әл-Фараби атындағы Қазақ ұлттық университеті

ӘӨЖ 004.352.243(043) Қолжазба құқығы негізінде

**ЕЛЕУСИНОВ АРМАН ИНАБАТҰЛЫ**

# Қазақ тілінде қолжазба мәтіндерді тану және машиналық оқыту алгоритмдері мен ақпараттық моделін құру

6D075100 – Информатика, есептеу техникасы және басқару

Философия докторы (PhD)
дәрежесін алу үшін дайындалған диссертация

Ғылыми жетекшілер:
PhD, доцент міндетін атқарушы
Бурибаев Ж.А.

PhD, профессор Миачи Тайзо
Токай университеті, Жапония

Қазақстан Республикасы
Алматы, 2025

**МАЗМҰНЫ**

**НОРМАТИВТІ СІЛТЕМЕЛЕР 3**
**БЕЛГІЛЕУЛЕР МЕН ҚЫСҚАРТУЛАР 4**
**КІРІСПЕ 5**
**1 ҚОЛЖАЗБА МӘТІНДІ ТАНУДЫҢ ЕСЕБІ МЕН АЛГОРИТМДЕРІ 10**
1.1 Қазақ тілінде қолжазба деректер жинағын құру және өңдеу әдістері 10
1.1.1 Қолжазба мәтіндер деректер жинағына шолу 10
1.1.2 Қазақ қолжазба мәтіндердің жинағын құру кезеңдері 12
1.1.3 Деректер жинағын аннотациялау және өңдеу 14
1.2 Алдын-ала өңдеу әдістерінің мәтінді танудағы маңыздылығы 19
1.2.1 Бейнені кеңістіктік сүзу операциялары 22
1.2.2 Гистограмманы өңдеу 25
1.2.3 Шуды жою 26
**2 ҚАЗАҚ ҚОЛЖАЗБА МӘТІНДЕРІН ТАНУДЫҢ МАШИНАЛЫҚ ОҚЫТУ АЛГОРИТМІН ҚҰРУ 28**
2.1 Қолжазба мәтіндерді сегменттеу әдістері 28
2.1.1 Жолдар мен сөздерді сегменттеу алгоритмі 29
2.1.2 Сегменттеу нейрондық желілерін оқыту және бағалау 33
2.2 Қолжазба мәтінді тану алгоритмдеріне шолу 36
2.3 Қолжазба таңбаларды классификациялау 40
2.3.1 Нейрондық желіні оқыту 41
2.3.2 Әріптерді классификациялау моделін бағалау 44
2.4 Деректерді жасанды көбейтудің GAN әдісі 45
2.4.1 GAN модельдері үшін қолданылатын шығын функциялары 46
2.4.2 Генеративті қарсылас желілер (GAN) алгоритмдерін бағалау метрикалары 47
2.4.3 GAN оқыту тәжірибелері 49
2.5 Сөзді тану алгоритмдері 52
2.5.1 Нейрондық желілерді оқыту және нәтижелері 63
**3 ЖҮЙЕНІҢ АҚПАРАТТЫҚ МОДЕЛІН ҚҰРУ ЖӘНЕ ЖҮЗЕГЕ АСЫРУ 68**
3.1 Қолжазбаны тану жүйесінің архитектуралық компоненттері 68
3.1.1 Модельдерді Nvidia Triton платформасында әзірлеу 68
3.1.2 Жүйенің нәтижелерін тестілеу және бағалау 72
3.2 Құжаттардан қажетті ақпаратты іздеу жүйесін әзірлеу 74
**ҚОРЫТЫНДЫ 76**
**ПАЙДАЛАНҒАН ӘДЕБИЕТТЕР ТІЗІМІ 77**
ҚОСЫМША А – Авторлық куәлік 82
ҚОСЫМША Б – Қолжазба мәтінді тану алгоритмінің бағдарлама листингі 83

## НОРМАТИВТІ СІЛТЕМЕЛЕР

Бұл диссертацияда келесі стандарттарға сәйкес сілтемелер қолданылды:

ҚР МЖМБС 5.04.034 - 2011 “Қазақстан Республикасының Мемлекеттік жалпыға міндетті білім беру стандарты. Жоғары оқу орнынан кейінгі білім. Докторантура”. Негізгі ережелер ҚР білім және ғылым министрімен бекітіліген. “17” маусым 2011 ж. №261. Астана, 2011.

“Диссертацияларды және авторефераттарды рәсімдеу бойынша нұсқаулық”, ҚР БҒМ, Жоғары аттестаттау комитеті, Алматы, 2004.

МЕСТ 7.1-2003. Библиографиялық жазба.

## БЕЛГІЛЕУЛЕР МЕН ҚЫСҚАРТУЛАР

АЖ – ақпараттық жүйе
OCR – оптикалық белгілерді тану
БК – бақылау комитеті
ЖИ – жасанды интеллект
CER – Character Error Rate
WER – Word Error Rate
CNN – Convolutional Neural Network
CRNN – Convolutional Recurent Neural Network
CTC – Connectionist Temporal Classification
DAS – Document Analysis Systems
FCN – Fully Convolutional Network
FPR – False Positive Rate
GMM – Gaussian Mixture Models
GRCL – Gated Recurrent Convolutional Layers
GRU – Gated Recurrent Units
LSTM – Long Short-Term Memory
HMM – Hidden Markov Models
HOG – Histograms of Oriented Gradients
HTR – Handwritten Text Recognition
GAN – Generative Adversarial Network

# КІРІСПЕ

Компьютерлік көру (КК) – компьютерлер мен жүйелерге цифрлық кескіндерден, бейнелерден және басқа да визуалды кірістерден мағыналы ақпаратты алуға және сол ақпарат негізінде әрекеттер немесе ұсыныстар жасауға мүмкіндік беретін жасанды интеллект саласы (ЖИ). Егер ЖИ компьютерлерге ойлауға мүмкіндік берсе, компьютерлік көру оларға көруге, бақылауға және түсінуге мүмкіндік береді. Қазіргі таңда КК көп салаларда кеңінен қолданыс тапты. Олардың негізгілері:

- Медициналық бейнелерді тану: науқастарды диагностикалауға және емдеуге көмектесу үшін рентген немесе КТ сияқты медициналық кескіндерді талдау үшін компьютерлік көруді пайдалану
- Бетті және саусақ ізін тану және биометрия: жеке басын тексеру және басқа қолданбалар үшін саусақ іздері сияқты бірегей физикалық сипаттамаларды талдау және тану үшін компьютерлік көруді пайдалану
- Таңбаларды оптикалық тану (OCR): суреттерден немесе сканерленген құжаттардан мәтінді тану және шығару мүмкіндігі
- Автокөлік қауіпсіздігі: жолдағы кедергілер мен басқа қауіптерді анықтау және оларға әрекет ету үшін жүргізушіге көмек көрсету жүйелерінде және автономды көліктерде компьютерлік көруді пайдалану
- Бақылау жүйелері: қауіпсіздік және бақылау мақсатында бейне жазбаларды талдау үшін компьютерлік көруді пайдалану

Осылардың ішінде мәтінді тану саласында зерттеулер жарты ғасырдан астам уақыт бойы жүргізіліп келеді. Қазіргі уақытта көптеген ұйымдар өнімділік пен тиімділікті арттыру үшін адамның өзара әрекеттесуін болдырмау үшін мәтінді оптикалық тану жүйелеріне сүйенеді [1, 2].

Қолжазба мәтіндерді тану информатика мен инженерлік дамудың алғашқы кезеңінен бастап маңызды зерттеу саласы ретінде пайда болды. Сонымен қатар, біздің қағаз құжаттарымызды автоматты түрде оқып, оларды сандық форматқа ауыстырудың қажеттілігі мен ыңғайлылығы әрдайым осы ғылым саласын алға жылжытты. Ғылыми ортада да, өндірісте де цифрлық құжаттарды өңдеуге бағытталған зерттеулер мен өнімдер әзірленуде. Сонымен қатар, қағазсыз қоғамды құруға бағытталған үлкен күш-жігерге қарамастан, бүкіл әлемде ақпаратты өңдеу, іздеу және сақтау мақсатында қағаз жүзінде көптеген құжаттар күн сайын жасалуда және өңделуде.

Қолмен жазылған мәтінді автоматты түрде тану деректердің үлкен көлемін өңдеуді қажет ететін өндіріс салаларында өте пайдалы болуы мүмкін. Мысалы, конверттегі мекен-жайлар мен индекстерді тану, банктік чектердегі сомаларды тіркеу, құжаттарды талдау және қолтаңбаларды тексеру, баспахана жұмыстарында және т.б. Соңғы ғылыми зерттеулерде мәтіндерді тану айтарлықтай прогреске қол жеткізілді, бірақ қолмен жазылған мәтінді тану адамның жұмысына әлі жақындай алмайды. Үлкен қиындықтар уақыт бойынша жазудың өзгеруінен, кейбір жазу стильдердің бір-біріне ұқсастығынан және әр түрлі жазу формалары мен жазу стильдерінің шексіз алуан түрлілігінен

туындайды. Сонымен қатар, мәтіндік кескіннің сапасының төмендігі, фондық шудың және әр түрлі бұрмаланулардың (мысалы, нашар жазылған немесе қабаттасқан жазулардың) болуы мүмкін. Сондықтан, қолжазбаны тану әлі де зерттеуді қажет ететін және жаңа әдістер үшін ашық әрі қызықты бағыт болып табылады.

Қолжазба мәтінді тану процесі жазбаша ақпараттың қабылдануына қарай өңделеді. Таңбалар туралы ақпарат алудың ең кең тараған екі әдісі бар: 1) қолмен жазылған құжаттар болған кезде және сканерлер немесе камералар көмегімен суреттері алынған кезде, бұл процесс «оффлайн тану» деп аталады. Бұл сценарийде бізде кескіннің қарқындылығы туралы ғана ақпарат бар, яғни кодталған суреттегі әрбір пиксельдің мәні; 2) жазба декарттық координаттарды және жазу процесінің өзіне тән ақпаратты, мысалы, соққы жылдамдығы, қаламның қысымы немесе тректің реті сияқты түсіруге қабілетті құрылғыларда орындалған кезде, бұл процесс «онлайн тану» деп аталады. Әдетте, ақпаратты онлайн режимінде тану әдістері оффлайн әдістерге қарағанда жақсы нәтиже береді.

Қазіргі таңда әр түрлі тілде жазылған қолжазба мәтіндерді тануға арналған бағдарламалық шешімдер саны көбеюде. Олардың ішінде кеңінен таралғандары ABBYY FineReader, Tesseract, Google Cloud Vision, Yandex Vision, Microsoft Azure Read, Microsoft OneNote, Online OCR. Бірақ та бұлардың ішінде ешқайсысында қазақ тілінде жазылған мәтінді тануға мүмкіндік жоқ.

**Зерттеу тақырыбының өзектілігі.** Қазақстан Республикасы Президентінің Архиві Қазақстанның қазіргі саяси тарихы бойынша бірегей құжаттардың сақтаушысы болып табылады. 2020 жылдың 1 қаңтарындағы жағдай бойынша мұрағатта 1918-2008 жылдардағы 830 мыңнан астам сақтау бірлігі бар, оның ішінде қолжазба қағаз түріндегі сақталған саны 762 мың [3].

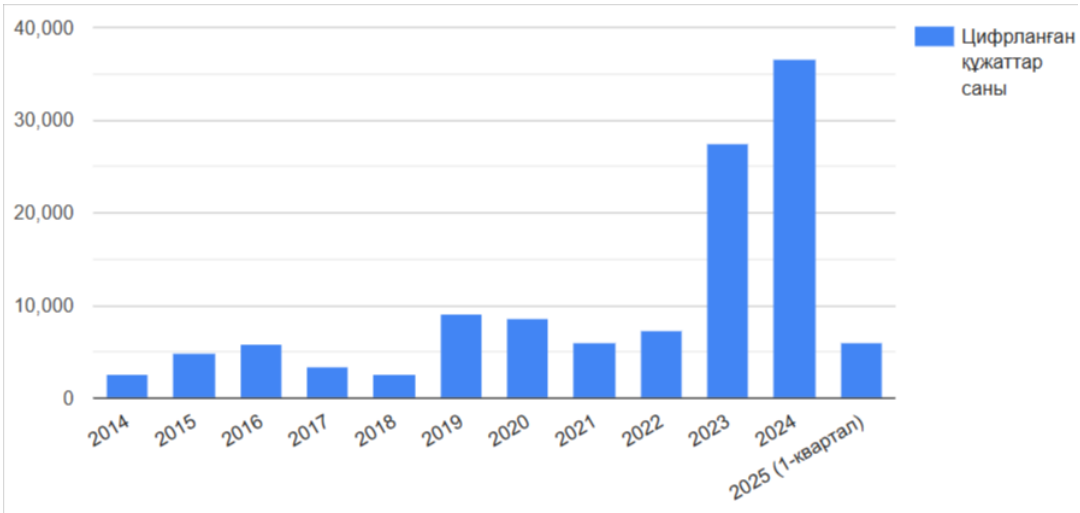
Мұрағаттық құжаттарды цифрландыру – бұл көптеген факторларға байланысты техникалық күрделі процесс: арнайы бағдарламалық-аппараттық жүйенің болуы, ұтымды жұмыс стратегиясын анықтау, сканерлеу операторларының біліктілігі және қажетті нормативтік-әдістемелік құжаттаманың болуы.

Цифрландыру қауіпсіз және үнемді сақтауға, қажетті ақпаратты ыңғайлы іздеуге, сондай-ақ мұрағаттық құжаттарға жедел қол жеткізуге мүмкіндік береді. Мұрағаттық құжаттарды цифрландыру бір уақытта бірнеше мәселені шешуге мүмкіндік береді:

- қор есебінде заттарды ретке келтіру;
- түпнұсқаларды айналымнан кезең-кезеңімен алып тастау арқылы олардың сақталуын қамтамасыз ету;
- мұрағат қорынан құжаттарға қол жеткізуді кеңейту және жеңілдету;
- электрондық қорға қолжетімділікті қамтамасыз ету;
- мемлекеттік қызмет көрсету процесін жеделдету;
- мұрағаттың ғылыми анықтамалық аппаратымен жұмыс жасау

Бұл мәселелерді шешу үшін жинақталған деректердің үлкен жиынтығын қажет ететін қолжазба мәтінді тану үшін соңғы технологиялық және зерттеу нәтижелерін жан-жақты зерттеу және енгізу қажет.

Мысалы, Алматы қаласындағы мемлекеттік архивтен алынған статистикаға қарасақ, онда 2014 жылдан бастап 2025 жылдың 1-тоқсанына дейінгі цифрланған құжаттар саны көрсетілген (1-сурет). Горизонталь осьте жылдар берілген, ал вертикаль осьте құжаттар саны (мыңдап) көрсетілген. Әр баған сол жыл ішінде цифрланған құжаттар санын білдіреді. 2023 және 2024 жылдары айтарлықтай өсу байқалады. 2025 жылдың алғашқы тоқсанында да елеулі көлемде құжат цифрланғаны көрінеді. Яғни бұл статистика құжаттарды цифрландырудың автоматтандырылған жүйенің қажеттілігін көрсетеді.




Сурет 1 – Алматы қаласындағы мемлекеттік архивтегі құжаттарды цифрландыру статистикасы

Дегенменде 01.04.2025 жыл жағдайы бойынша 120615 іс немесе 16,478,926 парақ цифрланған, бұл барлық құжаттардың тек 3% -ын ғана құрайды [4] (2-сурет).




Сурет 2 – Цифрландырылған құжаттардың үлесі

**Диссертациялық жұмыстың мақсаты** – қазақ тіліндегі архивтік қолжазбаларды тану үшін машиналық оқыту алгоритмдері мен нейрондық желі архитектурасын құру

**Зерттеудің міндеттері.** Қойылған мақсатқа жету үшін келесі міндеттерді орындау қарастырылады:

1) Қазақ тіліндегі жазылған қолжаза мәтінді тануға арналған машиналық оқыту әдістерін оқытуға және тестілеуге арналған деректер қорын жинау және өңдеу.
2) Қолжазба мәтінді тануға арналған жүйелерді, әдістер мен алгоритмдерді зерттеу және талдау;
3) Архивтік құжаттарды алдын ала өңдеу алгоритмдерін және сегменттеу моделін әзірлеу;
4) Қазақ тіліне арналған қолжазба мәтіндер деректерін жасанды көбейту үшін генеративті қарсылас нейрондық желілер (GAN) негізінде машиналық оқыту моделін әзірлеу;
5) Мәтінді тану үшін трансформер негізіндегі машиналық оқыту моделін әзірлеу;
6) Құжаттардан керек мәліметті іздеу жүйесін әзірлеу

**Зерттеу нысаны.** Қазақ тіліндегі қолжазба мәтіндердің кескіндері.

**Зерттеу пәні.** Қазақ тіліндегі қолжазба мәтіндерді тануға арналған машиналық оқыту алгоритмдері, нейрондық желілер және ақпараттық модельдер

**Зерттеу әдісі.** Компьютерлік көру алгоритмдері, терең нейрондық желілер, конволюциялық желілер, генеративті қарсылас желі (GAN), трансформер архитектурасы

**Жұмыстың ғылыми жаңалығы:**

1. Қазақ тілінде қолжазба мәтінін тану үшін бейімделген CNN+Transformer гибридті архитектурасы құрылды.
2. Қолжазбаның шынайы вариацияларын тудыратын қазақ тіліндегі қолжазба мәтін деректерін толықтыруға GAN архитектурасы негізіндегі әдіс ұсынылды. Бұл әдіс деректер саны аз кезде де тану дәлдігін жоғарылатуға болатынын көрсетті.

**Жұмыстың теориялық және практикалық маңызы.** Құжаттарды интеллектуалды өңдеудің автономды жүйесін дамыту Қазақстан Республикасында ақпараттық және телекоммуникациялық технологияларды дамытудың перспективалық бағыты болып табылады. Бұл жобаның нәтижелерін жүзеге асыру бірінші кезекте қолжазба мәтіндерін цифрландыру процесінің тиімділігін арттырады. Сәйкесінше, жұмыстың нәтижелерін іске асырудың негізгі әлеуметтік-экономикалық әсері әртүрлі түрдегі және көлемдегі құжаттарды өңдеуге уақыт пен қол күшін үнемдеумен байланысты.

**Қорғауға шығарылған негізгі тұжырым.**

1. Қолжазба мәтінді тану үшін конволюциялық нейрондық желі мен трансформер гибридті архитектурасы wer (сөздік қате көрсеткіші) және cer (әріптік қате көрсеткіші) метрикаларын сәйкесінше 10.44% және 2.85%

төмендетуге мүмкіндік береді.

2. Қолжазба мәтін деректерін жасанды көбейтудің GAN негізіндегі әдісі сөздерді тану дәлдігін 75.8%-дан 86.3%-ға арттыруға мүмкіндік береді.

**Диссертациялық жұмыс нәтижелерінің апробациясы.** Зерттеудің негізгі нәтижелері келесідей конференциялар мен ғылыми семинарларда ұсынылды:

1. **Yeleussinov A.**, Islamgozhayev T., Kozhirbayev Z. Creation and Annotation of a Handwritten Text Database for the Kazakh Language: Methodologies and Preliminary Results. In2024 International Conference on Electrical, Communication and Computer Engineering (ICECCE) 2024 Oct 30 (pp. 1-6). IEEE. (**Scopus**)
2. **Елеусинов А.И.**, Бурибаев Ж.А. Сегментация рукописных символов с исправлением наклона текста. V-международная конференция «Информатика и прикладная математика», Алматы, ИИВТ, 29.09.2020-01.10.2020.
3. **Елеусинов А.И.,** Амиргалиев Е. Н., Мажитов Ш. С. О разработке базы изображений рукописных символов казахского языка, Конференция"IV международная научно-практическая конференция "Информатика и прикладная математика", посвященной 70-летнему юбилею профессоров Биярова Т.Н., Вальдемара Вуйцика и 60-летию профессора Амиргалиева Е.Н. 25-29 сентябрь 2019, Алматы, Казахстан", уровень Международный, КАЗАХСТАН, Институт информационных и вычислительных технологий МОН РК, 25.09.2019-29.09.2019.

