-сурет – Адам бетіндегі 68 нүкте

Осы әдістің нәтижесінде біз көздің және ауыздың қай жерде орналасқанын анықтай аламыз. Көз және ауызды мүмкіндігінше ортаға қарай бағыттап айналдыру, аудару әдістерін қолданамыз. Бұл әдіс аффиндік қайта құру деп аталады. Афиндік қайта құру көптеген объектілерді бұруға арналған.

Адиндік қайта құру үшін мен OpenCV кітапханасының warpAffine функциясын қолдандым.

2.6 Белгілерді есептеу және салыстыру

Соңғы қадам – бұл белгілерді есептеу және олардың жиынтығын салыстыру. Төменде ең танымал алгоритмдердің сипаттамасы берілген.

Графикаға икемді салыстыру әдісі**.** Әдістің мәні – беттердің суретін сипаттайтын серпімді графикті салыстыру. Адамдар өлшенген шеттер мен шыңдары бар графиктер ретінде ұсынылған. Тану кезеңінде референс график өзгеріссіз қалады, ал екіншісі анықтамалыққа жақсы бейімделу үшін деформацияға ұшырайды. Осы әдіске негізделген тану жүйелерінде графиктер тіктөртбұрыш тор немесе адамның антропометриялық нүктелерінен құралған құрылым болуы мүмкін. Антропометриялық нүктелердің саны 68 болған жағдайда нүктелер арасында n(n-1)/2 формуласының көмегімен мүмкін болатын 2278 қашықтық аламыз. Соның ішінен 128 ең мінезділерін қалдырамыз.

Графиктің шыңдарында атрибуттардың мәндері әдетте Габор сүзгілерінің кешенді мәндерін немесе Габор сүзгілерімен пикселдердің жарықтық мәндерін өзгерту арқылы график шыңының белгілі бір жергілікті аймағында есептелген Габор толқындарын пайдалану арқылы есептеледі.

Графиктің шеттері көрші шыңдар арасындағы қашықтыққа қарай өлшенеді. Екі графиктің арасындағы қашықтық шиектерде есептелген ерекшеліктердің мәндері мен графиктің шеттері деформациясының арасындағы айырмашылықты ескеретін белгілі бір баға деформациясының функциясы арқылы есептеледі.

Графика оның әр шыңдарын белгілі бір бағыттар бойынша белгілі бір қашықтықта бастапқы орнына қатысты және деформацияланған графиктің шыңдарындағы тән мәндері мен анықтамалық кестенің тиісті шыңдары арасындағы айырмашылықты анықтайтын шыңдардағы орнын таңдау арқылы деформацияланады. Бұл операция сілтеме мен деформацияланған графиктердің сипаттамалары арасындағы ең аз жалпы айырмашылыққа жеткенше графикалық шыңдардың әрқайсысы үшін орындалады. Деформацияланған графиктің осындай жағдайында баға деформациясының функциясы мәндік графиктің және бет кірісінің кескінінің айырмашылығы болып табылады. Бұл деформация процедурасы жүйе дерекқорында бар барлық анықтамалық тұлғалар үшін орындалуы керек. Осы тану жүйесінің жұмысының нәтижесінде деформацияның баға функциясының ең жақсы құндылығы бар стандартты аламыз [5].

Кейбір көздер фотосуреттердегі әртүрлі бет әлпеттерімен және 15 градусқа дейін айналдыру кезінде тіпті 95-97% тану дәлдігін көрсетеді. Дегенмен, осындай жүйелерді әзірлеушілер жүйе үлкен есептеуіш қуатты талап етеді деп мәлімдейді.

2.7 Іске асыру кезінде қолданылатын әдістер мен идеялар

Қолданбалы нарық ашық бастапқы коды ашық өнімдерге арналған. OpenCV кітапханасын жасаушылардың белсенді қолдауымен және кескінді өңдеудің кең мүмкіндіктері арқасында ең оңтайлы шешім болды. Өтінімнің соңғы нұсқасы статистиканы және тану модулін жинақтау модулінен тұрады.

Тану модулі OpenCV кітапханасы арқылы Python-да жазылған. Оның орындалуында:

- бет әлпетті табу үшін Хаар каскады қолданылды;
- dlib кітапханасының көмегімен суреттегі бет әлпетті өңдеу;
- есептеу және салыстыру үшін терең нейрондық желі көмегімен алдын ала оқытылған модель қолданылды.

Кіріс деректер ретінде екі сурет пайдаланылды. Шығу кезінде біз қашықтықты алдық. Бұл қашықтық аз болса, суреттердің «ұқсастығы» соғұрлым көп болады.

3 Зерттеу бөлімі

3.1 “Deep Face Recognition” архитектурасы

“Oxford Visual Geometry Group” зерттеушілер тобы өздерінің бет әлпетті танудың архитектурасын жариялады [1]. Біз трансферлік оқытудың көмегімен Deep Face Recognition архитектурасын қолдана отырып, жүздеген бейнелерді тани аламыз.