Диссертациялық жұмыс сонымен қатар кафедраның ғылыми семинарларында талқыланды.

**Ғылыми жұмыстың жарияланымдары.** Диссертация тақырыбы бойынша алынған нәтижелер келесідей баспаларда жарияланды.

Scopus:

1. **Yeleussinov, A**., Amirgaliyev, Y., Cherikbayeva, L. Improving OCR Accuracy for Kazakh Handwriting Recognition Using GAN Models. *Appl. Sci.* **2023**, *13*, 5677 (SJR 0.492, процентиль 75 ) https://doi.org/10.3390/app13095677
2. Amirgaliyev B., **Yeleussinov A.**, Taizo M. Kazakh handwritten recognition. Journal of Theoretical and Applied Information Technologies, 2020, 98(14), pp. 2744–2754.

КОКСНВО:

3. Амиргалиев Е. Н., Миачи Т., **Елеусинов А.И.** Методы и алгоритмы формирования базы и обработки рукописных текстов на казахском языке, Вестник КазНИТУ, 2020г., №138, сс. 653-660.

**Диссертация құрылымы және көлемі.** Диссертациялық жұмыс кіріспеден, 3 бөлімнен, қорытынды, әдебиеттер тізімі және қосымшалардан тұрады. Диссертацияның толық көлемі: 84 бет машиналық мәтіні, оның ішіне 54 сурет, 6 кесте, 83 пайдаланылған әдебиеттер тізімі және 2 қосымша кіреді.

# 1 ҚОЛЖАЗБА МӘТІНДІ ТАНУДЫҢ ЕСЕБІ МЕН АЛГОРИТМДЕРІ

Бұл бөлімде қазақ қолжазба әріптерінің дерекқорын жасау үшін қолжазба кескіндерінің деректерін жинау процесі сипатталады. Қолжазба мәтінді тану есебі тривиалды емес болуының себептерін келесідей тұжырымдауға болады:

- жазу стилінің әр түрлілігі;
- курсивті мәтін;
- әріптер арасындағы ұқсастық.

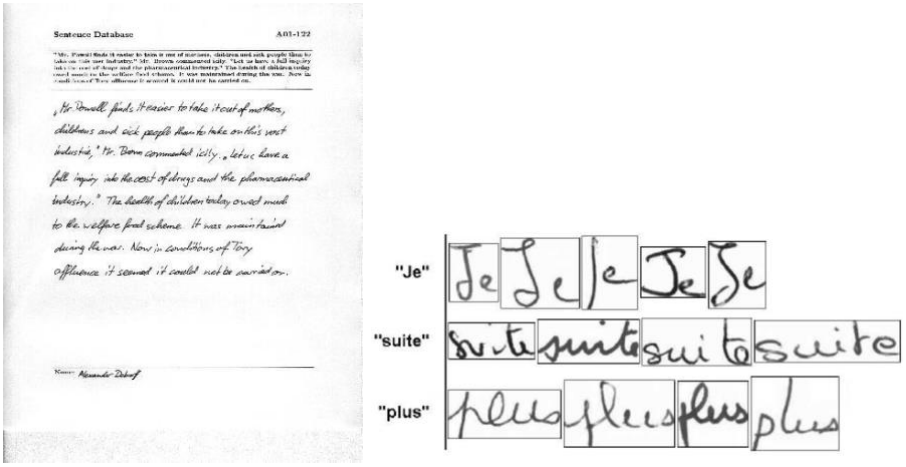
## 1.1 Қазақ тілінде қолжазба деректер жинағын құру және өңдеу әдістері

1.1.1 Қолжазба мәтіндер деректер жинағына шолу

Қолжазба мәтінді тану саласында айналысқан ғалымдар OCR жүйені бағалау үшін көптеген деректер базасын құрған. Офлайн қолжазбаны тануды бағалауға арналған мәліметтер базасы көптеген тілдер мен жазу жүйелері үшін қол жетімді. Мысалы, ағылшын тіліндегі қолжазба мәтіндер IAM және CEDAR деректер базасында көрсетілген [5]. IAM дерекқорында 657 автор жасаған қолмен жазылған мәтін жолдарының 13353 суреті бар. Бұл жазушылар транскрипциялаған мәтіндер британдық ағылшын тілінің Ланкастер-Осло/Берген корпусынан алынған. Ол 115320 сөзден тұратын және қолмен жазылған 1539 бетті құрайтын 657 автор үлесін қосады және қазіргі жинақтың бөлігі болып табылады. Деректер базасы сөйлем, жол және сөз деңгейлерінде белгіленеді. Сөздер автоматты түрде сегменттеу схемасының көмегімен сканерленген мәтін беттерінен алынған және қолмен тексерілген.

Барлық форма, сызық және сөздік кескіндер PNG файлдары ретінде беріледі және сәйкес пішін белгісінің файлдары, оның ішінде сегменттеу ақпараты мен бағаланатын параметрлердің әртүрлілігі мета ақпарат ретінде кескін файлдарына енгізілген.

Француз тіліндегі мәтіндер RIMES базасында көрсетілген [6]. Дерекқорды жинауға 1300-ден астам адам қатысқан және әрбір адамға бес мәтіннен жазу тапсырылған. Осылайша алынған RIMES дерекқорында екі-үш беттен тұратын 5605 хатқа сәйкес келетін 12 723 бет бар.

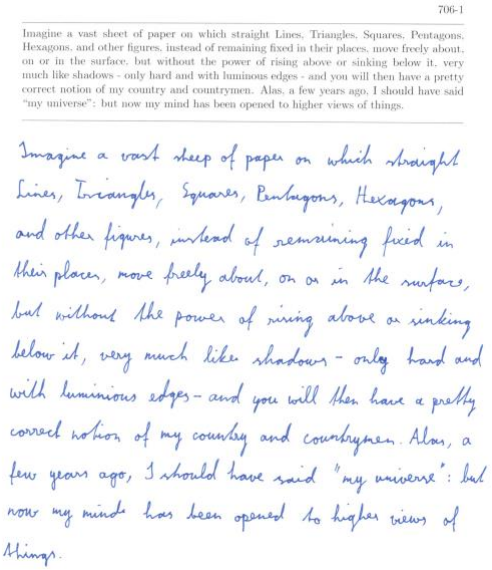


(a) (b)

Сурет 1.1 – (a) кескіндер, (b) RIMES базасынан мысал [6]

CVL (Computer Vision Lab) дерекқоры – бұл авторды іздеу, анықтау және сөз іздеу үшін жалпыға қол жетімді мәліметтер базасы [7]. Деректер базасы 7 түрлі қолжазба мәтіндерінен тұрады (1 неміс және 6 ағылшын мәтіндері). Деректер жиынтығына барлығы 310 автор қатысты. Оның 27-сі 7 түрлі мәтін жазды, ал 283 автор 5 түрлі мәтін жазды. Әр мәтін үшін қолмен жазылған мәтіннен және басылған мәтіннің үлгісінен тұратын түрлі-түсті RGB (300 dpi) кескіні, сондай-ақ кесілген нұсқасы (тек қолмен жазылған мәтін) бар. Бірегей идентификатор авторды анықтайды, ал әрбір жеке сөзге арналған шектеулер XML файлында сақталады.

CVL дерекқоры әдеби шығармалардан таңдалған қолмен жазылған неміс және ағылшын мәтіндері бар суреттерден тұрады. Барлық беттерде автордың бірегей идентификаторы және жоғарғы оң жақ бұрыштағы мәтін нөмірі (сызықшамен бөлінген), содан кейін басылған мәтін үлгісі болады. Мәтін екі көлденең бөлгіштің арасына орналастырылады. Басылған мәтіннің астында адамдардан мәтін жолдарының бұралуын болдырмау үшін сызғышқа төменгі парақты пайдаланып мәтін жазуды сұрады. Орналасу IAM мәліметтер базасының стиліне сәйкес келеді. Деректер базасы 09.12.2013 ж. жаңартылды, өйткені бір жазба идентификаторы (265/266) қате болды. Нұсқа нөмірі 1.1-ге өзгерді.


706-1

Imagine a vast sheet of paper on which straight Lines, Triangles, Squares, Pentagons, Hexagons, and other figures, instead of remaining fixed in their places, move freely about, on or in the surface, but without the power of rising above or sinking below it, very much like shadows - only hard and with luminous edges - and you will then have a pretty correct notion of my country and countrymen. Alas, a few years ago, I should have said "my universe": but now my mind has been opened to higher views of things.

Imagine a vast sheep of paper on which straight Liners, Triangles, Squares, Pentagons, Hexagons, and other figures, instead of remaining fixed in their places, move freely about, on or in the surface, but without the power of rising above or sinking below it, very much like shadows - only hard and with luminious edges - and you will then have a pretty correct notion of my country and countrymen. Alas, a few years ago, I should have said "my universe": but now my mind has been opened to higher views of things.

Сурет 1.2 – CVL дерекқорынан мысал [7]

Сонымен қатар осындай дерекқорлар орыс, ұйғыр, парсы, араб, испан, тамиль, корей, қытай, жапон, грек, индонезия тілдерінде бар [8, 9]. Аталған дерекқорлар құрылымдарымен де, мазмұндарымен де ерекшеленеді. Кейбірінде тек әріптердің бейнелері болса, басқаларында сөздердің немесе тұтас сөйлемдердің бейнелері бар (кесте-1.1)

Кесте 1.1 – Қолжазба деректер жинағының салыстырмалы кестесі

| **Деректер жинағы** | **Мақсаты** | **Құжат түрі** | **Аннотация түрі** | **Бет сандары** |
|---|---|---|---|---|
| IAM | сөзді тану | ағылшын қолжазбалар | сөздер, сөйлемдер | 13353 мәтін жолдары |
| Esposalles | сөзді тану | архив құжаттары | сөздер | 173 бет, 5447 мәтін жолдары |
| READ-BAD | жолды анықтау | архив құжаттары | беттер | 2036 бет |
| DIVA-HisDB | бетті тану | көне тарихи құжат | бетте | 150 бет, 3 қолжазба |
| St.Gall | сөзді тану | латын қолжазбасы | сөз және жолдар | 60 бет, 1410 жол, 11597 сөз |
| Parzival | сөзді тану | неміс қолжазбалары | сөз және жолдар | 47 бет, 4477 жол, 23478 сөз |
| Pinkas | бетті сегменттеу | еврей қолжазбалары | жолдар мен беттер | 30 бет, 13744 сөздер |
| Cyrillic Handwriting Dataset | сөзді тану | орыс қолжазбалар | сөз | 73830 сөздер |
| Digital Paper | сөзді тану | тарихи орыс қолжазбасы | сөз және жолдар | 9694 бет, 50998 сөз |
| HKR, KOHTD | сөзді тану | қазақ, орыс қолжазбалар | сөздер | 1500 бет, 15245 жол, 106718 сөз<br>3000 бет, 140335 сөз |

Деректер жинақтарына талдау жасай келе олардың келесідей кемшіл тұсын байқауға болады:

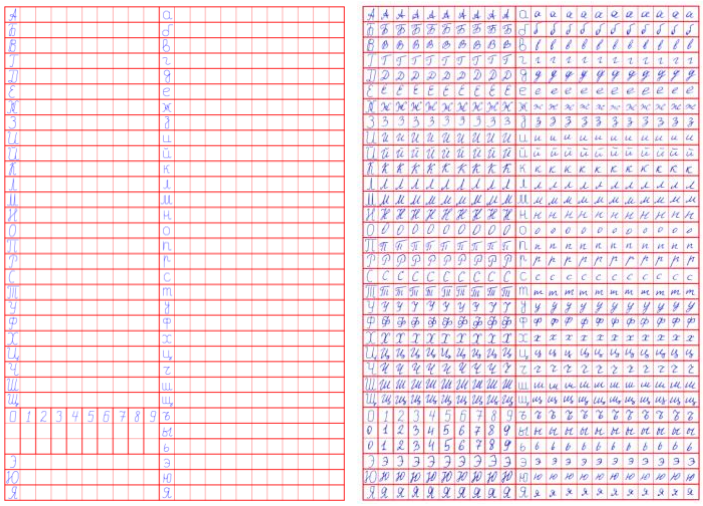
- жолдар мен сөздердің, әріптерді сегменттеу үшін аннотациялардың жоқтығы

Осындай кемшіліктерді толықтыру мақсатында біз өзіміздің деректер қорын құруға шештік.

### 1.1.2 Қазақ қолжазба мәтіндердің жинағын құру кезеңдері

Әріптер кластары қазақ тілінің орфографиялық құрылымы негізінде анықталады. Кириллица негізіндегі қазақ әліпбиі 42 кіші және үлкен әріптерден тұрады, оның ішінде орыс әліпбиінің 33 әрпі және қазақ тілінің 9 ерекше әрпі бар. Осы жұмыста қазақ алфавитінің барлық таңбаларын ("ё, ъ, ь, h" - әріптерінен басқа, бас және кіші) қамтиды.

Жоғарыда жасалынған шолулардан байқағанымыздай, көптеген деректер базасын құру алдымен авторларға арналған жазу сауалнамасын құрудан басталады. Мысалы, төменде көрсетілген орыс тіліне арналған сауалнамадан байқауға болады [10]. Бұл сауалнамаға қатысушы автор көрсетілген үлгі бойынша барлық ұяшықтарды қолмен толтырып шығады. Содан кейін қалған өңдеу амалдары жүргізіледі.



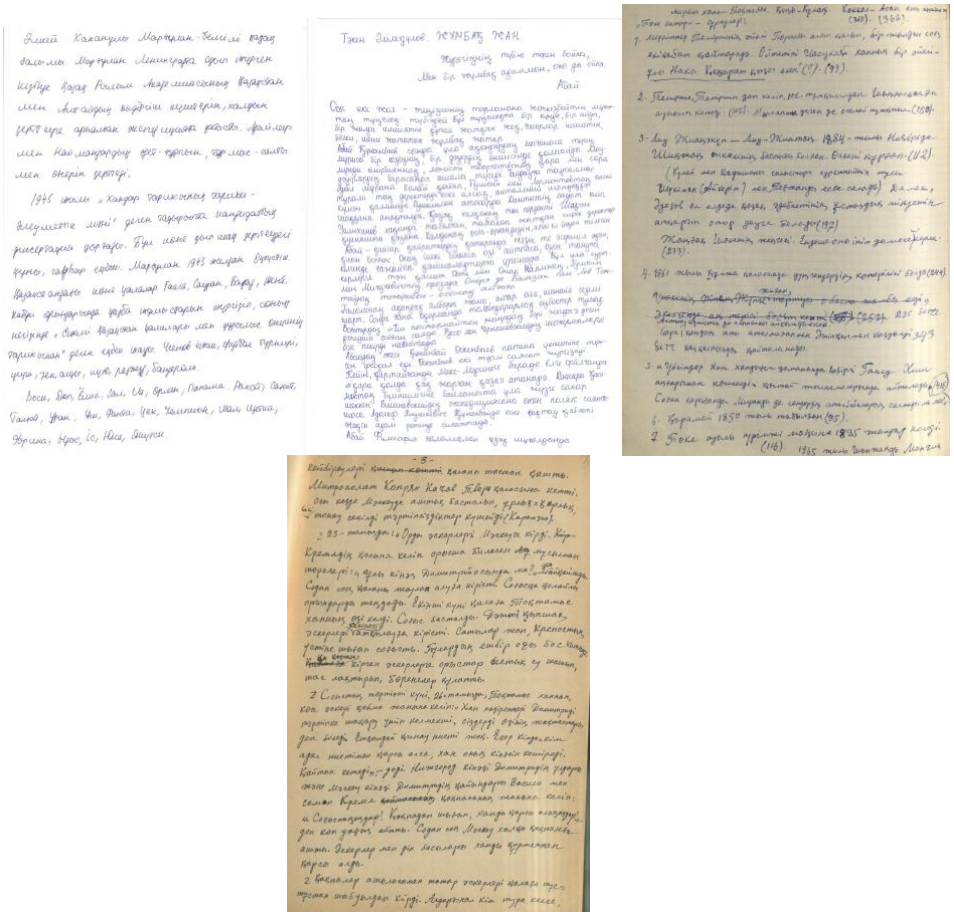
Сурет 1.3 – Бос (А) және толтырылған (Б) бланкілердің мысалдары [10]

Біз бұл жұмыста мәтіндер шынайылыққа жақын болу үшін арнайы сауалнама формасын қолданудан бас тарттық. Себебі қазақ тілінде жазба мәтіндерде әріптер көбіне бір біріне жалғанып жазылады. Ал арнайы сауалнама кезінде әріптер жеке жазылатындықтан ол жалғаулар жазылмайды. Бұл өз кезегінде келесі қадам сегментация үшін шынайы жазба болып есептеледі. Сонымен қатар арнайы ұяшықтың ішінде ғана жазу кейбір қазақша әріптердің формасын өзгертіп жібереді. Нәтижесінде автордың жазу стилі арнайы бланкіде басқаша, шынайы мәтінде басқаша болуы мүмкін.

Біздің деректер қорымызда А4 форматындағы қағазға жас және жыныс ерекшеліктері әр түрлі 120 дан аса авторлардан жиналған жазба мәтіндерден құралды. Мәтіндер барынша қазақ алфавитіндегі барлық 42 әріпті қамтитындай етіліп, әр түрлі әдебиет көздерінен алынды. Соңында жазу стильдері әр түрлі жазбалар жинақталды. Жазылған мәтіндер сканерден өткізіліп, *jpeg* форматында кескін файл ретінде сақталды. Файлдар аттары *1.jpeg* ден бастап нөмірленеді. Төменді суретте датасеттен алынған бейнелердің мысалдары көрсетілген.

Сонымен қатар Алматы қаласындағы мемлекеттік архивтен алынған қолжазбалардың кескіндері бар. Ол 56 автордың 105 архивтік жазбасынан тұрады.

Нәтижесінде барлық авторлар саны 176 және 745 беттен құрылған қазақ тіліндегі қолжазба мәтін кескіндерінен тұратын деректер жинағы құрылды (сурет -1.4)



Сурет 1.4 – Қолжазба кескін мысалдары

1.1.3 Деректер жинағын аннотациялау және өңдеу

Келесі қадам қолжазба мәтіндердің бейнесін алдын ала өңдеу амалдарын жүргізу. Ол үшін сканерленген кескіндерді қолмен аннотациялау (labeling) жұмыстарын жүргізу қажет.

Кескінді аннотациялау жасанды интеллектті дамытудың ажырамас бөлігі болып табылады және бұл компьютерлік көру технологиясының негізгі міндеттерінің бірі. Аннотацияланған суреттер нейрондық желілерді оқыту үшін кіріспе деректер ретінде қажет: кескіндегі объектілерді тану компьютерлерге бейнелерді адам сияқты "көруге" мүмкіндік береді.

Көптеген машиналық оқыту модельдері "мұғаліммен оқыту" (supervised learning) әдісі негізінде жұмыс істейді. Жоғарыда аталған процестерді қолдау үшін алдымен барлық деректерді аннотациялау қажет. Мұнда адам іске араласады және бағдарламалық жасақтаманың көмегімен барлық тиісті шикі

деректерді белгілейді. Мысалы, көліктері бар суреттерді тани алатын машиналық оқыту моделін құру үшін алдымен көліктерді белгілеу керек. Машиналық оқыту процесін жақсарту үшін пиксельді белгілеуді жүзеге асыруға болады. Суреттердегі нысандарды қолмен белгілеу көп уақытты қажет етеді және өте қымбат жұмыстардың бірі, әсіресе үлкен деректер жиынтығын белгілеу қажет болған кезде.

Аннотациялау құралдары мақсаттары мен қолайлығына байланысты әр түрлі болып келеді. Сонымен қатар аннотацияланатын деректердің типіне байланысты бағдарламалық жасақтаманы келесідей жіктеуге болады:

- Табиғи тілдерді өңдеу (NLP)
- Аудио деректер
- Компьютерлік көру деректері

Деректер осы түрлеріне қарай әрқайсысы өзінше аннотацияланады. Процесс тек деректер түріне ғана емес, сонымен қатар белгілі бір бағдарламалық өнімнің мүмкіндіктеріне де байланысты. Мысалы, суреттерді белгілеу кезінде негізгі нүктелер мен пикселдерді орналастыру қажет, ал мәтін болған жағдайда оқу мәліметтер жиынтығын құру үшін мәтін үзінділерін таңдау керек. Егер аудио туралы айтатын болсақ, онда аудиоматериалды санаттау үшін оны декодтау қажет болады. Дәл осы айырмашылықтарға байланысты барлық бағдарламалық құралдар әртүрлі өлшемдерге сәйкес тексеріледі. Деректерді белгілеу құралдарының маңыздылығын ескере отырып, бұл құралдар егжей-тегжейлі зерттелген көптеген мақалалар бар екендігі таңқаларлық емес. Көптеген компаниялар бағдарламалық жасақтаманы бірнеше критерийлер бойынша таңдайды:

- Мүмкіндіктер - бұл құрал қандай мүмкіндіктер береді;
- Қиындық - бұл бағдарламалық жасақтаманы пайдалану қаншалықты қиын;
- Өнімділік - белгілеу бағдарламасы қаншалықты тез жұмыс істейді;
- Ынтымақтастық - бірнеше адамның бір уақытта аннотациялаумен айналысуын қамтамасыз етуі.

Деректерді аннотациялау жұмысын жеңілдететін бірнеше ашық бастапқы арнайы құралдар бар. Солардың негізгілеріне тоқталсақ.

CVAT - бұл сандық кескіндер мен бейнелерді белгілеуге, сонымен қатар компьютерлік көру алгоритмдеріне арналған мәліметтер жиынтығын дайындауға арналған ақысыз және ашық құрал. Бұл бірнеше машиналық тапсырмаларды орындау үшін деректерді орналастыруға мүмкіндік береді: нысандарды тану, суреттерді жіктеу және сегментациялау. CVAT бірқатар қосымша компоненттерді қолдайды: Deep Learning Deployment Toolkit (Open VINO құрамындағы компонент), NVIDIA CUDA Toolkit, TensorFlow Object Detection API және басқалары [11].

LabelImg құралы суреттегі нысандарды бөлектеу үшін пайдаланылатын python-да жазылған суреттерді аннотациялауға арналған тегін ашық кодты графикалық құрал [12]. Аннотацияларды ImageNet пайдаланатын pascal, voc

форматтарында xml файл ретінде сақтауға болады. Аннотациялау qt графикалық интерфейсіндегі нысандарды белгілеу үшін тіктөртбұрыштар (bounding boxes) жасауға мүмкіндік береді. Сонымен қатар, ол yolo және CreateXml форматтарын қолдайды.

Біз бұл жұмыста қазақ тілінің жазба мәтіндерінің базасын құру үшін LabelMe аннотациялау құралын пайдаландық. Себебі бұл құрал керекті объектіні полигонмен белгілеуге мүмкіндік береді, яғни ол әр түрлі стильдегі жазуларды еркін әрі дұрыс аннотациялауға мүмкіндік береді.

LabelMe - Массачусетс технологиялық институтының (CSAIL) информатика және жасанды интеллект зертханасы жасаған жоба [13]. Көбінесе ол компьютерлік көру зерттеулерінде қолданылады. 2010 жылғы 31 қазандағы жағдай бойынша LabelMe-де 187 240 сурет, 62 197 аннотацияланған сурет және 658 992 таңбаланған объект бар. LabelMe құралы пайдаланушыларға жобаға өз үлестерін қосуға мүмкіндік береді. Құралға жасырын түрде немесе тегін есептік жазбаға кіру арқылы кіруге болады. Құралға қол жеткізу үшін пайдаланушыларда JavaScript қолдайтын үйлесімді веб-шолғыш болуы керек. Құрал жүктелген кезде, ол LabelMe деректер жиынтығынан кездейсоқ суретті таңдайды және оны экранда көрсетеді. Егер объектілердің белгілері кескінге қосылған болса, олар кескіннің үстіне көпбұрышты форматта қойылады. Нысанның әр жеке белгісі басқа түспен көрсетіледі. Егер кескін толығымен белгіленбеген болса, пайдаланушы тінтуірді суреттегі объект бар көпбұрышты салу үшін қолдана алады. Мысалы, көрші суретте, егер адам ғимараттың алдында тұрса, пайдаланушы адамның шекарасындағы нүктені нұқып, бастапқы нүктеге оралғанша сыртқы жиекті нұқуды жалғастыра алады. Көпбұрыш жабылғаннан кейін экранда қалқымалы терезе пайда болады, ол пайдаланушыға нысанға тег енгізуге мүмкіндік береді. Пайдаланушы объектіні жақсы сипаттайтын кез-келген белгіні таңдай алады. Егер пайдаланушы суреттің алдыңғы таңбалануымен келіспесе, ол объект контурының көпбұрышын нұқып, көпбұрышты толығымен алып тастай алады немесе жаңа атау беру үшін мәтіндік белгіні өңдей алады. Пайдаланушы кескінге өзгертулер енгізгеннен кейін олар сақталады және барлығына LabelMe деректер жиынтығынан жүктеуге болады. Осылайша, осы құралды пайдаланатын пайдаланушылар қауымдастығының қосқан үлесі арқасында деректер үнемі өзгеріп отырады. Пайдаланушы суретті аяқтағаннан кейін "басқа суретті көрсету" сілтемесін нұқып, пайдаланушыға көрсету үшін басқа кездейсоқ кескін таңдалады.

LabelMe құралдарының функциялары:

- Көпбұрыш, тіктөртбұрыш, шеңбер, сызық пен нүкте пішіндері арқылы аннотациялау мүмкіндігі ;
- Классификациялау және тазарту үшін жалауша кескінінің аннотациясы.
- Видео аннотация;
- Графикалық интерфейсті баптау (алдын-ала анықталған белгілер, жалаулар, авто сақтау, белгілерді тексеру және т.б.);

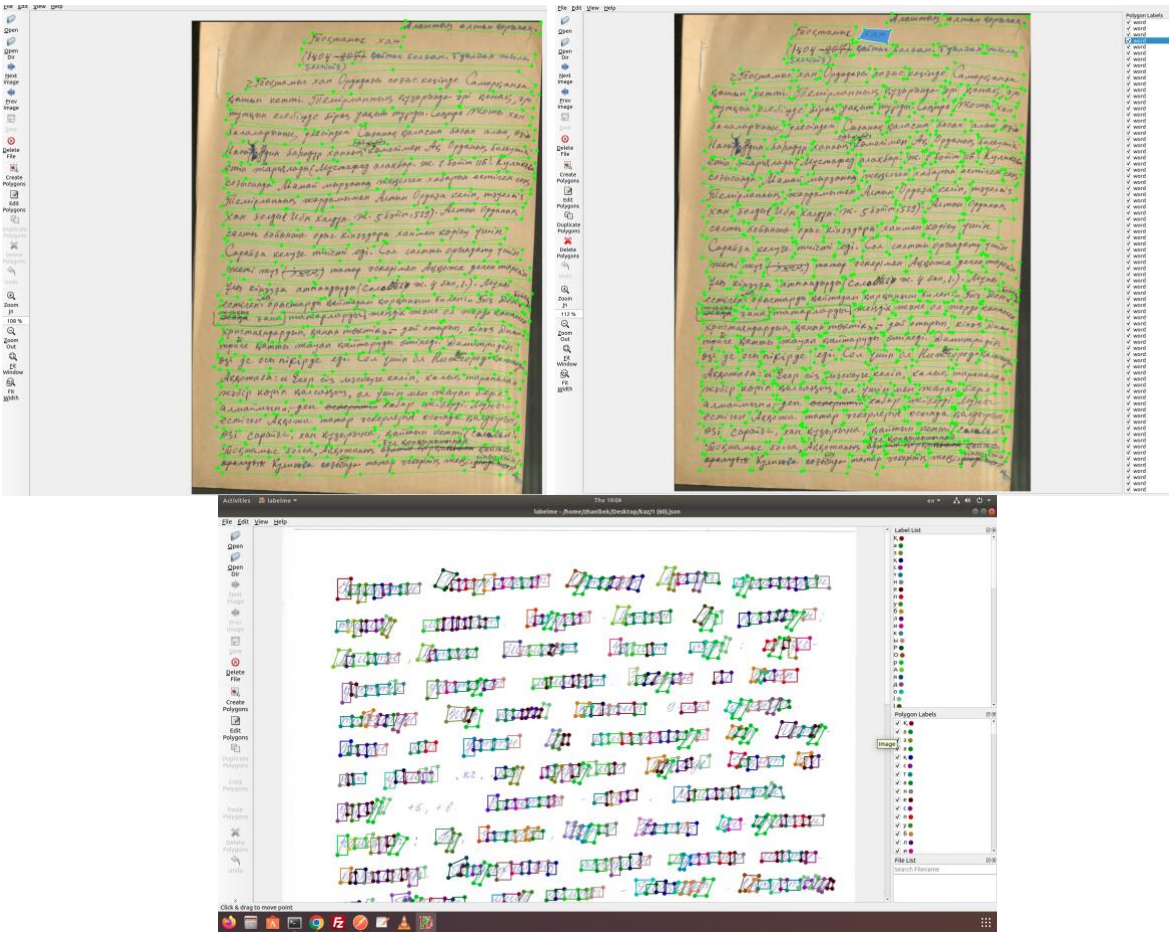
- Семантикалық және дана (экземпляр) сегментация үшін voc форматындағы мәліметтер жиынтығын экспорттау;
- Экземплярларды сегментациялау үшін COCO форматындағы мәліметтер жиынтығын экспорттау.

Қазақ қолжазба мәтінін аннотациялау үшін Labelme бағдарламасын ашу және қажетті белгіленетін суретті таңдау қажет. Аннотациялаудың бірінші кезеңі - мәтін жолдарын таңдау, олардың әрқайсысы полигон құралының көмегімен қолмен белгіленеді:

- аннотатор мәтіннің әрбір жолының айналасына тұйық көпбұрыш салады;
- әрбір көпбұрыш line_# белгісін алады (мысалы: жол_1, жол_2, …);
- барлық керекті объектілерді қалдырмай белгілеп болғасын файл сақталады.

Аннотациялау кезінде сызықтық көпбұрыштар арасындағы қабаттасуды болдыртпау және сызық шекарасынан шықпай бүкіл мәтінді жабу үшін көпбұрышты салу кезінде мәтіннің еңісін ескеру маңызды.

Осылайша, LabelMe заманауи OCR жүйелерін оқыту және бағалау үшін маңызды болып табылатын сызықтан таңбаға дейін бірнеше деңгейде қолжазба мәтіндердің дәл және икемді аннотациясына мүмкіндік береді. Аннотациялау сөздер мен әріптер үшін де қайталанады (1.5-сурет).



Сурет 1.5 – Labelme бағдарламасымен кескіндерді аннотациялау

Файл үнсіз келісім бойынша JSON форматынды сақталады. JSON (ағылш. JavaScript Object Notation) - JavaScript-ке негізделген деректермен алмасудың мәтіндік форматы. Көптеген басқа мәтіндік форматтар сияқты, JSON-ны адамдар оңай оқи алады. JavaScript-тен шыққанына қарамастан (дәлірек айтқанда, 1999 жылғы ECMA-262 стандартты тіл жиынтығынан), формат тілге тәуелсіз болып саналады және оны кез-келген бағдарламалау тілінде қолдануға болады. Көптеген тілдер үшін JSON форматында деректерді құруға және өңдеуге дайын код бар. Төменде көрсетілген суретте (сурет 1.6) сақталған JSON файлдың құрылымын көруге болады.


```
"shapes": [
  {
    "line_color": null,
    "shape_type": "rectangle",
    "points": [
      [
        304.4117647058823,
        229.41176470588235
      ],
      [
        245.58823529411762,
        319.1176470588235
      ]
    ],
    "flags": {},
    "fill_color": null,
    "label": "К"
  },
```


Сурет 1.6 – Аннотациялаған мәтін үлгісі

Әрбір суретті аннотациялап болғаннан кейін сақталған файлдар тізімі келесідей нәтижемен көрінеді (сурет 1.7).

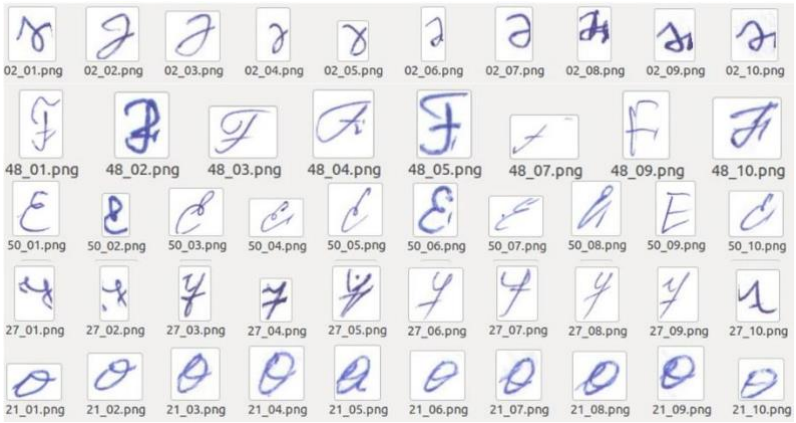



Сурет 1.7 – Файлдар тізімі

Мұндағы *img.png* - аннотацияланған суреттің түпнұсқасы, *info.yaml* - класстардың аттарының кодтары, *label.png* - барлық сызылған полигондар, *label_names.txt* -

класстардың аттары, *label_viz.png* - полигондар мен класстардың бірге көрсетілген визуализация файлы.

Келесі қадам - қолмен жазылған мәтін бейнелерінен белгіленулері (кластары) бойынша әріптерді “қырқып алу”. Ол үшін *json* файлдардан әріптердің белігілеудегі координаталары қолданылады. Мысалы жоғарыдағы суретте json файлдың points деп аталатын өрісінде полигонның нүктелерінің координаталары жазылады. Сол координаталы бойынша қырқып алынған әріптердің бөліктері сәйкесінше жеке қалталарға, “01_01.png” үлгі бойынша нөмірленіп салынады. Мұндағы бірінші нөмір әріптің класын білдірсе, екінші нөмер ретін білдіреді. Төменде осылайша жиналған әртүрлі әріптердің әртүрлі жазылу мысалы көрсетілген (сурет 1.8).




Сурет 1.8 – Әріптердің әртүрлі жазылу түрлері

Осылайша сөздер де аннотацияланған координаталары бойынша “қиылып” алынып жеке сурет ретінде сақталады. Файл аты бастапқы сурет атынан және сөздің реттік нөмерінен құралады. Осылай барлық суреттер бір csv файлға жазылады (сурет 1.9).

```
60_7.png    және
60_8.png    өзіндік
60_9.png    заңды
60_10.png   құқығы
60_11.png   құрылатын
60_12.png   белгілейтін
60_13.png   бірлестік
60_14.png   әріп жазуда
60_15.png   дыбысты
60_16.png   бар
60_17.png   қабілеттіліктің
60_18.png   ауқымды
60_19.png   шартты
60_20.png   таңба
```



Сурет 1.9 - Датасеттегі сөздердің аннотациясы және мысалы

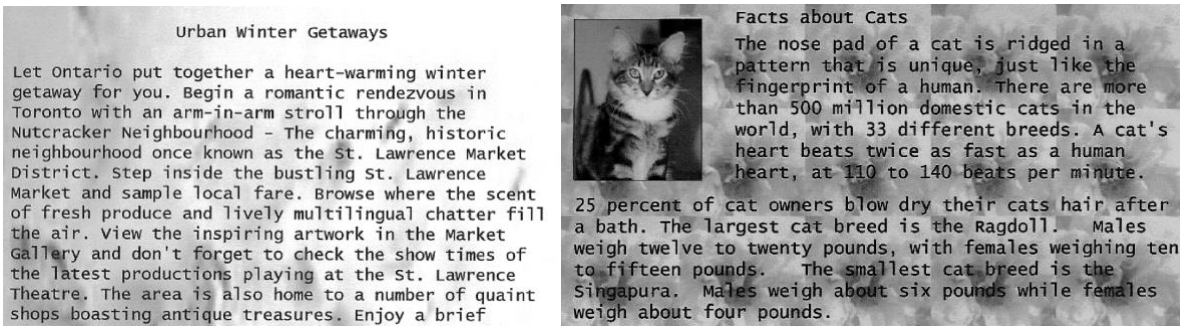
Нәтижесінде біздің дерекер жинағы 745 беттен, 8345 жолдан, 45211 сөзден және 245654 символдан тұрады [81, 82, 83].

### 1.2 Алдын-ала өңдеу әдістерінің мәтінді танудағы маңыздылығы

Бейне тану аймағы – бұл суретті өңдеу, машиналық көру және жасанды интеллект сияқты басқа салалардың негізін құрайтын көп салалы аймақ. Демек, OCR суретті өңдеусіз және/немесе жасанды интеллектсіз қолданыла алмайды. Кез-келген OCR жүйесі көптеген кезеңдерден өтеді, соның ішінде деректерді жинау, алдын-ала өңдеу, белгілерді алу, жіктеу және кейінгі өңдеу, мұнда ең маңызды аспект деректерді өзгерту немесе шектеулерге байланысты деректерді жинау процесінде кемшіліктерді түзету үшін қажет алдын-ала өңдеу болып табылады. Деректерді алдын-ала өңдеу оларды басқа өңдеу процедурасына дайындау үшін шикі деректермен жасалатын кез-келген өңдеу түрін сипаттайды. Сондықтан, алдын - ала өңдеу-бұл деректерді өңдеуге оңай және тиімді болатын форматқа түрлендіретін алдын-ала кезең. Сондықтан, түсірілген деректерді алдын-ала өңдеудің негізгі міндеті - тану жылдамдығының төмендеуіне әкелетін және күрделілікті арттыратын вариацияларды азайту, өйткені, мысалы, таңбалардың штрихтарын алдын-ала өңдеу таңбаларды танудың тиімді жүйелерінің сәттілігі үшін өте маңызды. . Осылайша, алдын-ала өңдеу белгілерді алудың алдында маңызды қадам болып табылады, өйткені ол нәтижелердің кейінгі кезеңдерге жарамдылығын анықтайды. Бейнені тану жүйесіндегі кезеңдер конвейер сияқты, яғни әр кезең оңтайлы, сенімді нәтиже алу үшін

алдыңғы кезеңнің сәттілігіне байланысты болады.

OCR көмегімен мәтінді тану дәлдігіне әсер ететін бірқатар факторлар бар. Бұл факторларға мыналар жатады: сканердің сапасы, сканерлеудің ажыратымдылығы, басып шығарылған құжаттардың түрі (лазерлік принтер немесе фотокөшірме), қағаз сапасы, мәтінде қолданылатын қаріптер, тілдік қиындықтар және қолданылатын сөздік. Ескі қағаз құжаттардағы белгілері және біркелкі емес жарықтандыру ақ фонда таза мәтінмен салыстырғанда мәтінді тану дәлдігіне әсер ететін мәселелердің мысалы болып табылады. Мысалы, 1.10 (a) суретте жарық аз болған кезде құжаттың сұр деңгейдегі бейнесі, ал 1 (b) суретте аралас мазмұны мен күрделі фоны бар құжаттың суреті көрсетілген. Басқа факторларға мәтіннің біркелкілігі, туралануы және бетте орналасуы, графика және графикалық мазмұн сияқты басып шығару ерекшеліктері кіреді [14].




Сурет 1.10 (a, b) – Біртекті емес және күрделі фоны бар құжаттар суреттерінің мысалдары [14]

Қолжазбаны оффлайн танудың көптеген заманауи тәсілдері үш негізгі кезеңнен тұрады, атап айтқанда алдын-ала өңдеу, сегментациялау және тану.