Негізгі архитектіра A, B және D арзитектураларына негізделген. Конволюциялық нейрондық желі архитектурасы 3.1-суретінде егжей-тегжейлі келтірілген [1].

layer	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
type	input	conv	relu	conv	relu	mpool	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv
name	–	conv1_1	relu1_1	conv1_2	relu1_2	pool1	conv2_1	relu2_1	conv2_2	relu2_2	pool2	conv3_1	relu3_1	conv3_2	relu3_2	conv3_3	relu3_3	pool3	conv4_1
support	–	3	1	3	1	2	3	1	3	1	2	3	1	3	1	3	1	2	3
filt dim	–	3	–	64	–	–	64	–	128	–	–	128	–	256	–	256	–	–	256
num filts	–	64	–	64	–	–	128	–	128	–	–	256	–	256	–	256	–	–	512
stride	–	1	1	1	1	2	1	1	1	1	2	1	1	1	1	1	1	2	1
pad	–	1	0	1	0	0	1	0	1	0	0	1	0	1	0	1	0	0	1

layer	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
type	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	relu	mpool	conv	relu	conv	relu	conv	softmx
name	relu4_1	conv4_2	relu4_2	conv4_3	relu4_3	pool4	conv5_1	relu5_1	conv5_2	relu5_2	conv5_3	relu5_3	pool5	fc6	relu6	fc7	relu7	fc8	prob
support	1	3	1	3	1	2	3	1	3	1	3	1	2	7	1	1	1	1	1
filt dim	–	512	–	512	–	–	512	–	512	–	512	–	–	512	–	4096	–	4096	–
num filts	–	512	–	512	–	–	512	–	512	–	512	–	–	4096	–	4096	–	2622	–
stride	1	1	1	1	1	2	1	1	1	1	1	1	2	1	1	1	1	1	1
pad	0	1	0	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

3.1-сурет – Конволюциялық нейрондық желі архитектурасы

Нейрондық желі 11 блок, 22 қабаттан тұрады және тереңдігі 37. Әрбір блок сызықтық операторлар және сызықтық емес ReLU және maxpooling белсендіру функцияларынан тұрады. Соңғы 8 блок толық қосылған (Fully Connecteed) деп аталады. Олар да конволюциондық қабат, бірақ фильтрлер өлшемі кіріс деректерінің өлшеміне тең. Бірінші екі толық қосылған қабаттардың шығысы 4096 өлшемді және соңғы толық қосылған қабатның өлшемі N=2622 немесе L=1024 оңтайландыру үшін пайдаланылатын жоғалу функцияларына байланысты. Барлық желілер кіріс өлшемі ретінде 224x224 өлшемді бет суретін қабылдайды. Бұл оңтайландыру алгоритмі тұрақтылығы үшін өте маңызды.

3.2-сурет – Нейрондық желі моделі

3.2 Деректер қоры

Бірінші деректер қоры “VGG Face” 2622 класстан тұрады. Әрбір класста тек бір адамның суреттері. Әрбір класста суреттердің URL мекен-жайы көрсетілген тексттік файл бар [7].

Екінші деректер қоры “VGG Face2” 9131 класстан тұрады. Оның ішінде 9631 класс оқытуға, 500 класс тестілеуге арналған (3.3-сурет). Бұл кең ауқымды тұлғаларды тануға арналған деректер қоры. Суреттер Google Image Search қызметінен жүктеледі және әртүрлі позада, жас ерекшелігі, жарықтығы, этностықтық, мамандықтық өзгешеліктері бар жиынтығы. Ер адамдар үлесі 60%, әйел адамдар үлесі 40%.

3.3-сурет – Тестілеуге арналған деректер қоры

3.3 Бағдарламалық қамтама жазу барысы

Бағдарламалық қамтама “Deep Face Recognition” архитектурасы негізінде жазылды. Арнайы конвективті нейрондық желі жаттығады. Нейрондық желі кіріс параметрлерінен 128 сандық вектор жасайды. Сәйкесінше нейрондық желі әртүрлі адамдарға мүмкіндігінше әртүрлі вектор жасайды. Бастапқы деректер мен нәтиже арасындағы тәуелділік өте күрделі болуы мүмкін, бірақ бұл заңдылықтарды зерттеп және анықтауға болады.

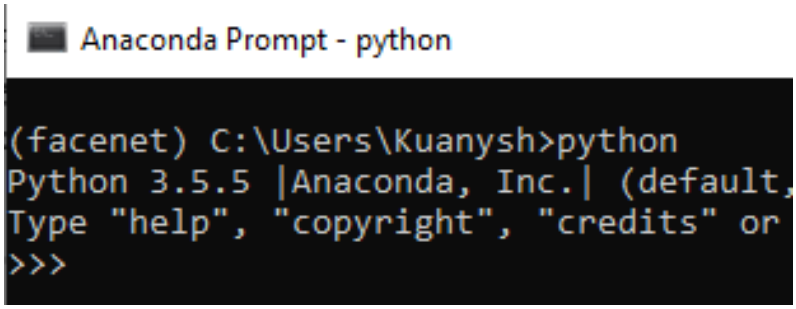
3.3.1 Python бағдарламалау тілінде жұмыс

Бағдарламалық қамтама құру барысында негізгі бағдарламалау тілі ретінде Python жоғары дәрежедегі бағдарламаллау тілі таңдалды. "Python"— ең жаңа, заманауи бағдарламаллау тіліне жатады. Python тілі қазіргі кезде ең көп таралған бағдарламаллау тілі болып саналады. Бұл тілде жасалған ең атақты

бағдарлама ол бәріміз білетін Instagram әлеуметтік желісі. Бұдан баска Python бағдарламалау тілінде BitTorrent, Ubuntu, World of Tanks секілді бағдарламалары жазылған. Python-ды қолданатын компанияларға Google, Facebook, Yahoo, NASA, Red Hat, IBM, Instagram, Dropbox, Pinterest, Quora, Яндекс, Mail.Ru т.б келтіре аламыз.