Алдын ала өңдеу кезеңі тиісті мәтін бөліктерін алуға және оларды сегментация мен тануға дайындауға бағытталған. Бұл әдістерге әдетте шуды жою, кескінді сегментациялау, кесу, масштабтау және басқалар жатады. Біздің жобада бұл әдістер негізінен кескінді тануда қолданылды, бірақ олардың кейбіреулері, мысалы, жазылған таңбаны кесу және оны енгізу көлеміне дейін масштабтау сенсорлық режимде де орындалады. Ең алдымен, көлбеу мен қисаюды түзету үшін кескінді физикалық түрде бұруға және / немесе жылжытуға болады. Осы мақсатта белгілі бір мәтіндегі көлденең / тік соққылардың орташа көлбеуі Хаф контурын түрлендіру арқылы немесе пиксель тығыздығының бағытталған гистограммаларын есептеу арқылы бағаланады [15]. Содан кейін әр мәтін жолының негізгі сызығы анықталады, ол қолжазба мәтін орналасқан аймақты анықтайды.

Осылайша, алдын-ала өңдеу әдістерін қолдану құжатты таңбаларды тану жүйесіндегі келесі кезеңге дайындау арқылы оның кескінін жақсарта алады. Сондықтан жоғары тану көрсеткіштеріне қол жеткізу үшін алдын-ала өңдеудің тиімді кезеңі болуы керек. Тиімді алдын-ала өңдеу алгоритмдерін қолдану мәтінді оптикалық тану жүйесін сенімді етеді, негізінен кескінді дәл жақсарту, шуды жою, кескіннің шекті мәндерін белгілеу, бұрмалануды анықтау / түзету,

бетті сегментациялау, таңбаларды сегментациялау, таңбаларды қалыпқа келтіру және морфологиялық әдістер.

Суреттен шуды жою үшін құжат кескіні сүзіледі. Содан кейін әр пиксель қара немесе ақ болып қалуы үшін шекті мән орнатылады. Ақыр соңында, сызықтардың енін 1 пиксельге дейін азайту үшін жұқарту алгоритмі қолданылады. Жұқарту бейнелердің сипаттамаларын анықтауды жеңілдету үшін әріптердің бейнесінің қаңқасын табады. Содан кейін шикі қаңқалар шеткі нүктелерді алып тастау арқылы тегістеледі. Келесі маңызды мәселе автор қолмен толтыруы керек ақпараттық өрістердің орнын көрсету үшін пайдаланылатын баспа жақтаулары мен көлденең көрсеткіштерді жою. Мұндай өрістердің мысалдары сандық өрістер, заңды сомалар және банк чектеріндегі клиенттің қол қою өрістері болып табылады. Нұсқаулар қолмен жазылған мәтінмен қабаттасуы мүмкін болғандықтан, кейбір мәтіндік пикселдер де алынып кетеді. Байланыстырылған штрих немесе сызықтық сегментті екі немесе бірнеше бөлікке бөлуге болады, мысалы, нәтижесінде "ш" әрпі "м" - ге айналады. Мұндай түсініксіздік сөздіктегі сөздерді салыстыру арқылы шешіледі.

Қолжазба мәтінінің эрозиясын жоюдың бірнеше түрлі тәсілдері ұсынылды [16], [17]. Типтік мысалдары морфологиялық эрозия және дилатация операторлары [18]. Жалпы алғанда, қарапайым эвристикалық критерийлер ұсыныстарды мәтіннен ажыратуға көмектеседі. Мысалы, бағыттағыштың орташа бағытын өзгерту мәтінге жататын пикселдердің болуын көрсетеді.

Көлденең тығыздықтың гистограммаларын сөздің жоғарғы, ортаңғы және төменгі аймақтарын анықтау үшін де қолдануға болады. Сонымен қатар, қолжазбаның ерекшеліктерін (негізгі әріптерге сәйкес келетін) жоғарғы немесе төменгі аймақтағы тік гистограммалар арқылы анықтауға болады. Ұқсас әдістерді масштабты өзгергіштікті қамтамасыз ету үшін таңбаның биіктігін бағалау үшін қолдануға болады. Сонымен, екі өлшемді деректерді бір өлшемді деректерге түрлендіру әдістерін онлайн тану әдістеріне қолдануға болады.

Алдын-ала өңдеу әдістері мәтінді немесе графиканы қамтитын құжаттардың түрлі-түсті, жартылай тонды немесе бинарлы бейнелер үшін қажет. Таңбаларды тану жүйелерінде көптеген қосымшалар сұр немесе екілік кескіндерді пайдаланады, өйткені түрлі-түсті кескіндерді өңдеу үлкен есептеу ресурстарын қажет етеді. Мұндай суреттерде гетерогенді фон және белгілер болуы мүмкін, бұл құжат мәтінін кескіннен алдын-ала өңдеусіз шығаруды қиындатады. Алдын -ала өңдеудің қажетті нәтижесі – бұл тек мәтіннен тұратын екілік кескін. Сонымен, бұған жету үшін бірнеше қадамдар қажет, біріншіден, шуды жою немесе кескіннің контрастын түзету үшін суретті жақсартудың кейбір әдістері, екіншіден, кез-келген көріністер, артық белгілері және (немесе) шу бар фонды алып тастау үшін шекті мәндерді орнату, үшіншіден, мәтінді графикадан ажырату үшін бетті сегментациялау, төртіншіден, таңбаларды бір-бірінен ажырату үшін таңбаларды сегментациялау және соңында алдын-ала өңдеу әдістері таңбалардың бөліктерін бұлдыратады немесе оларға артық пиксель қосатын жағдайларда таңбаларды жақсарту үшін морфологиялық өңдеу. Жоғарыда аталған әдістер таңбаларды тану жүйелерінде және кейбір

қосымшаларда қолдануға болатын кейбір әдістер болып табылады. Осы әдістердің кейбірін OCR жүйесінің әртүрлі кезеңдерінде қолдануға болады. Енді таңбаларды тану жүйесін алдын-ала өңдеу кезеңінде қолданылатын кейбір әдістерге тоқталайық.

1.2.1 Бейнені кеңістіктік сүзу операциялары

Бейнені өңдеу кезінде сүзгілер негізінен суреттегі жоғары жиілікті, яғни кескінді тегістеу немесе төменгі жиілікті басу, яғни кескіннің жиектерін жақсарту немесе анықтау үшін қолданылады. Кескінді қалпына келтіру және жақсарту әдістері кеңістіктік және жиілік аймақтарында, яғни Фурье түрлендіруімен сипатталады. Алайда, Фурье түрлендіруі айтарлықтай есептеуді қажет етеді, ал кейбір жағдайларда күш салудың қажеті жоқ. Жиілік аймағындағы көбейту уақыт пен кеңістік аймақтарындағы жинақтауға сәйкес келеді. 3х3 сияқты кішкене жинақтау маскасын қолдану және осы масканы кескінге бүктеу Фурье түрлендірулері мен көбейту әдістеріне қарағанда әлдеқайда оңай және жылдам. Сондықтан бұл тарауда тек кеңістіктік сүзу әдістері ұсынылады. Жиі түсірілген суреттерге шу әсер етуі мүмкін. Алайда алынған суреттер талдау үшін қажетті суреттерді ұсынбауы мүмкін. Сонымен қатар, қолайлы сападағы суреттерде белгілі бір аймақтарды бөлектеу қажет болуы мүмкін. Кеңістіктік өңдеу нүктелік өңдеу және маска арқылы өңдеу болып бөлінеді. Нүктелік өңдеу кескіннің басқа пикселдеріне тәуелсіз жеке пикселдерді түрлендіруді қамтиды. Бұл қарапайым операциялар, әдетте, кескіндердің аппараттық ақауларын түзету үшін қолданылады, мысалы, шамадан тыс кескіндердің орнын толтыру үшін. Екінші жағынан, масканы өңдеген кезде пиксель квадрат немесе дөңгелек маскадағы пиксельдердің айналасымен жақсартылған кескінде *(x, y)* координаттарда пиксель құруға қатысады.

Нүктелік өңдеу (1) теңдеуінде көрсетілгендей жақсартылған кескінде сәйкес пиксель мәндерін құру үшін бастапқы кескіндегі пиксель мәндерін өзгертеді.

$$O(x, y) = T[I(x, y)] \qquad (1.1)$$

Мұндағы *I(x, y)* – бастапқы (кіріс) сурет, *O(x, y)* – өңделген сурет, ал *T* екі сурет арасындағы түрлендіруді сипаттайды. Нүктелік өңдеудің кейбір әдістеріне мыналар жатады: контрастты созу, жаһандық шекті өңдеу, гистограммаларды түзулеу. Маскаларды өңдеудің кейбір әдістеріне орташа сүзгілер, айқындықты жоғарылату сүзгілері, жергілікті шектер және т. б. жатад
ы.

Суреттің контраст деңгейі жарықтың нашар болуына немесе түсіру сенсорының дұрыс орнатылмауына байланысты өзгеруі мүмкін. Демек, кескінді алу қиындықтарын өтеу үшін кескіннің контрастын басқару қажет. Идея – суреттегі сұр деңгейлердің динамикалық диапазонын өзгерту. Бұл жағдайда жұмыс істей алатын әдіс сызықтық дисплей деп аталады, (1.2) теңдеу кескіннің барлық спектріндегі динамикалық диапазонды *0* - ден *(L-1)* - ге дейін кеңейту

арқылы төмен контрастты немесе жоғары контрастты кескіннің пиксель мәндерін арттыру үшін қолданылады:

$$O(x,y) = O_1 + \left(\frac{O_2 - O_1}{I_2 - I_1}\right)[I(x,y) - I_1] \quad (1.2)$$

мұндағы $O_1$ *0*-ге сәйкес келеді, ал $O_2$ қажетті деңгейлер санына сәйкес келеді (*L-1=255*). $I_1$ және $I_2$ сұр деңгейлердің кіріс диапазонының минималды және максималды мәндерін қамтамасыз етеді. Өңдеудің ең қарапайым түрі – кескіннің жарықтығын *B* барлық пиксель мәндеріне қосу арқылы реттеу. Мұндағы *b>0* кескіннің жарықтығын арттырады, ал *b <0* кескінді күңгірт етеді. Сонымен қатар, ығысудың орнына *A*-ны қолдануға болады, мұнда *A* көбейтіндісі кіріс пикселдерінің мәндеріне шығыс кескінінің жарықтығын өзгертеді. *0<a<1* мәндері қараңғы кескінге әкеледі, ал *a>1* мәндері жарық кескінге әкеледі. Орын ауыстыру мен күшейтуді біріктіру (1.3) теңдеуді береді:

$$O(x,y) = a * I(x,y) + b \quad (1.3)$$

Кескіннің шекті мәнін белгілеу – бұл кескіннің ақпаратын (нысандарын) оның фонынан бөлу процесі, сондықтан шекті мән әдетте сұр немесе түрлі-түсті құжат деңгейімен сканерленген құжат кескіндеріне қолданылады. Шекті мәндерді белгілеуді екі негізгі санатқа бөлуге болады: ғаламдық және жергілікті. Шекті мәндерді анықтаудың ғаламдық әдістері көбінесе сурет қарқындылығының гистограммасы бойынша фон деңгейін бағалауға негізделген құжаттың бүкіл кескіні үшін бір шекті мәнді таңдайды. Демек, бұл нүктелік өңдеу операциясы деп саналады. Екінші жағынан, жергілікті адаптивті шекті орнату жергілікті аймақ туралы ақпаратқа сәйкес әр пиксель үшін әртүрлі мәндерді қолданады. Әдебиетте жарияланған жүздеген шекті анықтау алгоритмдері бар және барлық әдістерді ұсыну үшін бірнеше кітап қажет болады, сондықтан мұндағы мақсат белгілі әдістерді ұсыну болып табылады.

Сұр кескінді екілік кескінге автоматты түрде түрлендіру үшін ғаламдық шекті анықтау әдістері қолданылады. Мұндай әдістерге қолданылатын суреттерде пиксельдің екі класы бар (алдыңғы және артқы(фон)). Шекті мәндерді анықтаудың ғаламдық әдісінің мақсаты *T* шекті мәнін автоматты түрде көрсету болып табылады, онда одан төмен пиксель мәндері алдыңғы және одан жоғары мәндер фон болып саналады. Қарапайым әдіс кіріс кескініндегі барлық пикселдердің орташа немесе медианасын таңдау болар еді, орташа немесе медианасы шекті мән ретінде жақсы жұмыс істейді, алайда олай емес, әсіресе пиксельдер кескінге біркелкі бөлінбесе. Неғұрлым күрделі тәсіл кескіннің пиксель қарқындылығының гистограммасын жасау және шекті мән ретінде қуыс нүктесін (минимумды) пайдалану болуы мүмкін. Гистограмма тәсілі фон мен объектінің пиксельдерінің орташа мәні бар деп болжайды, бірақ нақты пиксель мәндерінде осы орташа мәндерден кейбір ауытқулар бар. Алайда, бұл есептеу тұрғысынан қымбатқа түсуі мүмкін, ал суреттердің гистограммаларында депрессияның нақты нүктелері болмауы мүмкін, бұл көбінесе дәл шекті

таңдауды қиындатады. Кескін туралы көп білімді қажет етпейтін салыстырмалы түрде қарапайым әдістердің бірі – итерациялық әдіс [19]:

1-қадам. Бастапқы шекті мәнді $T$ кездейсоқ немесе суреттегі пиксельдің орташа немесе медианалық мәні сияқты кез-келген әдіске сәйкес таңдау.

2-қадам: $T$ көмегімен суретті объект пен фонның пикселдеріне бөлу. $R_1$ (фон аймағы) қарқындылығы $\geq T$ мәні бар пиксельдерден тұрады, ал $R_2$ (объектілер аймағы) қарқындылығы $< T$ пиксельдерден тұрады.

3-қадам: сәйкесінше $R_1$ және $R_2$ аудандары үшін әр аймақтың, $\mu_1$ және $\mu_2$ орташа мәнін есептеу.

4-қадам: 1.4 теңдеуде көрсетілгендей жаңа $T$ шекті мәнін есептеу

$$T = 1/2(\mu_1 + \mu_2) \tag{1.4}$$

5-қадам: 2-ден 4-ке дейінгі қадамдарды жаңа $T$ көмегімен соңғы мән алдыңғыға сәйкес келгенше қайталау.

Әдебиеттерде шекті мәндерді анықтаудың көптеген әдістері бар, мысалы [20, 21]. Біркелкі шаралар немесе пішін көрсеткіштерін қолдана отырып, шекті анықтаудың 20-дан астам жаһандық алгоритмдерінің жұмысын салыстырды. Салыстыру Оцу кластарға бөлу әдісі жақсы нәтиже беретіндігін көрсетті [22]. Триер мен Джейн жүргізген OCR мақсатты бағалау зерттеулерінің аясында Оцу әдісі зерттелген басқа әдістерден асып түсетінін көрсететін төрт жаһандық әдіс зерттелді [23]. Сонымен қатар, [24] әдістерді салыстыру нәтижесінде Оцу әдісі құжат кескіндерін өңдеуде қолайлы екенін растады. Оцу әдісі – сұр деңгейдегі кескінді екілік кескінге түрлендіру үшін қолданылатын кең қолданылатын әдістердің бірі, клас ішіндегі дисперсия минималды болатындай оңтайлы шекті есептейді. Оцу әдісі кластар арасындағы дисперсияны азайту үшін шекті мәнді есептейді, ол 1.5 теңдеуде көрсетілгендей екі кластың дисперсияларының өлшенген қосындысына тең [25]:

$$\sigma_w^2(t) = w_1(t)\sigma_1^2(t) + w_2(t)\sigma_2^2(t) \tag{1.5}$$

мұндағы $\omega_i$ – бұл $t$ шегі бойынша бөлінген екі кластың ықтималдықтары, ал $\sigma_i^2$ – бұл кластардың дисперсиясы. Оцу класс ішіндегі дисперсияны азайту класс аралық дисперсияны максимизациялаумен бірдей екенін көрсетеді, келесі 1.6 теңдеу:

$$\sigma_w^2(t) = \sigma^2 - \sigma_2^2(t) = w_1(t)w_2(t)[\mu_1(t) - \mu_2(t)]^2 \tag{1.6}$$

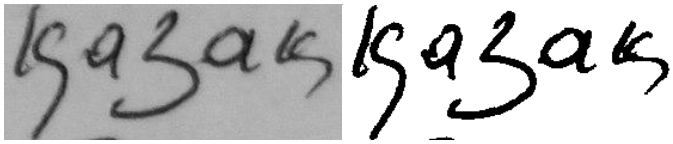
бұл кластардың ықтималдықтары $\omega_i$ мен итеративті түрде жаңартылып отыратын $\mu_i$ орта мәнмен өрнектеледі.

Алгоритм қадамдары:

- Әр қарқындылық деңгейінің гистограммасы мен ықтималдығын есептеу
- $\omega_i(0)$ және $\mu_i(0)$ инициализациялау

- Максималды қарқындылыққа дейін барлық шекті мәндер *t = 1....* бойынша жүреміз
  - $\omega_i(0)$ және $\mu_i(0)$ жаңарту
  - Қажетті шекті деңгейге сәйкес келетін максимум $\sigma_2^2(t)$ мәнін есептеу.

1.11-суретте алгоритмнің нәтижесі көрсетілген.



Сурет 1.11 – Оцу алгоритмінің нәтижесі

1.2.2 Гистограмманы өңдеу

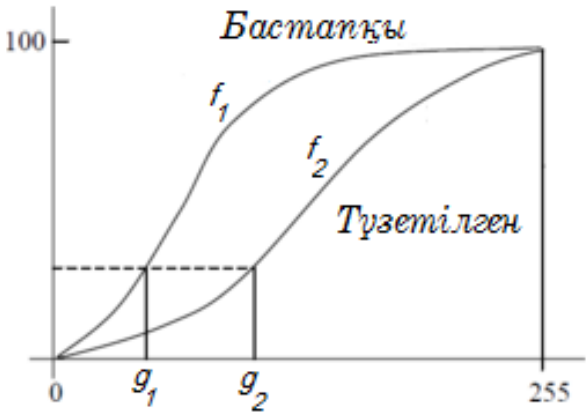
Гистограмманы өңдеу кескінді жақсарту үшін қолданылады және кескінді сығымдау және сегментациялау кезінде пайдалы болуы мүмкін. Гистограмма әр сұр деңгейдің 0-ден (қара) 255-ке (ақ) дейінгі жиілігін көрсетеді. Сканерленген немесе түсірілген суреттер түстердің шектеулі диапазонына ие болуы мүмкін немесе контраст жеткіліксіз болуы мүмкін. Гистограмманы өңдеу суретті жақсартуға мүмкіндік береді, бірақ сонымен қатар сегментация сияқты машиналық көрудің басқа жұмыстарына да көмектеседі. Осылайша, гистограмманы өңдеу алдын-ала өңдеудің бастапқы кезеңі болуы керек. Гистограмманы түзеу және сәйкестендіру – бұл жақсырақ кескін алу үшін кескін гистограммасын өзгерту үшін кеңінен қолданылатын екі әдіс.

Гистограмманы түзеу жаһандық әдіс болып саналады [26]. Ол гистограмманы пиксельдің барлық спектріне созады (0-255). Бұл адамның түпкілікті түсінуі үшін кескіндердің контрастын арттырады және оны кескінді түсіну кезінде жарықтың өзгеруін қалыпқа келтіру үшін қолдануға болады. Бұл процесс өте қарапайым, бастапқы сурет жарықтығының әр *j* деңгейі үшін жаңа пиксель деңгейінің *k* жаңа мәні (1.7)-теңдеуінде көрсетілгендей есептеледі:

$$k = \sum_{i=0}^{j} N_i/T \qquad (1.7)$$

мұндағы сумма суреттегі (гистограмманы біріктіру арқылы) жарықтылық мәні *j*-ге тең немесе одан аз пиксельдер саны, ал *T* – пикселдердің жалпы саны. Сонымен қатар, гистограмманы түзеу – бұл гистограмманың спецификациясы немесе модификациясы негізінде жаңа кескіндерді алу үшін қолданылатын амалдардың бірі.

Гистограммаларды сәйкестендіру – бұл екі суреттің түсін олардың гистограммаларын қолдана отырып түзету үшін суретті өңдеу әдісі.




Сурет 1.12 – Эталондық және түзетілген кескіндер үшін кумулятивті тарату функциялары

Гистограммаларды сәйкестендіру бұл бейімделетін суреттің $f_2$ үлестіру функциясының эталондық кескіннің $f_1$ кумулятивті үлестіру функциясына (CDF) сәйкестендіру [27]. Гистограмманы өзгерту алдымен екі суреттің гистограммаларын есептеу арқылы жүзеге асырылады, содан кейін эталондық және түзетілетін суреттердің CDF есептейді. Нәтижесінде $f_2$-ден $f_1$-ге ең жақын CDF сәйкестендіру арқылы алынады. Содан кейін 2-суретте көрсетілгендей әрбір суреттің сұр деңгейлері үшін $f_1(g_1) = f_2(g_2)$ есептелінеді және гистограмманы сәйкестендіру функциясының нәтижесі $M(g_1) = g_2$.

1.2.3 Шуды жою

Технология саласындағы жетістіктер сапалы бейне алу үшін жетілдірілген құрылғылардың пайда болуына әкелді. Қазіргі заманғы технологиялар әр түрлі электро-оптикалық құрылғылармен байланысты шу деңгейін шамалы деңгейге дейін төмендетуге мүмкіндік бергенімен, әлі де жойылмайтын кейбір шу көздері бар. Заманауи сенсорлар арқылы алынған суреттер әртүрлі шу көздерімен ластануы мүмкін. Шу астында біз көлеңке немесе фокустың болмауы сияқты детерминистік бұрмалануларға қарағанда стохастикалық вариацияларды түсінеміз. Электрондық түсіру құрылғыларымен немесе қолданылатын жарық көзімен байланысты шудың әртүрлі түрлері бар, олар фотон, жылу, кристалға салынған электронды және кванттық әсерлерден болатын шулар. Шудың көп бөлігін түсіру сенсорлары немесе CCD (charge-coupled device) камералары арқылы жоюға болады.

Құжаттарды талдау жүйелері алдын-ала өңдеу кезеңіндегі шуды азайту есебінен жақсарады, бұл OCR жүйесіндегі белгілерді алу және тану кезеңдерінің сенімділігі мен тұрақтылығын едәуір арттыруға мүмкіндік береді. Екілік бейнелердегі шудың әдеттегі көрінісі оқшауланған пикселдерден құралады, оны "тұз және бұрыш" шуы деп те атайды. Шудың бұл түрін алып тастау әдісі тұз бен бұрыштың әр пиксельдік "аралын" қоршаған "теңізбен" толтыру деп аталады [28]. Сұр деңгейдегі немесе орташа сүзгілері бар суреттерде және гаусстың орташа немесе бұлыңғыр сүзгілері сияқты төмен өту сүзгілерінде олар

оқшауланған пиксель шуын жоятыны дәлелденді. Гаусстың бұлыңғырлығы және орташа сүзгілер – бұл кескіннің тегіс құрылымын қамтамасыз ету үшін ең жақсы таңдау. Екінші жағынан, Фурье спектрінде жиі көрінетін импульстік жарылыстар түрінде пайда болатын мерзімді шуды кесу фильтрациясы арқылы сүзуге болады [29]. *n, H(u, v)* ретті Баттерворт кесу сүзгісінің беріліс функциясы 1.8 теңдеуімен берілген:

$$H(u,v) = \frac{1}{1+\left[\frac{D_0^2}{D_1(u,v)D_2(u,v)}\right]^n} \quad (1.8)$$

Мұндағы

$$D_1(u,v) = \sqrt{\left[(u-\frac{M}{2}-u_0)^2+(v-\frac{N}{2}-v_0)^2\right]} \quad (1.9)$$

және

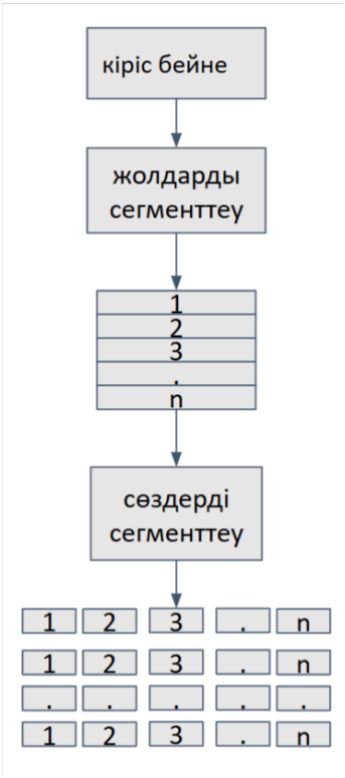
$$D_1(u,v) = \sqrt{\left[(u-\frac{M}{2}-u_0)^2+(v-\frac{N}{2}-v_0)^2\right]} \quad (1.10)$$

мұндағы $\mu_0, \upsilon_0$ және $-\mu_0, -\upsilon_0$ симметрияға байланысты ойықтардың орналасуы, ал $D$ – олардың радиусы. Сүзгі жиілік тіктөртбұрышының центріне қатысты орнатылады.

# 2 ҚАЗАҚ ҚОЛЖАЗБА МӘТІНДЕРІН ТАНУДЫҢ МАШИНАЛЫҚ ОҚЫТУ АЛГОРИТМІН ҚҰРУ

## 2.1 Қолжазба мәтіндерді сегменттеу әдістері

Қолжазба мәтіндерді тану процесінде сегменттеу – ең алғашқы және маңызды кезеңдердің бірі. Бұл кезеңнің негізгі мақсаты – бейнедегі мәтін элементтерін (жолдар, сөздер, таңбалар) бөлу арқылы оларды кейінгі тану модельдеріне қолайлы пішінге келтіру. Қолжазбалар көбінесе еркін түрде, әртүрлі шектеулі емес жоларалық аралықтармен және фонның біркелкі еместігімен жазылатындықтан, оларды бірден сөз немесе таңба деңгейінде дұрыс сегменттеу қиын [30]. Сондықтан, сегменттеу кезеңін иерархиялық түрде – алдымен жолдарды, содан кейін сөздерді, соңында таңбаларды бөліп шығару – тиімді және дәл шешім болып табылады.




Сурет 2.1 – Жолдар мен сөздерді сегменттеу алгоритмі

Қолжазба мәтін көбінесе парақ бетінде бірнеше жолдан тұрады. Егер бірден сөздерді немесе таңбаларды бөліп алуға тырыссақ, бұл мәтіндегі жолдардың қисықтығы мен әртүрлі тығыздық салдарынан жиі қате нәтижеге алып келеді. Алдымен жолдарды анықтау – мәтін құрылымын біршама ретке келтіріп, әрі қарайғы сегментацияны жеңілдетеді.

Жолдар – мәтіннің мағыналық блоктары. Жол бойынша сегменттелген мәтіндерде сөздер бір-біріне қатысты дұрыс орналасады, бұл кейінгі сөздік және таңбалық сегментацияда контекстті дұрыс сақтауға мүмкіндік береді. Мысалы, кейбір сөздер жол соңында бөлініп немесе жолдар бір-бірімен қабаттасып кеткен жағдайда жол сегментациясы бұл қиындықтарды локализациялауға көмектеседі.

Жолдарды алдын ала бөліп алу арқылы мәтін бөліктері кішірейіп, әрбір жолға жеке талдау жасау мүмкіндігі пайда болады, бұл жалпы қателіктер санын едәуір азайтады.

2.1.1 Жолдар мен сөздерді сегменттеу алгоритмі

Құжат кескініндегі әр мәтін жолының ауданын анықтау процесі болып табылатын мәтін жолының сегментациясы қолжазба мәтінін тану процесінің маңызды кезеңдерінің бірі болып табылады. Осы кезеңде алынған сапасыз нәтижелер қолжазбаны тану процедурасының дәлдігіне айтарлықтай әсер етеді. Мәтіндік жолдарды сегменттеу әдісімен шешілуі керек бірнеше мәселелер бар. Бұл мәселелерге;

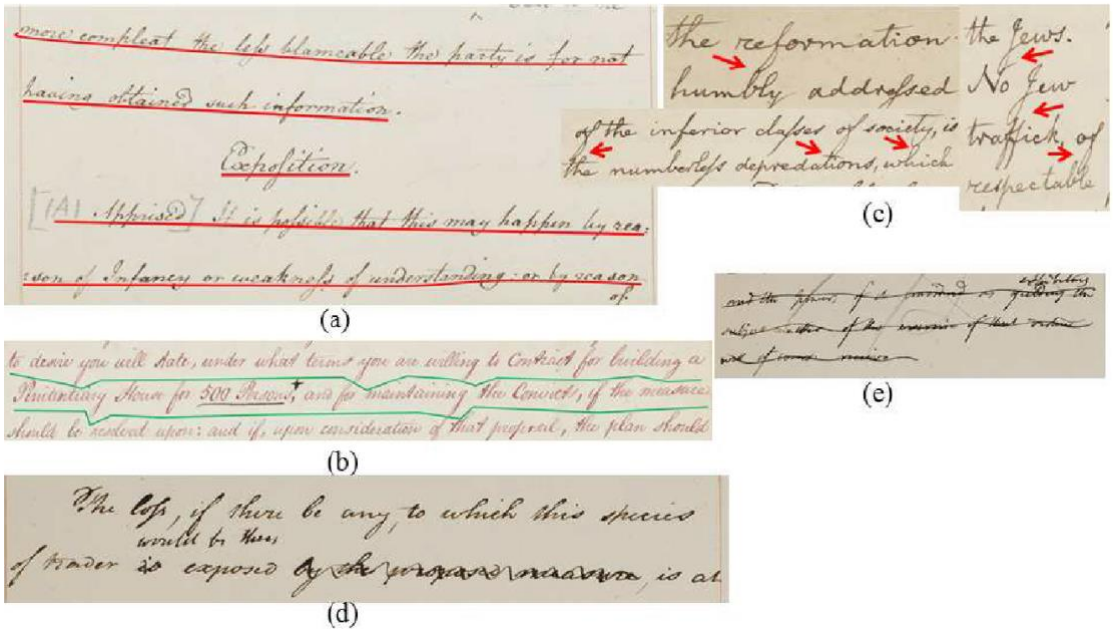
а) беттегі жолдар арасындағы немесе тіпті сол мәтіндік жол бойындағы көлбеу бұрыштағы айырмашылық

б) қабаттасатын және жанасатын мәтіндік жолдар

в) мәтін жолының үстіндегі толықтырулар және

г) жойылған мәтін жатады.

2.2-суретте осы мәселелердің әрқайсысының бір мысалы келтірілген.




Сурет 2.2 – Мәтіндік жолдарды сегментациялау үшін қолжазба құжаттарының суреттерінің алдында тұрған мәселелер. (a) беттегі жолдар арасындағы немесе тіпті сол мәтіндік жол бойындағы көлбеу бұрыштағы айырмашылық, (b) қабаттасатын мәтіндік жолдар, (c) мәтіндік жолдарды түрту, (d) мәтіндік жолдың үстіне қосу, e) жойылған мәтін

Қолжазба мәтінді сегменттеуде дәстүрлі бейнелік өңдеу әдістері – мысалы, проекция профилі, шекті мән арқылы бинаризация, контур табу немесе Connected Components Analysis – кейбір қарапайым жағдайларда жұмыс істегенімен, күрделі фон, бұрмаланған жолдар немесе әртүрлі жазу стилі сияқты

факторлар әсер еткен кезде өз тиімділігін жоғалтады. Мұндай әдістер қолжазба мәтіннің контекстін ескермейтін болғандықтан, жиі жағдайда қате сегментацияға алып келеді.

Осыған байланысты соңғы жылдары терең нейрондық желілерге негізделген сегменттеу әдістері қарқынды дамып, нақты әрі бейімді шешім ретінде ұсынылып келеді. Бұл модельдер мәтін құрылымын кең контекстте қарастырып, күрделі жағдайларда да жоғары дәлдікпен сегменттеу нәтижесін береді.

Жолдар мен сөздердің сегментациясы бұл семантикалық сегменттеу түріне жатады, яғни кескіндегі әрбір пикселді объектілердің белгілі бір класына жіктеуге бағытталған компьютерлік көру тапсырмасы. Классификациядан (суретке бір класс тағайындау) және анықтаудан (нысандар локализацияланған жерде) айырмашылығы, семантикалық сегменттеу көрініс пиксель деңгейінде түсінуді талап етеді.

Семантикалық сегменттеу есебі - берілген $I \in R^{HxWxC}$ кескініндегі әрбір *i,j* пикселге К классардың бірін тағайындайтын

$$f_\theta : R^{HxWxC} \rightarrow \{1, \dots, K\} \qquad (2.1)$$

функциясын оқыту. Мұндағы *H* мен *W* кескіннің биікігі мен ұзындығы, *C* – каналдар саны, *K* – кластар саны (біздің жағдайда К=1). Нәтижесінде $S \in \{1, \dots, K\}^{HxW}$ сегментация маскасы пикселдің қай класқа жататынын анықтайды:

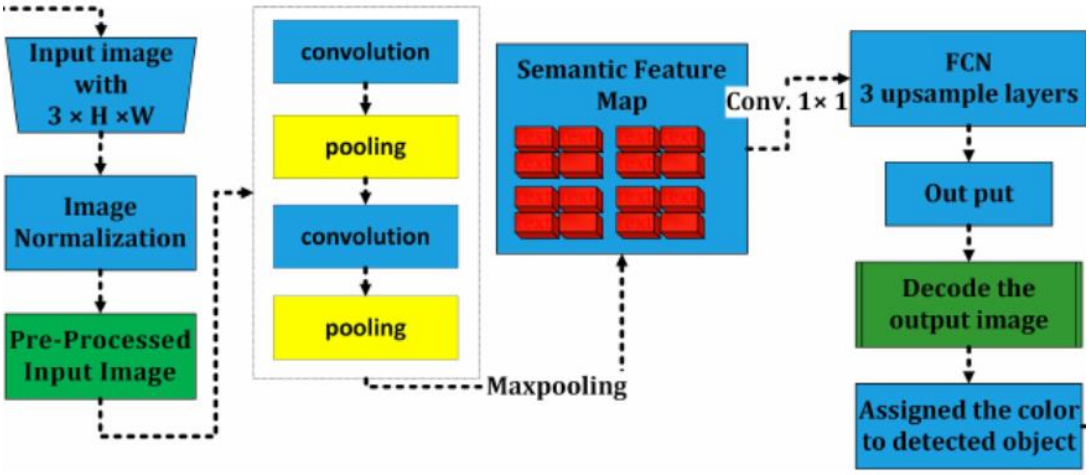
$$S_{i,j} = k \rightarrow (i, j) \qquad (2.2)$$

[31] зерттеуінде классикалық алгоритмдер Thresholding, K-means мен терең оқыту модельдері U-Net, FCN-8, SegNet салыстырылды. Зерттеу нәтижесінде классикалық әдістердің дәлдігі 80 %–ке жуық, ал Dice коэффиценті 58–67 % аралығында болғаны байқалды. Ал U-Net және FCN-8 секілді терең модельдер 99 % дейін дәлдікке және 98–99 % Dice мәніне жеткені көрсетілді.

Ұқсас тұжырымды [32] өз еңбегінде де келтірген: олар классикалық және терең оқыту негізіндегі бейне сегментация әдістерін шолып, соңғыларының шуға төзімділігі мен контурларды нақтылау қабілетін ерекше атап өтті.

Осы шолуларға сүйене отырып, біз кеңінен қолданылатын және танылған үш сегменттеу архитектурасын таңдадық: FCN (толық конволюционды желі), U-Net және DeepLabv3.

FCN – семантикалық сегментацияға арналған алғашқы толығымен сверткалық желі. Бұл модель деконволюция арқылы пиксель деңгейінде нәтиже алуға мүмкіндік береді және қарапайым құрылымы арқасында жылдам жұмыс істейді. Оның басты кемшілігі - ұсақ детальдар мен контурларды дәл ажырата алмауы [33].

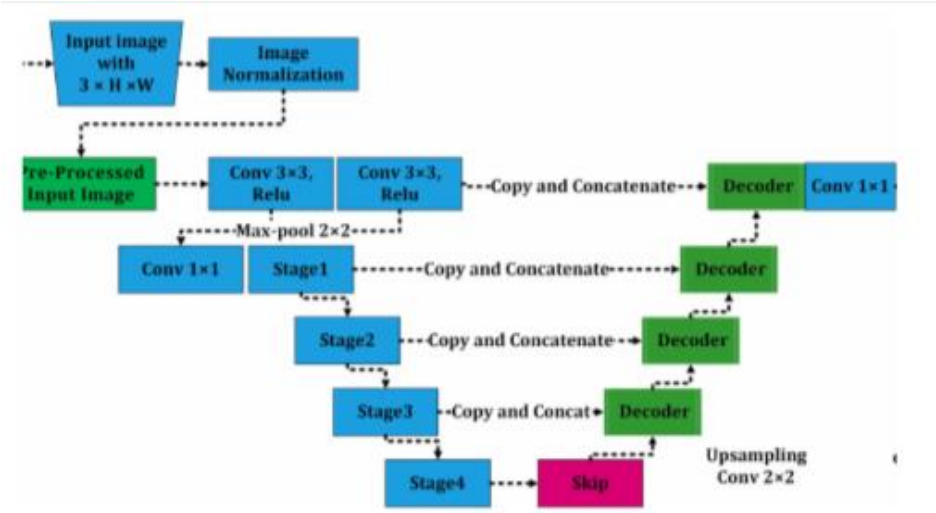



Сурет 2.3 – FCN архитектурасы

FCN архитектурасы бірнеше бөліктерден тұрады (2.3-сурет):

- Кескінді енгізу және қалыпқа келтіру кіріс ретінде түсті кескінді (3 × H × W) алады. Кескінді қалыпқа келтіру өңдеуге дайындау үшін пиксель мәндерін стандарттайды.
- Сипаттарды шығару - конволюция және біріктіру қабаттары иерархиялық керекті сипаттарды шығарады. Шығарылым кескін мазмұнының ықшам көрінісі болып табылады.
- Семантикалық сипаттар картасы - кескін туралы жоғары деңгейдегі дерексіз ақпаратты түсіреді. Ол пиксель бойынша жіктеу үшін қолданылады.
- Үлгілеу қабаттары мүмкіндік картасын бастапқы кескін өлшеміне қалпына келтіреді. Олар сипаттарды ұсақ түйіршікті кеңістік өлшемдеріне қайта проекциялау арқылы тығыз болжам жасауға мүмкіндік береді.
- Соңғы нәтиже сегменттеу картасы болып табылады, онда әрбір пиксел класспен белгіленген. Бұл карта әртүрлі анықталған нысандарды көрсету үшін түсті пайдаланып визуалдайды.