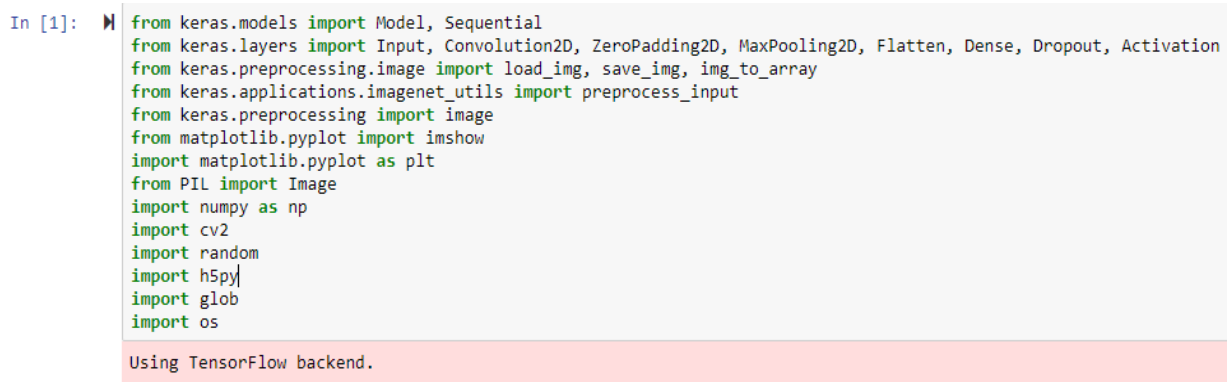
Ең алғаш рет Ван Руссом Python тілін жасауды 1898-жылдың желтоқсан айында қолға алды. Алғашында Python тілі Ameoba операциялық жүйесімен әрекет етуге қабілетті ABC бағдарламаллау тілінің ұрпағы ретінде алынған. Кеиінірек Python дамып,ең басты және маңызды бағдарламаллау тіліне айналды.Ең алғаш жаңарту Python 2.0 жаңа нұсқасы жарық көрді. Біл жаңа нұсқада көптеген функциялар мен жаңа алгоритмдер түрін көрсетті. Жаңартудан кейін Python тілінің қолданушылары арта түседі. Алайда бұл тілдің жаңарту бұнымен біткен жоқ. 2008 жылдың 3 желтоқсанында Python 3.0 жарық көреді.

Мен бұл жүйеде Anaconda3 бағдарламалау ортасында Python 3.5.5 нұсқасын қолдандым (3.4-сурет).



3.4-сурет – Python нұсқасы

Python бағдарламалау тілінде қосымша кітапханаларды қолдану өте ыңғайлы. Нейрондық желіге арналған Tensorflow, Keras кітапханаларында нейрондық жүйе моделін құруға деген мүмкіншіліктер өте көп. Жүйде қолданылған барлық қосымша кітапханалар тізімі 3.5-суретінде көрсетілген. Бағдарламалау ортасы – “Jupyter Notebook”



3.5-сурет – Жүйені құруға қолданылған Python кітапханалары

Жүйеге кіріс дерегі ретінде кез келген тексеру керек екі суретті береміз. Яғни верификациялау керек болатын адамдардың беті бар суреттер. Бірінші кезекте суреттен бет әлпетті іздейміз (3.6-сурет)

```
def normalize(image):
    image = cv2.imread(image)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    detected_faces = face_detector(image, 1)
    if (detected_faces):
        points = face_pose_predictor(image, detected_faces[0])
        landmarks = list(map(lambda p: (p.x, p.y), points.parts()))
        npLandmarks = np.float32(landmarks)
        H = cv2.getAffineTransform(npLandmarks[npLandmarkIndices],dst)
        thumbnail = cv2.warpAffine(image, H, (imgDim, imgDim))

        return thumbnail
    else:
        return None
```

3.6-сурет – Суреттен бет әлпетті іздеу коды

Бет әлпетті іздеуді dlib.get_frontal_face_detector() функциясы орындайды. Суреттен бет табылғаннан кейін беттеге 68 белгілі нүктені белгілеп аффиндік қайта құруды орындаймыз. Бет әлпет табылған суретті өндеуге функциясына (image_process()) жібереміз. Яғни суретті 224x224x3 өлшемді жиынға келтіреміз (3.7-сурет). Себебі нейрондық желіның бірінші қабаты сол өлшемдегі параметрлерді қабылдайды.

```
def verifyFace(img1, img2):

    im1 = normalizee(img1)
    im2 = normalizee(img2)

    image1_process = image_process(im1)
    image2_process = image_process(im2)

    img1_representation = descriptor.predict(preprocess_image(image1_process))[0,:]
    img2_representation = descriptor.predict(preprocess_image(image2_process))[0,:]
```

3.7-сурет – Бет әлпетті өңдеп жиын алу

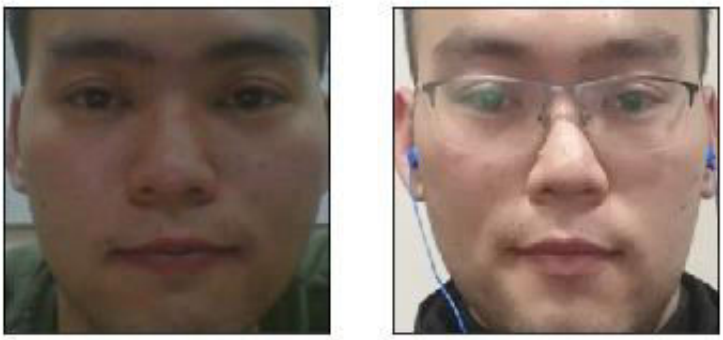
Келесі кезекте суреттен алынған жиынды дескриторға жіберіп 2622 өлшемді жиын аламыз. Жиын элементтері бөлшек мәнді тип сандардан тұрады. Яғни бейне нейрондық желі моделінің соңғы қабатынан шығады. Тексеру керек екі суреттен алған дескрипторларды салыстырып, екі сурет арасындағы косинус қашықтығын есептейміз (3.8-сурет).

```python
def findCosineSimilarity(img1_representation, img2_representation):
    a = np.matmul(np.transpose(img1_representation), img2_representation)
    b = np.sum(np.multiply(img1_representation, img1_representation))
    c = np.sum(np.multiply(img2_representation, img2_representation))
    return 1 - (a / (np.sqrt(b) * np.sqrt(c)))
```

3.8-сурет – Косинус қашықтығын есептеу

Косинус қашықтығы бөлшек мәнді сан қайтарады. Косинус қашықтығы кіші болған сайын екі суреттегі адамдардың бір екені дәл болады. Менің зерттеулерім бойынша екі суреттегі адамдарды верификациялау үшін косинус қашықтығы 0.3-тен кіші болу керек. Екі суретті верификациялау нәтижесі 3.9-суретте.

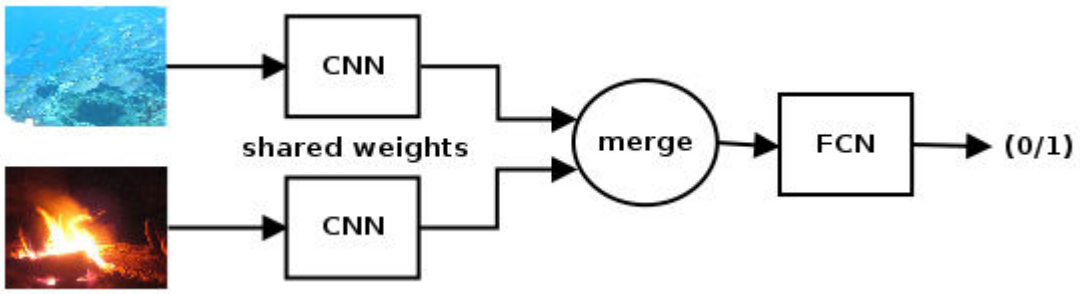
```
image #1
[ 1.4968789  -0.36789462 -1.0652597  ... -1.2693383   2.4245064
  0.99724746]

image #2
[ 1.4953235  -0.0731675  -0.39745075 ... -0.9450702   1.4391849
  0.92923844]
Косинус қашықтығы:  0.12341445684432983

Верификацияланды... Екі суретте бір адам!
```



3.9-сурет – Нәтиже

Бұл процесс бір оқпен оқыту деп аталады. Яғни біз нейрондық желіге адамдардың бірнеше суреттерін емес бір ғана сурет береміз. Адамдардың мыңдаған суретін деректер қорына сақтамаймыз. Кейбір зерттеушілер бұл процессті Сиам желісі деп те атайды. Осы бағдарламалық қамтама негізінде рұқсат беру жүйесін құрылды.

Сиам нейрондық желісі – бұл нейрондық желінің ерекше түрі (3.10-сурет). Алдымен біз қабаттарды біріктіретін және толығымен байланысқан қабаттардың түйме қабаттарының кезектілігі арқылы суретті жаттықтырамыз, нәтижесінде біз F(x1) белгілерінің векторын аламыз [8].



3.10-сурет – Сиам желісі

Бұл әдіс қолданылатына негізгі себептердің бірі - деректердің болмауы. Қазіргі заманғы машиналық оқыту алгоритмдері өте жақсы жұмыс істейді, егер деректердің саны үлкен болса.

3.3.2 Рұқсат беру жүйесіне арналған деректер қорын басқару жүйесі

Деректер қорын басқару жүйесі негізінде PostgreSQL – объектілі реляциялық деректер қорын басқарудың тегін жүйесі (ДҚБЖ) алынды. PostgreSQL AIX, түрлі BSD жүйелерін, HP-UX, IRIX, Linux, macOS, Solaris / OpenSolaris, Tru64, QNX және Microsoft Windows сияқты көптеген Unix платформаларында іске асырылады.