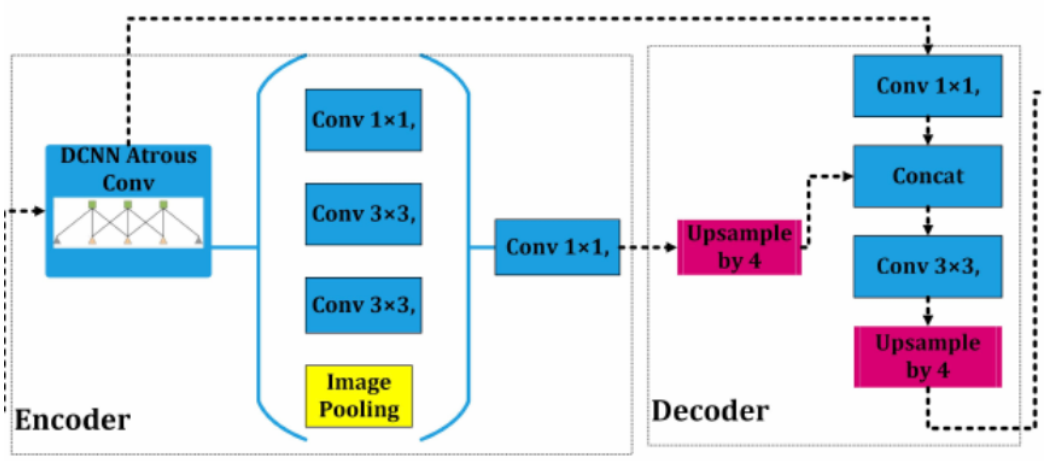
U-Net – skip-connection арқылы төмен деңгейлі ақпаратты қалпына келтіретін encoder–decoder архитектурасы. Бұл модель қолжазба жолдарын дәл локализациялау үшін тиімді және көптеген HTR зерттеулерінде қолданылады [34].




Сурет 2.4 – U-net архитектурасы

- Кіріс кескіні алдын ала қалыпқа келтіру арқылы желіге беріледі. Бұл жаттығуды жылдамдату және тұрақтандыру үшін пиксель мәндерінің біркелкі масштабталуын қамтамасыз етеді. Кескін өлшемі ерікті болуы мүмкін, бірақ көбінесе ол екіге еселік болады.
- Қайталанатын конволюция және макспулинг блоктарынан тұрады. Әрбір блок сипаттарды шығарып, кеңістіктік ажыратымдылықты біртіндеп азайтады. Бұл желіге әртүрлі деңгейлерде семантикалық мағыналы ақпаратты түсіруге мүмкіндік береді.
- Кодер мен дешифраторды қосатын желінің орталық бөлігі. Мұнда сипаттар ең төменгі кеңістіктік ажыратымдылыққа және ең үлкен тереңдікке ие. Ол талдаудан құрылымдық қайта құруға көшу үшін түйін ретінде әрекет етеді.
- Кеңістіктік ажыратымдылықты бірте-бірте қалпына келтіруді жоғары үлгілеуді (әдетте транспозицияланған конволюциялар) пайдалана отырып орындайды.
- Соңғы қабат әрбір пиксельді сынып кеңістігіне проекциялайтын 1×1 конволюциясы болып табылады. Нәтиже - әрбір пикселге класс тағайындалған кіріспен бірдей өлшемдегі сегменттеу маскасы.

DeepLabv3 – dilated convolutions мен Atrous Spatial Pyramid Pooling (ASPP) арқылы әртүрлі масштабтағы кең контекстті меңгереді. Бұл модель күрделі фон, қисық жазу немесе әртүрлі тығыздық жағдайында да жоғары дәлдік көрсетеді [35].




Сурет 2.5 – DeepLabv3 архитектурасы

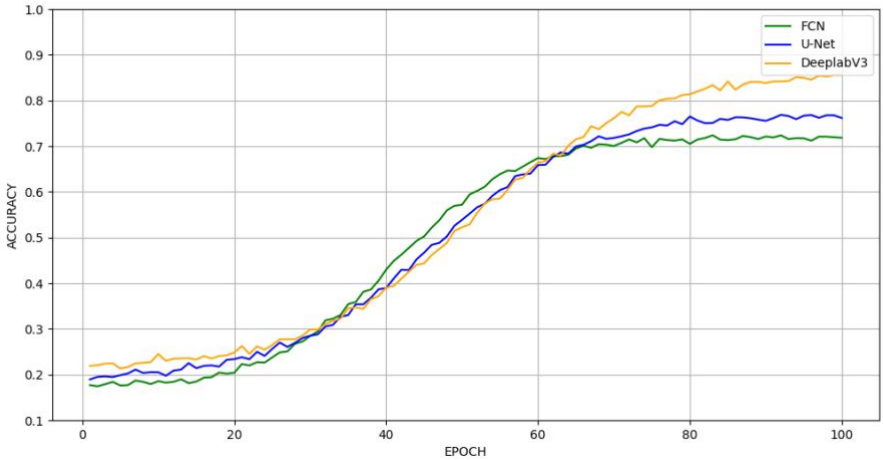
Бұл архитектураның ерекшелігі – FCN және U-Net жүйелерінен ажыратымдылықты жоғалтпай көру өрісін ұлғайтуға мүмкіндік беретін кеңейтілген конволюцияларды қолдануы. U-Net-тен айырмашылығы, DeepLabv3 үлгілеуі бар симметриялық декодерді пайдаланбайды, бірақ ASPP (Atrous Spatial Pyramid Pooling) модулі арқылы көп деңгейлі мүмкіндіктерді біріктіру арқылы мәліметтерді қалпына келтіреді. Бұл әртүрлі масштабтағы объектілерді тиімді өңдеуге және кең контекстті қарастыруға мүмкіндік береді. Архитектура әсіресе күрделі фондық және бұлыңғыр нысан шекаралары бар кескіндерде жақсы жұмыс істейді.

### 2.1.2 Сегменттеу нейрондық желілерін оқыту және бағалау

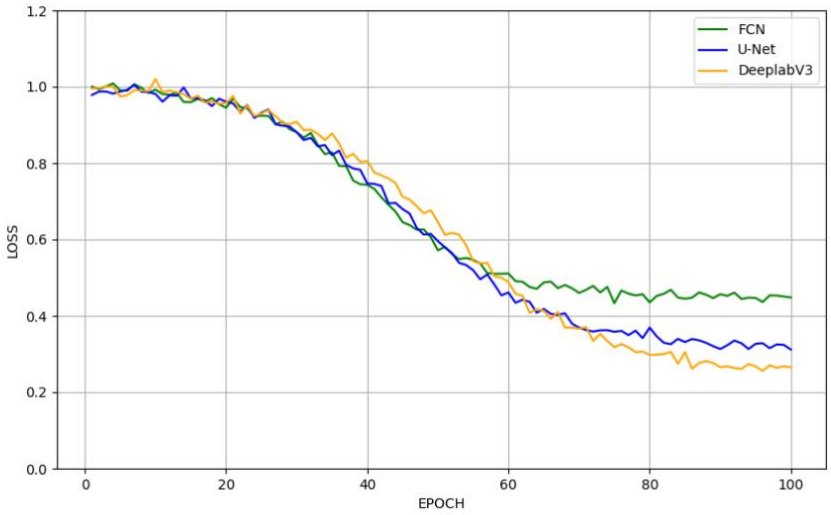
Таңдалған архиекуралардың сипаттамалары 2.1-кестеде берілген. Бұл архитектуралар жолдар мен сөздер аннотацияларын қолданып жеке оқытылды. Жолды сегменттеу үлгілерін үйрету және бағалау үшін қолжазба құжаттарының 745 белгіленген кескінін қамтитын деректер жинағы пайдаланылды. Компьютерлік көру тапсырмаларындағы жалпы қабылданған тәжірибеге сәйкес үлгі 80:20 қатынасында оқыту және валидация қосалқы үлгілеріне бөлінді.

Кесте 2.1 – Модельдердің сипаттамалары

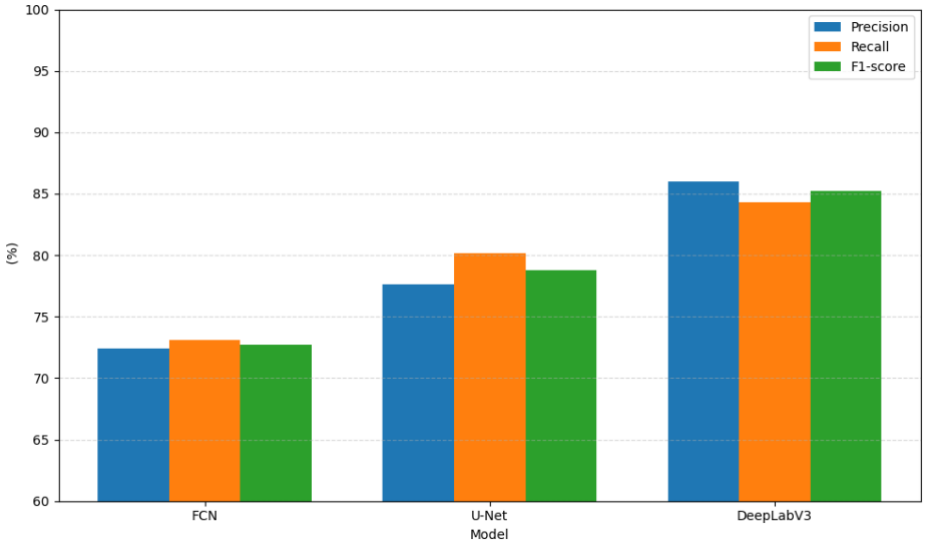
| **Архитектура** | **Қабаттар саны** | **Параметрлер саны** |
|---|---|---|
| **FCN** | 25 | 23.6M |
| **U-Net** | 28 | 31M |
| **DeepLabv3** | 53 | 58M |




Сурет 2.6 – Оқытылу кезіндегі дәлдік




Сурет 2.7 – Оқытылу кезіндегі қателік



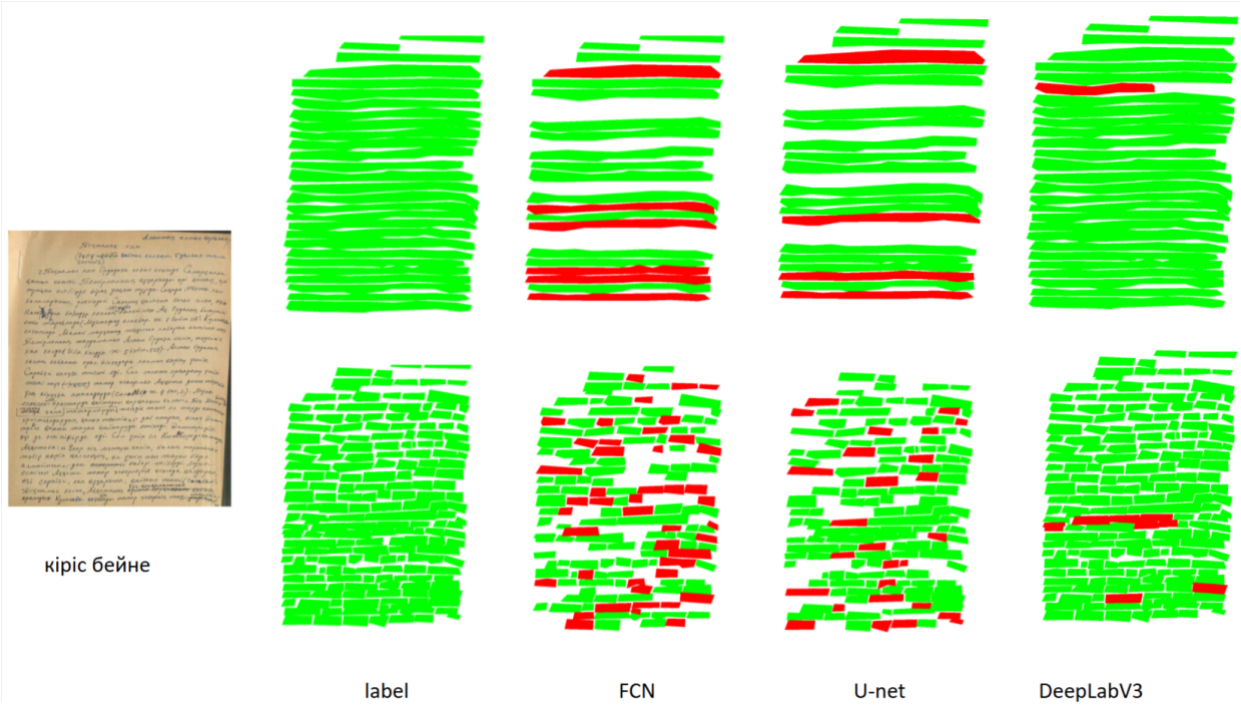

Сурет 2.8 – Модельдердің сапасының көрсеткіші

Оқыту нәтижелерінің метрикалар бойынша көрсеткіштері 2.8 – суретте көрсетілгендей DeeplabV3 Precision ≈ 85.5%, Recall ≈ 84%, F1–score ≈ 84.7% мәндерімне басқаларына қарағанда жақсы нәтиже көрсетті.

Сөздер үшін де модельдер алдыңғыдай жолмен оқытылды және келесідей нәтижелер алынды (2.2-кесте):

Кесте 2.2 – Сөзді сегменттеу нәтижелері

| **Модель** | **Precision (%)** | **Recall (%)** | **F1-score (%)** |
|---|---|---|---|
| FCN | 54.3 | 77.5 | 63.8 |
| U-net | 64.1 | 78.2 | 70.4 |
| DeepLabV3 | **70.02** | **82.1** | **75.5** |




Сурет 2.9 - Сегменттеу алгоритмдерінің нәтижесі

2.9-суретте үш сегменттеу моделінің көрнекі нәтижелері көрсетілген. Жоғарғы қатарда жолды сегменттеу нәтижелері, ал төменгі қатарда сөзді сегменттеу нәтижелері көрсетіледі. Дұрыс болжанған аймақтар жасыл түспен, қателер (жалған оң және жалған теріс пикселдер) қызыл түспен көрсетіледі. Нәтижелер сандық көрсеткіштерді растайды: DeepLabV3 жолды сегменттеу тапсырмасында да, сөзді сегменттеу тапсырмасында да FCN және U-Net жүйелерінен дәйекті түрде асып түседі.

## 2.2 Қолжазба мәтінді тану алгоритмдеріне шолу

Бұл бөлімде қолжазба мәтінді тану мәселесін шешудің әртүрлі тәсілдері қарастырылады. Қарастырылып отырған мәселе соңғы жиырма жылда белсенді түрде зерттеліп келеді. Сондықтан бұл бөлімде әртүрлі кезеңдерде пайда болған тәсілдер, бірнеше көзқарастар мен алгоритмдер қарастырылады.

Қолжазба мәтінін автоматты түрде тану (HTR) мәселесі құжатты цифрлаудан кейін басталады. Мәтінді тану адамдар үшін қарапайым тапсырма, бірақ ол автоматты жүйелер үшін күрделі екені дәлелденді. Шындығында, ол шешілмеген мәселе болып саналады және белсенді зерттеулерде [36, 37] қарастырылған. Жоғары вариативтілік және қолжазба мәтінінің курсивті сипаты - бұл мәселенің негізгі қиындықтары. Бұл қиындықтар тарихи тұрғыдан қолжазбаны тану технологияларының офлайнды практикалық қолдануларының айтарлықтай шектеулі болғанын білдірді.

Алғашқы офлайн HTR модельдері оқшауланған таңбаларды тануға бағытталған [38]. MNIST дерекқоры 0-ден 9-ға дейінгі қолмен жазылған цифрлардың бекітілген өлшемді кескіндерінен тұратыны жақсы белгілі [39]. 1998 жылы құрылған бұл деректер базасы әлі де сызықты емес модельдер үшін проблемалық эталон ретінде кеңінен қолданылады.

Әртүрлі жалпы HTR үлгілері әдетте үш қадамнан тұрады және әрқайсысы үшін әртүрлі тәсілдер қарастырылады. Біріншіден, қолжазба кескінінен сипаттарын алу стратегиясы. Екіншіден, осы сипаттарын шығыс сигналға түрлендіру моделі. Үшіншіден, кіріс кескінінде пайда болатын мәтінді болжау үшін шығыс сигналын декодтау тәсілі.

Бастапқыда кескін сипаттарын шығару стратегиялары өте күрделі болды, өйткені тану үлгілері кескін пикселінің мәндерін тікелей талдау үшін жеткілікті болмады. Мысалы, белгіленген өлшемді пиксельдер бойынша құрамдастарды шығару үшін негізгі құрамдас талдауды (PCA) қолданды [40]. Сонымен қатар бұл жағдайда бағытталған градиенттердің гистограммалары (HOG) көмегімен мүмкіндіктерді шығарды.

Рекурентті нейрондық желілерден (RNN) және конволюционды нейрондық желілерден (CNN) тұратын терең нейрондық желілер (DN) архитектурасы бар күрделі модельдер пайда болғаннан кейін айтарлықтай нәтижелер алына бастады. RNN мен CNN пайдалану сипаттарды шығаруды қажетсіз етеді және пиксельдік сипаттары тікелей қолжазба үлгісін енгізу ретінде пайдаланылады. Модельге кіріс ретінде кескіндегі пикселдерді тікелей пайдалану соңғы бірнеше жылда стандартқа айналды және бұл осы диссертацияда таңдалған стратегия.

Екінші компонент, тану моделі, HTR мәселесінің әрбір шешімінің өзегі болып табылады. Бұл модельдер ең дамыған және зерттелген модельдер болып табылады және осы модельдердің эволюциясы (және негізгі үлестері) келесі бөлімдерде егжей-тегжейлі сипатталған. Бұл соңғы жылдардағы нейрондық модельдердің маңызды дамуымен және терең архитектураның пайда болуымен байланысты. HTR мәселесінің архитектурасындағы прогресс көбінесе компьютерлік көру, сөйлеуді тану немесе тілді модельдеу саласындағы есептерде қолданылады.

2012-2013 жылдарға дейін қолжазбаны тану мәселесіне тәсілдер әдетте пәндік сараптаманы қажет ететін суреттерден мәтіндік мүмкіндіктерді құруға, содан кейін жасырын Марков үлгілерін (HMM) пайдалана отырып модельдеуге негізделген. Нейрондық желіні қолдану HMM-тің дұрыс таңбалар тізбегін декодтау мүмкіндігін жақсартады. Нейрондық желілер сәйкес кескін сипаттарын шығару немесе оларды транскрипцияларды туралау және декодтау үшін HMM пайдалану жолындағы алдыңғы қадам ретінде түрлендіру үшін пайдаланылады. Бұл әдетте HMM дұрыс таңбалар тізбегін декодтау мүмкіндігін жақсартады.

Терең нейрондық желілер компьютерлік көру мәселелерін шешуде табысқа жетуінің арқасында танымал болды және компьютерлердің есептеу қабілетінің артуына байланысты болды. Олардың жоғары танымалдығы сонымен қатар параметрлердің жоғары саны мен бірнеше қабаттары бар нейрондық модельдерді тиімді және қарапайым іске асыруға мүмкіндік беретін бағдарламалық жасақтама құрылымдарының пайда болуымен байланысты.

Содан бері нейрондық желілер офлайн қолжазбаны тану мәселесін шешудің ең жиі қолданылатын әдістеріне айналды. Әртүрлі авторлар бірнеше архитектураларды ұсынады және бұл мәселені шешу үшін әртүрлі желілер түрлерін пайдаланады. Келесі бөлімдерде біз қолжазбаны тану мәселесін шешу үшін осы әртүрлі нейрондық желілер архитектураларын пайдалануды егжей-тегжейлі сипаттаймыз. Бірақ алдымен біз HTR мәселесіне қолданылатын нейрондық желілердің ең өзекті тұжырымдамаларын қарастырамыз.

Мидың нейрондарының жұмысын имитациялайтын математикалық модель ретінде нейрондық желілердің алғашқы пайда болуын 1947 жылы Маккаллох пен Питтс ұсынды [41]. Бұл авторлар есептеу тапсырмаларын орындай алатын схема ретінде жеңілдетілген нейрондардың моделін енгізді. Нейрондық желілер 1969 жылға дейін Мински мен Паперт өздерінің Perceptrons [42] кітабында мұндай модельдердің бірнеше мәселелері мен шектеулерін ашқанға дейін қызығушылық таныта бастады.