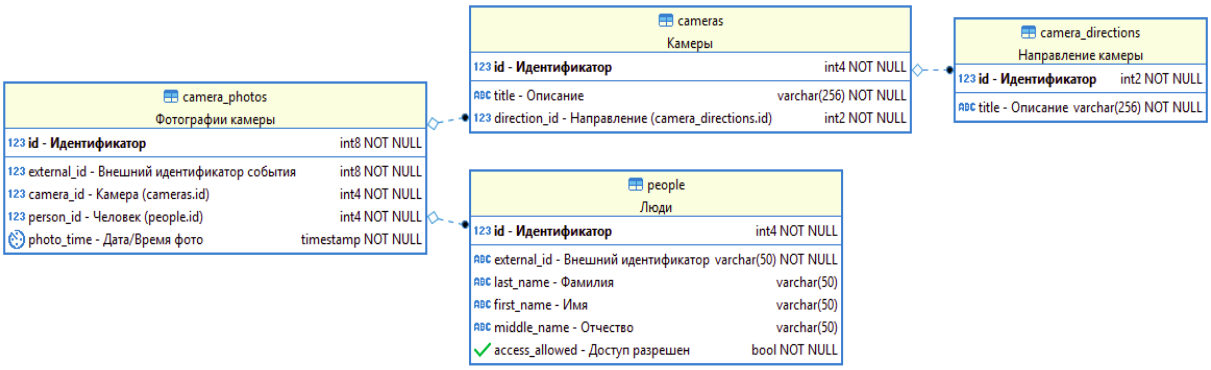
PostgreSQL SQL негізінде және SQL: 2011 стандартының көптеген мүмкіндіктерін қолдайды. PostgreSQL 9.5.3 нұсқасында 3.1-кестеде көрсетілген параметрлер бар.

3.1-кесте – PostgreSQL 9.5.3 нұсқасының параметрлері

Дерекқордың ең үлкен өлшемі	Шектеусіз
Кестенің ең үлкен өлшемі	32 Тбайт
Максималды жазу көлемі	1.6 Тб
Өріс көлемі	1 Гбайт
Кестедегі ең үлкен жазбалар	Шектеусіз
Жазбадағы ең үлкен өрістер	250-1600, өріс түрлеріне байланысты
Кестедегі ең үлкен индекстер	Шектеусіз

PostgreSQL Берклидтегі Калифорния университетінде Open-Source жобасы ретінде құрылған Postgres коммерциялық емес деректер базасына негізделген. 1986 жылы басталған Postgres-тің дамуы, Computer Associates-да сатып алынған бұрынғы Ingres жобасының жетекшісі Майкл Стоунбракермен тікелей байланысты болды. Бұл атау «Post Ingres» деп шешілді және Postgres-ті жасаған кезде бұрын жасалған көптеген жаңалықтар қолданылды.

Рұқсат беру жүйесіне деректер қорын құруға арналған ER моделі 3.11-суретте.



3.11-сурет – ER диаграмма

ER-моделі («субъект-қатынас» үлгісі) - тақырыптық аймақтың тұжырымдамалық схемасын сипаттауға мүмкіндік беретін деректер моделі.

ER үлгісі жоғары деңгейлі деректер қорын жобалау кезінде пайдаланылады. Оның көмегімен сіз негізгі субъектілерді таңдай аласыз және осы ұйымдар арасында орнатылатын қарым-қатынасты анықтай аласыз. Дерекқорды жасақтау кезінде ER үлгісі таңдалған деректер үлгісіне негізделген нақты дерекқор схемасына түрленеді.

ҚОРЫТЫНДЫ

Зерттеу барысында негізгі бағдарлама ретінде Python бағдарламалау тілі таңдалған болатын. Python нейрондық желі фреймворктарын қолдануға өте ыңғайлы заманауи бағдарламалау тілі. Қазіргі таңда нейрондық желі құруға арналған көптеген фреймворктер бар. Бұл жобада қолданылған қосымшалар ашық код жүйелерінен алынды.

Бет әлпетті табу – бағыт градиенттерінің гистограммасы әдісі дәлдігі жоғары әдістердің бірі болғанымен, бет әлпетті табу уақыты ұзақ. Бейнедегі кадрлардың пикселдері көп болған сайын бет әлпетті табу процесі көп мөлшерде уақыт аладын. Жүйе құрылу барысында дәлдік маңызды болғандықтан осы әдісті қолдандым.

Жүйені тестілеу барысында верификация дәлдігі 94%-ды көрсетті. Тестілеулер жүргізу барысында верификациялау дәлдігіне суреттердің (камераның) сапасы өте көп әсер ететіндігі байқалды. Камера орналасқан жерде жарықтың әсер ететіндігі байқалды. Жүйенің верификациялау дәлдігі жоғары болу үшін осы жағдайлар қанағаттандырылуы керек.

Жобының нәтижесінде барлық міндеттер орындалды. Атап айтқанда бет әлпетті екі суреттің көмегімен верификациялау алгоритмдері зерттеліп, ең тиімді нәтиже алынды. Сол алгоритм негізінде рұқсат беру жүйесі құрылды.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1 Omkar M. Parkhi, A. Vedaldi, A. Zisserman. Deep Face Recognition. // Сайттың электронды нұсқасы http://www.robots.ox.ac.uk/~vgg/publications/2015/Parkhi15/parkhi15.pdf http://www.robots.ox.ac.uk/~vgg/publications/2015/Parkhi15/poster.pdf

2 Самое главное о нейронных сетях. Лекция в Яндексе // Сайттың электронды нұсқасы https://habr.com/ru/company/yandex/blog/307260/

3 S. Mahapatra. Why Deep Learning over Traditional Machine Learning? // Сайттың электронды нұсқасы https://towardsdatascience.com/why-deep-learning-is-needed-over-traditional-machine-learning-1b6a99177063

4 V. Kazemi, J. Sullivan. One Millisecond Face Alignment with an Ensemble of Regression Trees // Сайттың электронды нұсқасы http://www.csc.kth.se/~vahidk/papers/KazemiCVPR14.pdf

5 Татаренков Д. А. Анализ методов обнаружения лиц на изображении // Сайттың электронды нұсқасы https://moluch.ru/archive/84/15524/

6 K. Simonyan, A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. // Сайттың электронды нұсқасы https://arxiv.org/pdf/1409.1556.pdf

7 Qiong Cao, Li Shen, Weidi Xie, Omkar M. Parkhi and Andrew Zisserman. VGGFace2: A dataset for recognising faces across pose and age // Сайттың электронды нұсқасы http://www.robots.ox.ac.uk/~vgg/publications/2018/Cao18/cao18.pdf

8 Subham Tewari. Siamese Neural Network for One-shot Image recognition-Paper Analysis. // Сайттың электронды нұсқасы https://medium.com/@subham.tiwari186/siamese-neural-network-for-one-shot-image-recognition-paper-analysis-44cf7f0c66cb

9 Рамис Ганиев. Microsoft назвала быстрое развитие технологий опасным для прав человека. // Сайттың электронды нұсқасы https://hi-news.ru/technology/microsoft-nazvala-bystroe-razvitie-texnologij-opasnym-dlya-prav-cheloveka.html

10 Jay Ricco. What is max pooling in convolutional neural networks? // Сайттың электронды нұсқасы https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks/

11 Hope Reese. Understanding the differences between AI, machine learning, and deep learning // Сайттың электронды нұсқасы https://www.techrepublic.com/article/understanding-the-differences-between-ai-machine-learning-and-deep-learning/

12 SHU Chang, DING Xiaoqing, FANG Chi. Histogram of the Oriented Gradient for Face Recognition. // Сайттың электронды нұсқасы https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6077989

А Қосымшасы
(міндетті)

Техникалық тапсырма

А.1 Кіріспе

Бүгінгі күнде бет әлпетті тану жүйелері жаңаша нұсқаларын көре отырып, заманның жетілгенің белгісін байқауға болады. Бет әлпетті тану жүйелері жоғары деңгейде жетіліп, дамуын бір ұғыммен анықтап айтуға болады, ол – технологиялардың біріктірілуі.

Қазіргі әлемде адамдарды анықтау үшін биометриялық әдістер қолданылады. Биометрия – адамның физиологиялық немесе мінез-құлық ерекшеліктеріне негізделген әдістер мен құралдарды жинау.

Қазіргі таңда ақпараттық технологиялардың дамуының ең жоғарғы сатысында. Адамдардың күнделікті өмірінде, тұрмыс тіршілікте, сонымен қатар мемлекет басқаруда да жаңа технологиялар қолданысы дамып келе жатыр.

А.1.1 Өңдеудің мақсаты мен қызметі

Қосымша пайдаланушы мен деректер қорының спецификасында жасалынды. Қосымша Jupyter Notebook, VSCode платформаларында Miniconda бағдарламасының көмегімен Python программалау тілінде жазылды.

Барлық қосымша кітапханалар Miniconda3 бағдарламасының көмегімен жазылады.

Нейрондық желі құру үшін Keras, Tensorflow фреймворктерін қолдандым. Нейрондық желі салмақтары ретінде “Deep Face Recognition” архитектурасында алдын ала оқытылған салмақтарын қолдандым. Бет әлпетті табу алгоритмі бағыт градиенттерінің гистограммасы.

А.1.2 Қолдану саласы

Қолдану аймағы – рұқсат беру жүйелерінде, банк жүйелерінде несие алуға, шекаралық бақылауда, идентификация жүргізуге. Мен бұл жобаны рұқсат беру жүйесінде қолдандым.

А Қосымшасының жалғасы

А.1 Анықтамалар, терминдер және қысқартулар

UML (ағыл. Unified Modeling Language – бірыңғай модельдеу тілі) – бұл графикалық сипаттама тілі Бағдарламаның нысаны сипатын әзірлеу үшін пайдаланылады. UML-моделі болып аталынатын графикалық символдарды пайдаланып жүйенің абстрактілік моделын құру. UML бағдарламалау тілі болып табылып, UML модельдері негізінде кодты генерациялауы мүмкін.

CNN (Convolutional Neural Networks) – конвективті нейрондық желі көрнекі суреттерді талдау үшін әдетте қолданылатын терең нейрондық желілер класы. Көп қабатты қабылдағыштар, әдетте, толығымен қосылған желілерді білдіреді, яғни бір қабаттағы әрбір нейрон келесі қабаттағы барлық нейрондарға қосылады.