1980 жылдарға дейін оларды оқыту үшін кері таралу әдісі (backpropagation) қолдану арқылы нейрондық желілер қайтадан ғылыми қоғамдастықтың назарын аударды. Кері таралу техникасын 1960 және 1974 жылдары бірнеше авторлар ұсынған. Пол Вербос өзінің тезисінде оны нейрондық желілерде пайдалануды ұсынды [43]. Бірақ ол 1986 жылы Дэвид Румельхарт, Джеффри Хинтон және Рональд Уильямстың екі мақаласын жариялағанға дейін танымал бола алмады: «Қателерді кері тарату арқылы оқыту көріністері» [44], ол бүгінгі күні белгілі болғандай оның практикалық іске асырылуын нақты түсіндіреді және «Қателерді тарату арқылы ішкі көріністерді үйрену» [45] бұл Мински 20 жыл бұрын дәлелдеген проблемаларды қарастырады және күрделі мәселелерді шешу үшін нейрондық желілерді қалай сәтті оқытуға болатындығын көрсетеді. Тек бірнеше жылдан кейін және осы екі мақалада ұсынылған әдістерді қолдану арқылы нейрондық желілердің алғашқы практикалық қолданбаларының бірі қолмен жазылған мәтінді тану мәселесі үшін пайда болды.

Нейрондық желілер оларды оқыту үшін кері таралуды қолданудың және 1989 жылы «Көпқабатты форвардтық желілер - әмбебап аппроксиматорлар» [46]

мақаласының жариялануының арқасында тағы да ғылыми қоғамдастықтың қызығушылығын тудырды. Бұл жұмыста көпқабатты нейрондық желі кез келген функцияны теориялық түрде жүзеге асыра алатыны математикалық түрде көрсетілді, бұл Минскийдің жиырма жыл бұрын көтерген негізгі мәселесін жойды.

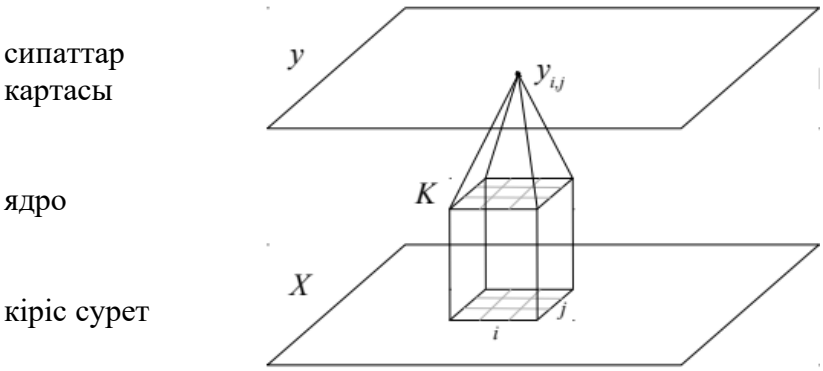
Сонымен қатар, 1992 жылы Support Vector Machines (SVM) деп аталатын жаңа алгоритм пайда болды [47]. Өте қарапайым түрде оны екі қабатты нейрондық желінің баламасын оқытудың математикалық оңтайлы тәсілі ретінде сипаттауға болады. SVM өзін нейрондық желілермен жұмыс істеу қиындығынан жоғарырақ көрсете бастайды, бұл дәл қолмен жазылған мәтінді тану мәселесіндегі алғашқы салыстырулармен көрсетілген [48].

Нейрондық желілерге деген қызығушылық «Терең желілері үшін жылдам үйрену алгоритмі» [49] мақаласы жарияланғанға дейін қайтадан төмендеді. Бұл көп деңгейлі нейрондық желілерді олардың салмақтары кездейсоқ емес, интеллектуалды түрде инициализацияланған кезде дұрыс жаттықтыруға болатынын көрсетеді және бақылаусыз деңгейлік оқыту арқылы инициализация процедурасын ұсынады. Бұл құжат нейрондық желілерге деген қызығушылықтың бетбұрысты кезеңін білдіреді, ол қазір қайта жандануда. Осы сәттен бастап бүгінгі күнге дейін нейрондық желілерді көптеген мәселелерге, сондай-ақ қабаттар саны мен олардың дәлдігіне қолдануда өсу байқалады. Мұның барлығы деректер жиынтығының көлемінің де, компьютерлердің есептеу қабілетінің де ұлғаюымен қамтамасыз етілген.

Қазіргі уақытта нейрондық желілер көптеген зерттеу салаларында толығымен үстемдік етеді және академиялық және өндірістік ортада бұл қарқынды зерттеулердің ең маңызды тақырыптарының бірі болып табылады.

**Конволюционды нейрондық желілер** (CNN) – екі өлшемді пиксельдік тор ретінде қарастырылатын кескіндер сияқты тор тәрізді құрылымы бар ақпаратты модельдеуге арналған нейрондық желінің бір түрі. Бұл желілер желі шығысын есептеу үшін стандартты матрицаны көбейтудің орнына конволюциялық математикалық операцияны пайдаланады.

Бір конволюциялық деңгей кірісті ядро ретінде белгілі $K$ параметрлерінің массивімен конвуляциялау арқылы $X$ кіріс массивін әдетте сипаттар картасы (feature maps) ретінде белгілі $y$ шығысына түрлендіреді. Бұл операция жергілікті түрде орындалады, осылайша $y_{i,j}$ шығысы $x_{i,j}$ көршілестігі негізінде ядроның конверсиялық операциясымен есептеледі. 2.10-суретте бұл әрекетті екі өлшемді кіріспен және жалғыз сипаттар картасымен шығысты жасайтын 3×3 өлшемді ядромен көрсетеді.




Сурет 2.10 – 3×3 ядросы және бір сипаттар картасы үшін CNN схемасы.

Осы желі үшін алға жүру қадамына сәйкес теңдеу келесідей:

$$y(i,j) = (K * X)(i,j) = \sum_n \quad \sum_m \quad X(i-n, j-m)K(m,n) \tag{2.3}$$

мұндағы $y$ - шығыс сиппаттар картасы, $K$ - жаттықтырылатын параметрлердің ядросы және $X$ - екі өлшемді кіріс массиві.

Конволюционды желілер кірістің жергілікті сипаттамаларын анықтауға мүмкіндік береді. Осы мақсат үшін әдетте кіріс өлшемінен әлдеқайда кіші өлшемі бар шағын ядролар пайдаланылады. Бұл кішігірім ядролар CNN-ді есептеу тұрғысынан өте тиімді және масштабты өзгертулер, аудармалар немесе айналдырулар сияқты енгізу түрлендірулеріне сенімді етеді.

Стандартты конволюциялық архитектура әдетте үш элементтен тұрады. Біріншіден, жоғарыда сипатталғандай конволюционды қабат. Екіншіден, сызықты емес белсендіру функциясы, әдетте шығыстың өзгеру ауқымын басқаратын түзетілген сызықтық бірлік (ReLU) функциясы. Үшінші кезең белсендіру функциясының шығысында біріктіру операциясын қолданудан тұрады.

Біріктіру операциясы берілген позициядағы *(i, j)* желі шығысын жақын позициялардағы шығыстың статистикалық қорытындылауымен ауыстырады. Ең кең тараған біріктіру функциясы максимум біріктіру (*max pooling*) деп аталады, ол позицияға *(i, j)* тікбұрышты маңайдағы шығыстардың максимумы ретінде есептелген мәнді тағайындайды. Біріктіру конволюционды архитектураға кіріс кескінінің шағын аудармаларына инвариантты болуына көмектеседі. Бұл қолжазба мәтіндік кескіндер үшін өте маңызды сипат, ол үшін кескіннің аудармасы тануға арналған онда жазылған мәтінді өзгертпейді. Конволюциялық архитектуралар кеңістіктік сипаттағы деректермен, әсіресе екі өлшемді кескіндерімен, сонымен қатар бір өлшемді уақыт сериялларымен немесе үш өлшемді бейнелермен жұмыс істейтін барлық модельдерде негізгі рөл атқарады. Көптеген жылдар бойы кескінді талдауға қатысты әртүрлі мәселелер үшін бірнеше конволюционды архитектуралар ұсынылды. Жалпы, бұл архитектуралар жоғарыда көрсетілген конволюцияның, белсендірудің және біріктірудің кем дегенде үш элементін қамтиды. Бұл архитектураларға толық

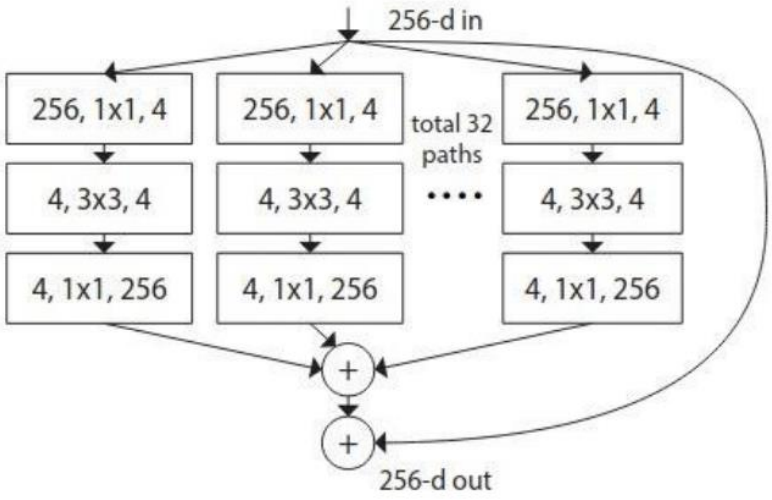
шолу осы дипломдық жұмыстың шеңберінен тыс болғандықтан танымал архитектураларға ғана тоқталатын боламыз.

Алғашқы кең танымал архитектуралар ретінде LeNet [50], VGG [51], and ResNet [52] архитектураларын айтуға болады. LeNet архитектурасы 1995 жылы қолжазба цифрларды тану үшін ұсынылған бірінші архитектура болды. Ол 5×5 ядроларды пайдаланумен және екі қабаты бар: біріншісінде 20 белгілер картасымен және екіншісінде 50 белгілер картасымен сипатталады. VGG архитектурасы 3×3 ядросы бар конволюция-конволюция-біріктіру қабатының конфигурациясын ұсынумен сипатталады. ResNet архитектурасы әрбір конволюциялық компоненттің кірісі мен шығысы арасындағы тікелей байланыстарды (қалдықтарды) ұсынумен сипатталады. Оған қоса, ол пакеттік нормалау деп аталатын регуляризация стратегиясын қолданады. Архитектураны таңдау, әсіресе қабаттар көп болған кезде, гиперпараметрлердің көбеюімен қиындай түседі.

### 2.3 Қолжазба таңбаларды классификациялау

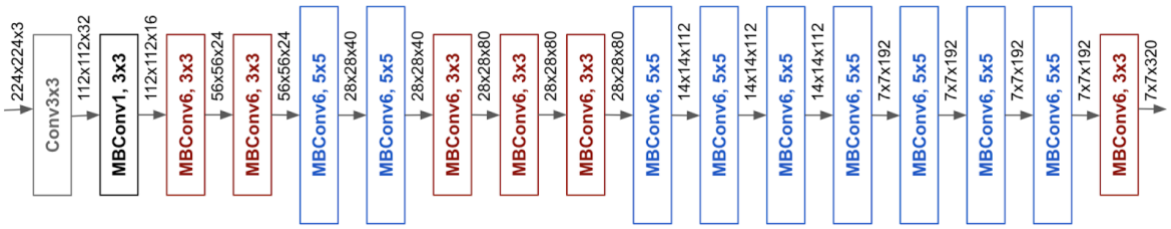
Негізгі мәселе қолжазба таңбаны оның символдық класына тағайындау болып табылады. Бұл жұмыста біз конволюционды архитектураның ImageNet [53] деректер қорында ResNeXt [54] және EfficientNet [55] сияқты SOTA (State-of-the-Art) моделдерін таңдауды шештік.

ResNeXt VGG және ResNets қабатты қайталау стратегиясын, бөлу-түрлендіру-біріктіру стратегиясын қарапайым және кеңейтілетін тәсілмен пайдаланады. Желідегі модуль түрлендірулер жинағын орындайды, олардың әрқайсысы шағын өлшемді ұя салу болып табылады, олардың шығысы қосындылау арқылы біріктіріледі. ResNeXt осы идеяны қарапайым жүзеге асыруға ұмтылады - біріктірілген түрлендірулер бірдей топологияға ие (2.11-сурет). Бұл дизайн кез келген көп түрлендірулерді арнайы модификацияларсыз кеңейтуге мүмкіндік береді.




Сурет 2.11 – ResNet архитектурасы

EfficientNet-B0 негізгі қабатынан бастап (2.12-сурет) біз оны екі қадаммен масштабтау үшін құрамдас масштабтау әдісін қолданамыз:



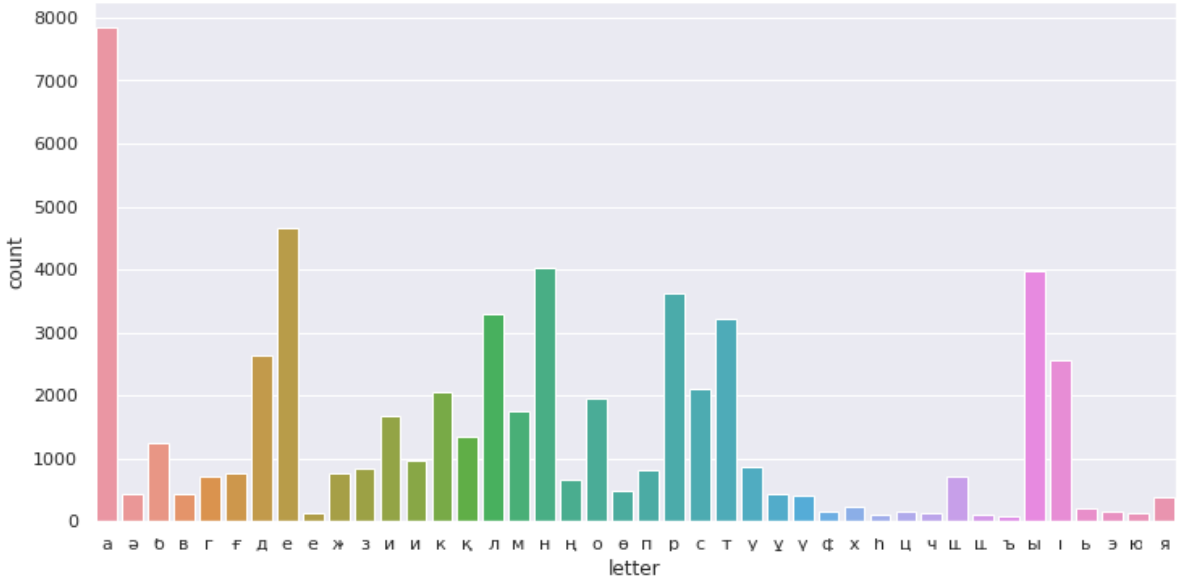

Сурет 2.12 – EfficientNet-B0 архитектурасы

2.3.1 Нейрондық желіні оқыту

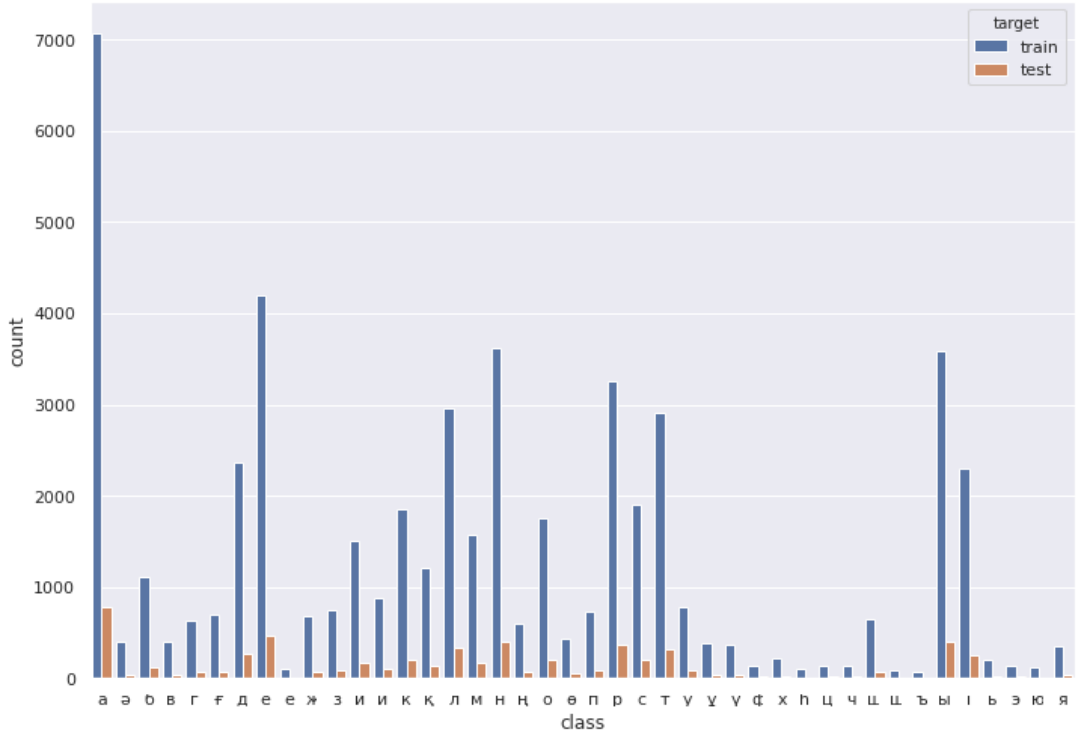
Сегменттелген таңбалар қазақ тілінің қарастырылатын 101 класына топтастырылған. Топтастыру кезінде қатты бұрмаланған кескіндер алынып тасталады және сегменттелген символдық кескіндер 32x32 өлшеміне дейін өзгертіледі. Олар туралы ақпарат csv файлына жазылады. Csv файлдарындағы әрбір жол таңба кескінін білдіреді. Бірінші баған таңбаны, екіншісі сынып белгісін, үшінші баған сурет нөмірін білдіреді. Жаттығу деректерінің жиыны үшін әр кластан таңбалық кескіндердің 80% алынды, қалған 20% сынақ (валидация) деректер жинағын құрайды. Құрылған деректер қорындағы суреттердің жалпы саны 245654. Таңбалардың кескіндерінің саны бірдей емес, өйткені кейбір әріптер сирек кездеседі, ал кейбір кескіндер таңбалардың дұрыс емес сегменттелуіне байланысты жойылады. Төменде 2.13 -суретте қазақ тілінің қолжазба таңбаларынан кездейсоқ алынған үлгілер көрсетілген.



Сурет 2.13 – Таңбалар дерекқорынан алынған кездейсоқ таңдалған кескіндер.