ReLU – Ректификацияланған сызықтық блок терең оқыту модельдерінде ең жиі қолданылатын белсендіру функциясы болып табылады. Функция кез келген теріс енгізуді қабылдайтын болса, 0 мәнін қайтарады, бірақ кез келген оң мән үшін x осы мәнді қайтарады.

А.2 Жалпы сипаттамасы

Соңғы жылдары тұлға тану жүйесіне сұраныс өте көп. Жалпы тұлға тану жүйесін көптеген үлкен компаниялар қолдануда. Қолдану көрсеткіші бірнеше мәртеге өскен. Осы көрсеткіштер үнемі өсуде, қазіргі уақытта бұл көрсеткіш әлі өзгермеген.

Қосымшалар интернет желісі арқылы ашық код жүйелерінен жүктелініп алынады. Менің мобильді қосымшам Android платформасында жазылған қосымша.

Және қосымшаны үш тілде көру мүмкіндігі бұл қосымшаның өзіндің зор көрсеткішін жүзеге асыратын қызығушылығын алып келетін бағдарлама болып табылады. Жасалу бойынша ерекшелігін табылығандықтан керек болады.

А.2.2 Аппараттық интерфейстер

Құрылғыға қойылатын жалпы талаптар:

- есептеу қуаты жоғары есептеу машинасы;
- бейнекартаның болуы өнімділікті арттырады.

А.2.3 Программалық интерфейстер

Серверлік программалық компоненттер:
– Python;
– PostgreSQL;
– Keras, Tensorflow, Dlib, OpenCV;
– Linux операциялық жүйесі.
Клиенттік программалық компонент:
– Jupyter Notebook;
– Miniconda.

А.2.5 Жады бойынша шектеулер

Жүйені пайдалануға арналған жалпы жады бойынша талаптар шектуі 4ГБайт және одан көп.

А.2.6 Коммуникациялық интерфейстер

Бұл жүйе операциондық жүйеге тәуелсіз, ғаламторға мүмкіндігі бар веб браузерде қолдайтын кез-келген жүйеде жұмыс жасайды.

А.3 Өңдеуге талаптар

А.3.1 Функционалдық талаптар

Функционалдық талаптар негізінен бағдарламалық қамтаманың атап айтқанда нені істейтінін анықтайды, қандай тапсырмаларды шешетінін анықтайды. Бұл жүйеде тестілеуде, жұмысты қолданушы тұрғысынан істейміз. Тестілеу кезінде келесі амалдарды орындаймыз:
– бейне бақылаудан бет әлпетті табу;
– бет әлпетті өңдеуден өткізу;
– бет әлпетті тану;
– рұқсат беру/бермеу.

Б Қосымшасы
(міндетті)

Бағдарлама мәтіні

```
from keras.models import Model, Sequential
from keras.layers import Input, Convolution2D, ZeroPadding2D, MaxPooling2D, Flatten, Dense, Dropout, Activation
from keras.preprocessing.image import load_img, save_img, img_to_array
from keras.applications.imagenet_utils import preprocess_input
from keras.preprocessing import image
from matplotlib.pyplot import imshow
import matplotlib.pyplot as plt
from PIL import Image
import numpy as np
import cv2
import random
import h5py
import glob
import os

model = Sequential()
model.add(ZeroPadding2D((1,1),input_shape=(224,224, 3)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(256, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(ZeroPadding2D((1,1)))
```

Б Қосымшасының жалғасы

```
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(ZeroPadding2D((1,1)))
model.add(Convolution2D(512, (3, 3), activation='relu'))
model.add(MaxPooling2D((2,2), strides=(2,2)))

model.add(Convolution2D(4096, (7, 7), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(4096, (1, 1), activation='relu'))
model.add(Dropout(0.5))
model.add(Convolution2D(2622, (1, 1)))
model.add(Flatten())
model.add(Activation('softmax'))

from keras.models import model_from_json
model.load_weights('vgg_face_weights.h5')

def preprocess_image(image_path):
    img = preprocess_input(image_path)
    return img

def findCosineSimilarity(img1_representation, img2_representation):
    a = np.matmul(np.transpose(img1_representation), img2_representation)
    b = np.sum(np.multiply(img1_representation, img1_representation))
    c = np.sum(np.multiply(img2_representation, img2_representation))
    return 1 - (a / (np.sqrt(b) * np.sqrt(c)))

def image_process(image):
    image_resize = cv2.resize(image, (224, 224))
    image_reshape = image_resize.reshape(1,224,224,3)
    return image_reshape

descriptor = Model(inputs=model.layers[0].input, outputs=model.layers[-2].output)
```

Б Қосымшасының жалғасы

```
predictor_model = "shape_predictor_68_face_landmarks.dat"
face_detector = dlib.get_frontal_face_detector()
face_pose_predictor = dlib.shape_predictor(predictor_model)
TEMPLATE = np.float32([
    (0.0792396913815, 0.339223741112), (0.0829219487236, 0.456955367943),
    (0.0967927109165, 0.575648016728), (0.122141515615, 0.691921601066),
    (0.168687863544, 0.800341263616), (0.239789390707, 0.895732504778),
    (0.325662452515, 0.977068762493), (0.422318282013, 1.04329000149),
    (0.531777802068, 1.06080371126), (0.641296298053, 1.03981924107),
    (0.738105872266, 0.972268833998), (0.824444363295, 0.889624082279),
    (0.894792677532, 0.792494155836), (0.939395486253, 0.681546643421),
    (0.96111933829, 0.562238253072), (0.970579841181, 0.441758925744),
    (0.971193274221, 0.322118743967), (0.163846223133, 0.249151738053),
    (0.21780354657, 0.204255863861), (0.291299351124, 0.192367318323),
    (0.367460241458, 0.203582210627), (0.4392945113, 0.233135599851),
    (0.586445962425, 0.228141644834), (0.660152671635, 0.195923841854),
    (0.737466449096, 0.182360984545), (0.813236546239, 0.192828009114),
    (0.8707571886, 0.235293377042), (0.51534533827, 0.31863546193),
    (0.516221448289, 0.396200446263), (0.517118861835, 0.473797687758),
    (0.51816430343, 0.553157797772), (0.433701156035, 0.604054457668),
    (0.475501237769, 0.62076344024), (0.520712933176, 0.634268222208),
    (0.565874114041, 0.618796581487), (0.607054002672, 0.60157671656),
    (0.252418718401, 0.331052263829), (0.298663015648, 0.302646354002),
    (0.355749724218, 0.303020650651), (0.403718978315, 0.33867711083),
    (0.352507175597, 0.349987615384), (0.296791759886, 0.350478978225),
    (0.631326076346, 0.334136672344), (0.679073381078, 0.29645404267),
    (0.73597236153, 0.294721285802), (0.782865376271, 0.321305281656),
    (0.740312274764, 0.341849376713), (0.68499850091, 0.343734332172),
    (0.353167761422, 0.746189164237), (0.414587777921, 0.719053835073),
    (0.477677654595, 0.706835892494), (0.522732900812, 0.717092275768),
    (0.569832064287, 0.705414478982), (0.635195811927, 0.71565572516),
    (0.69951672331, 0.739419187253), (0.639447159575, 0.805236879972),
    (0.576410514055, 0.835436670169), (0.525398405766, 0.841706377792),
    (0.47641545769, 0.837505914975), (0.41379548902, 0.810045601727),
    (0.380084785646, 0.749979603086), (0.477955996282, 0.74513234612),
    (0.523389793327, 0.748924302636), (0.571057789237, 0.74332894691),
    (0.672409137852, 0.744177032192), (0.572539621444, 0.776609286626),
    (0.5240106503, 0.783370783245), (0.477561227414, 0.778476346951)])
TEMPLATE[:,1] = np.minimum(TEMPLATE[:,1]+0.1,1)
imgDim=224
INNER_EYES_AND_BOTTOM_LIP = [39, 42, 57]
```

Б Қосымшасының жалғасы

```
OUTER_EYES_AND_NOSE = [36, 45, 33]
landmarkIndices=INNER_EYES_AND_BOTTOM_LIP
npLandmarkIndices = np.array(landmarkIndices)
dst = imgDim * TEMPLATE[npLandmarkIndices]
dst=dst-np.array([0,imgDim*0.13],dtype=np.float32)
dst[0,1]=dst[1,1]
dst[0,0]=dst[0,0]-imgDim*0.02
dst[1,0]=dst[1,0]+imgDim*0.02
dst[2,1]=dst[2,1]+imgDim*0.02

def normalize(image):
    image = cv2.imread(image)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    detected_faces = face_detector(image, 1)
    if (detected_faces):
        points = face_pose_predictor(image, detected_faces[0])
        landmarks = list(map(lambda p: (p.x, p.y), points.parts()))
        npLandmarks = np.float32(landmarks)
        H = cv2.getAffineTransform(npLandmarks[npLandmarkIndices],dst)
        thumbnail = cv2.warpAffine(image, H, (imgDim, imgDim))

        return thumbnail
    else:
        return None

metric = "cosine"

threshold = 0.3

def verifyFace(img1, img2):

    im1 = normalize(img1)
    im2 = normalize(img2)

    image1_process = image_process(im1)
    image2_process = image_process(im2)

    img1_representation = descriptor.predict(preprocess_image(image1_process))[0,:]
    img2_representation = descriptor.predict(preprocess_image(image2_process))[0,:]

    if metric == "cosine":
        print('image #1')
```

Б Қосымшасының жалғасы

```
        print(img1_representation)
        print(len(img1_representation))
        print()
        print('image #2')
        print(img2_representation)
        cosine_similarity = findCosineSimilarity(img1_representation,
img2_representation)
        print("Косинус қашықтығы: ",cosine_similarity)
        print()

        if cosine_similarity < threshold:
            print("Верификацияланды... Екі суретте бір адам!")
        else:
            print("Верификацияланбады! Екі суретте әртүрлі адам!")

    f = plt.figure()
    f.add_subplot(1,2, 1)
    plt.imshow(im1)
    plt.xticks([]); plt.yticks([])
    f.add_subplot(1,2, 2)
    plt.imshow(im2)
    plt.xticks([]); plt.yticks([])
    plt.show(block=True)
    print("-----------------------------------------")

verifyFace("321.jpg", "12.jpg")
```

Формат	Зона	Поз.	Обозначение	Наименование	Кол.	Приме-чание

					Дипломный проект	Лист
						41