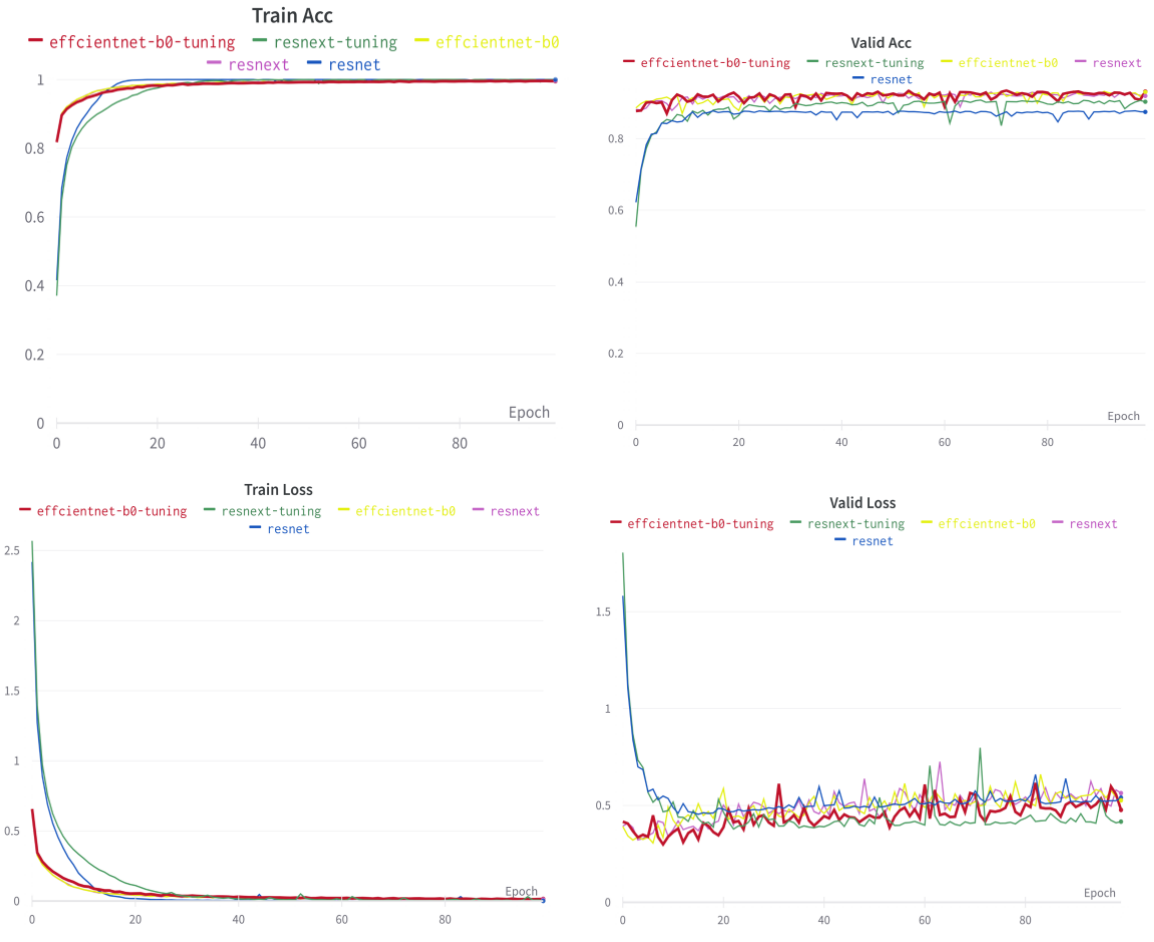

Сурет 2.14 – Датасеттегі таңбалардың үлесі



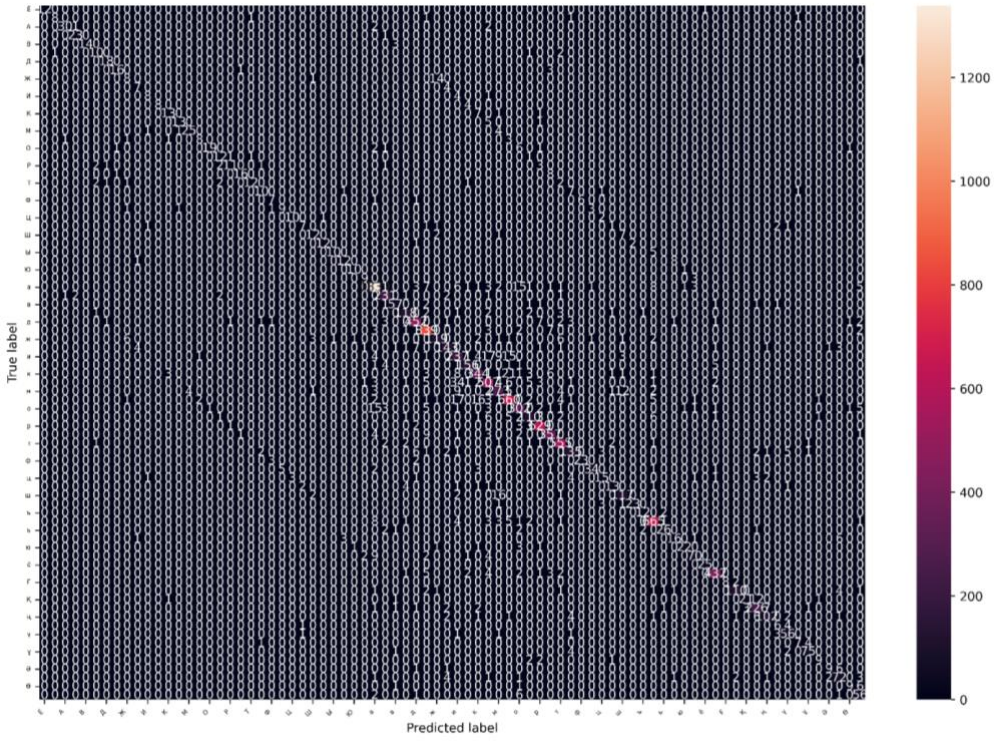

Сурет 2.15 – Таңбалардың жаттығу мен тестке бөліну саны

**Тәжірибе жүргізу ортасының талаптары**

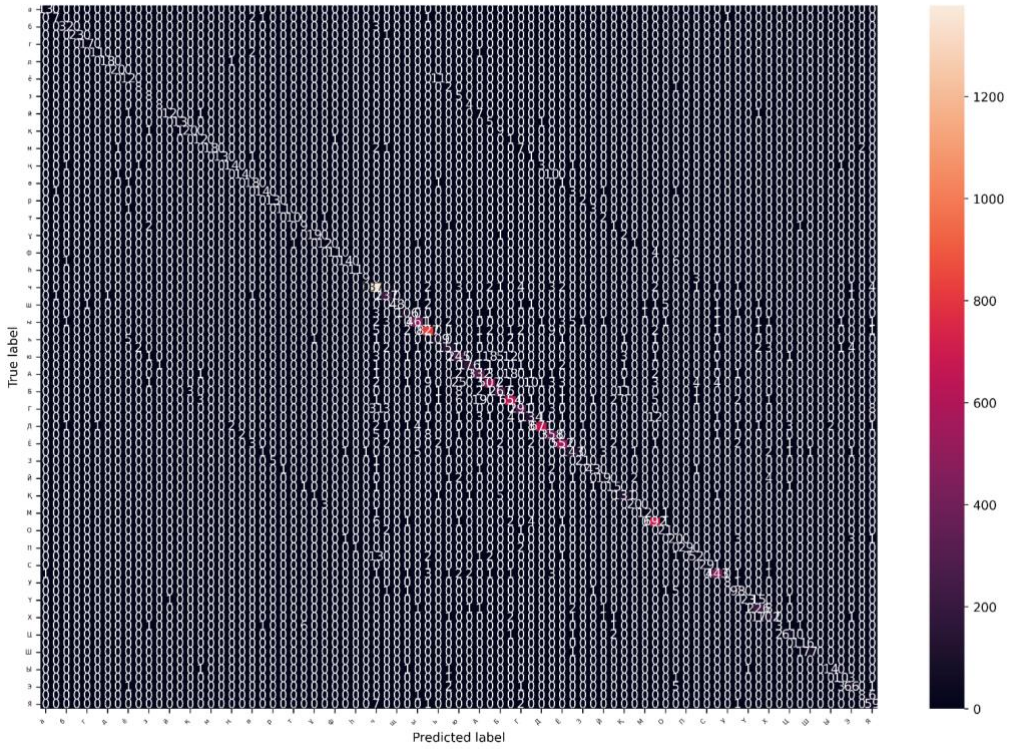
Терең нейрондық желіні оқыту үлкен есептеу ресурстарын қажет етеді. Эксперименттерді жүргізу үшін тапсырма үшін ресурстарды таңдау керек. Төмендегі кестеде ең төменгі және ұсынылған талаптар көрсетілген. Тәжірибелер 8 физикалық ядросы бар, Nvidia Geforce® RTX 3090 Ti 24 ГБ видеокартасы, 16 ГБ жедел жады бар Intel Xeon Silver процессорында жүргізілді. Қолданылған операциялық жүйе тек Ubuntu 18.04, x64 болды. Төменде 1-кестеде ең аз және ұсынылған ресурс талаптарын көре аласыз. Терең білім алу үшін ең төменгі талаптарды уақытында қолдансаңыз, қиындықтар туындауы мүмкін.




Сурет 2.16 – Оқыту дәлдігі және қателігі




Сурет 2.17 – ResNext қателер матрицасы




Сурет 2.18 – EfficientNet қателер матрицасы

2.3.2 Әріптерді классификациялау моделін бағалау

Шатасу матрицасы қолжазбаны сәйкестендіру жүйелерінің өнімділігін талдауды анықтау үшін қолданылады. 2x2 матрицадан тұратын шатастыру матрицасында жолдар сыныптарға жататын болжамды үлгі деректерін білдіреді, ал бағандарда әрбір сыныпқа жататын нақты үлгілер бар. Төмендегі 3-кестедегі шатастыру матрицасын пайдалана отырып, дәлдік, еске түсіру, дәлдік және F1-балл мөлшерлемелері алынады. Осы зерттеуде пайдаланылған мәндер төмендегі 2.3-кестеде түсіндіріледі

Кесте 2.3 – Кластар саны және бөлшектелген деректер.

| | | **Actual class** | |
|---|---|---|---|
| | | **True** | **False** |
| **Predicted class** | **Positive** | True positives – TP | False positives – FP |
| | **Negative** | False negatives – FN | True negatives – TN |

**accuracy**: бұл жүйенің бүкіл қолжазба деректер жинағынан қанша үлгіні дұрыс болжағанының көрсеткіші. Дәлдік келесі теңдеу арқылы алынады:

$$ACC = \frac{TP+TN}{TP+FP+TN+FN} \quad (2.4)$$

**recall**: бұл нақты қолжазба деректерінің жалпы санына дұрыс болжанған нақты қолжазба үлгілерінің қатынасы. Оны кейде шынайы оң көрсеткіш (TPR) деп те атайды. Ол келесі теңдеу арқылы алынады:

$$TPR = \frac{TP}{TP+FN} \quad (2.5)$$

**precision:** бұл тексеру жүйесі арқылы ақиқат ретінде көрсетілген қолжазба үлгілерінің санына дейін дұрыс болжанған шынайы қолжазба үлгілерінің қатынасы. Ол келесі теңдеу арқылы алынады:

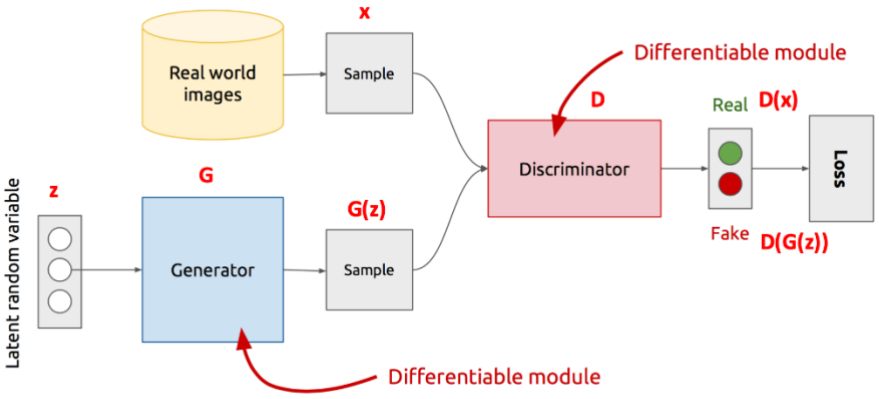
$$TPV = \frac{TP}{TP+FP} \quad (2.6)$$

**F1-score**: Бұл precision мен recall түсіру мәндерінің гармоникалық орташа мәні. F1 мәні келесі теңдеу арқылы алынады:

$$F1\ Score = \frac{2*TPV*TPR}{TPV+TPR} \quad (2.7)$$

### 2.4 Деректерді жасанды көбейтудің GAN әдісі

Қолжазба мәтінде әртүрлі адамдар жазған бір сөздің өзі жазу стилі мен жазылған контекстке байланысты өзгеріп отырады. Бұл өз кезегінде тану моделінің дәлдігіне тікелей әсер етеді. Зерттеу барысында пайдаланылған деректер жиынтығында жинақталған қолжазба сөз үлгілері жоғарыда аталған вариативтілікті толық қамти алмайды. Мұндай олқылықты толық аннотталған қолжазбалар корпусының жеткіліксіздігімен, сондай-ақ оларды жинау және алдын ала өңдеу үдерісінің күрделілігімен байланыстыруға болады. Аталған мәселені шешу мақсатында бұл бөлімде қолжазба мәтіндердің визуалды үлгілерін генерациялау, қолмен жасалатын аннотацияларға деген тәуелділікті азайту, сондай-ақ стильдік және лексикалық алуан түрлілік тұрғысынан оқыту деректерінің кеңеюін қамтамасыз ету жолдары қарастырылады. Осы бағытта Генеративті қарсылас желілер (GAN) тиімді шешімдердің бірі ретінде таңдалып отыр. GAN екі нейрондық желіні (генератор мен дискриминатор) салыстыру арқылы нақты деректерге ұқсас синтетикалық жолмен жаңа деректерді жасайды (сурет 2.3.1).




Сурет 2.19 – GAN архитектурасы

Генератор жаңа модельдер жасау үшін деректердің нақты таралуын анықтауға тырысады [56]. Бұл тәсіл соңғы жылдары компьютерлік көру саласында кеңінен таралып, зерттеушілер назарын аударуда. Себебі GAN ықтималдық тығыздығы функциясын нақты түрде анықтамай-ақ, бастапқы үлестірімге ұқсас үлкен көлемдегі синтетикалық деректерді генерациялауға жағдай жасайды. Генератор шынайы деректерге барынша ұқсас деректерді құрастыруға бағытталады, ал дискриминатор өз кезегінде генератордың жасаған жалған деректерін нақты деректерден ажыратуға үйренеді. Бұл бәсекелестік оқыту үдерісі екі желінің де біртіндеп жақсаруына алып келеді. Генеративті қарсылас желілер генеративті модельдер тобына кіреді. Мұндай модельдер оқыту жиынтығы негізінде деректердің ықтималдық таралуын үйреніп, сол үлестірімнен жаңа синтетикалық үлгілерді генерациялауға қабілетті. GAN архитектурасы нақты деректерге ұқсас жоғары сапалы синтетикалық үлгілерді генерациялау арқылы тану жүйелерін жетілдіруге мүмкіндік береді.

### 2.4.1 GAN модельдері үшін қолданылатын шығын функциялары

Шығын функциялары үйретілетін модельдің болжау нәтижелері мен нақты мәліметтер арасындағы айырмашылықты сипаттайды. Егер болжанған нәтиже мен нақты нәтиже арасындағы ауытқу тым үлкен болса, онда шығын функциясының мәні де жоғары болады. Уақыт өте келе, белгілі бір оңтайландыру функциясының көмегімен шығын функциясы бұл қателікті азайтып үйренеді.

**Қаріпті классификатордың шығын функциясы** - синтетикалық кескіндерде алдын ала үйретілген және бүкіл архитектураны оқыту барысында қатырып қоятын қаріп классификатор. Бұл классификатор генераторға қаріптердің мәнін тереңірек түсіну үшін кері байланыс (градиент) береді. Аталған шығын функциясы генерацияланған кескін мен мақсатты кескін арасындағы қаріп айырмашылығын азайтуға бағытталған.

Дискриминативті шығын функциясы. Дискриминаторды үйрету кезінде ол нақты деректер мен генератордан шыққан жалған деректерді жіктейді. Ол нақты мәліметті жалған деп немесе генератор жасаған жалған мәліметті нақты деп қате

жіктегені үшін өзін жазалайды. Бұл шығын функциясы келесі математикалық функцияны барынша арттыру арқылы жүзеге асады.

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \quad [log\ log\ D\ (x^{(i)}) + log\ log\ (1 - D(G(z^{(i)})))] \tag{2.8}$$

мұндағы log(D(x)) - генератордың нақты кескінді дұрыс жіктеу ықтималдығын білдіреді. Ал log(1 – D(G(z))) функциясын барынша арттыру - генератордан шыққан жалған кескінді дискриминатордың дұрыс анықтауына көмектеседі. Мұндағы ∇ белгісі стохастикалық градиентті білдіреді, ал θ_d - көпқабатты перцептронның гиперпараметрі болып табылады. Бұл гиперпараметр дифференциалданатын G(z; θ_d) функциясын сипаттайды, ол кіріс ретінде берілген шу z-і деректер кеңістігіне түрлендіреді.

**Генератордың шығын функциясы.** Генераторды оқыту кезінде оған кездейсоқ шу беріледі және ол осы шудың негізінде қандай да бір нәтиже (кескін) генерациялайды. Бұл нәтиже кейін дискриминатор арқылы өткізіліп, "нақты" немесе "жалған" ретінде классификацияланады - дискриминатордың бұл айырмашылықты қаншалықты дәл анықтай алатынына байланысты. Генератордың шығын мәні дискриминатордың бағалауы негізінде есептеледі: егер генератор дискриминаторды сәтті алдай алса, ол марапатталады, ал егер алдай алмаса - жазаланады. Генераторды үйрету келесі теңдеуге келіп тіреледі:

log(D(x)) - генератордың нақты кескінді дұрыс жіктеу ықтималдығын білдіреді, ал log(1 – D(G(z))) функциясын барынша арттыру генератор жасаған жалған кескінді дұрыс белгілеп, дискриминаторды жаңылыстыруға көмектеседі.

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \quad (D(G(z^{(i)})) \tag{2.9}$$

2.4.2 Генеративті қарсылас желілер (GAN) алгоритмдерін бағалау метрикалары

GAN алгоритмдерінің тиімділігі мен сапасы әртүрлі бағалау метрикалары арқылы өлшенеді. Олардың қатарына Inception Score (IS), Frechet Inception Distance (FID), Structural Similarity Index (SSIM) және Fréchet Distance (FD) жатады [57, 58]. Inception Score (IS) - генерацияланған кескіндердің сапасын бағалайтын метрика, ал FID және SSIM - нақты және генерацияланған кескіндер арасындағы ұқсастықты өлшейді. Fréchet Distance (FD) - нақты және жасанды кескіндердің таралу тығыздықтарының (distribution) арасындағы ұқсастықты бағалайды. Сонымен қатар, GAN алгоритмдерін бағалауда пайдаланушыларға негізделген зерттеулер, Human Visual System (HVS) немесе классификация дәлдігі сияқты метрикалар да қолданылады.

Inception Score (IS) - генеративті модельдердің, атап айтқанда, генеративті қарсылас желілер (GAN) арқылы алынған синтетикалық кескіндердің сапасын өлшейтін метрика. Бұл көрсеткіш конволюциялық нейрондық желіге негізделген және генерацияланған кескіндер үшін болжанған белгілердің таралуы мен нақты белгілердің таралуы арасындағы Кульбак-Лейблер дивергенциясының (KL-

divergence) орташа мәні арқылы есептеледі [59]. IS сандық түрде келесі теңдеу арқылы анықталады:

$$I = exp(E_x D_{KL}(p(y|x)||p(y))) \quad (2.10)$$

мұндағы x - генерацияланған бір моделді білдіреді, ал y - модель болжаған белгі.

Frechet Inception Distance (FID) - екі кескіндер жиыны арасындағы ұқсастықты өлшеуге арналған метрика. Бұл көрсеткіш екі кескіндер жиынының ерекшелік (feature) репрезентацияларының статистикалық сипаттамаларын салыстыру арқылы есептеледі. FID мәні неғұрлым төмен болса, екі кескіндер жиыны бір-біріне соғұрлым ұқсас деп есептеледі. Frechet қашықтығы екі көп айнымалыдан (multivariate) тұратын нормалды таралудың арасындағы қашықтықты есептеу үшін қолданылады. Ал бір айнымалы (univariate) нормалды таралу жағдайында Frechet қашықтығы келесідей түрде өрнектеледі:

$$d(X,Y) = (\mu_X - \mu_Y)^2 + (\sigma_X - \sigma_Y)^2 \quad (2.11)$$

мұндағы μ және σ - нормалды таралудың орта мәні мен стандартты ауытқуы, ал X және Y - екі нормалды (Гаусс) таралуды білдіреді. Жалпы жағдайда FID метрикасы - екі көпөлшемді Гаусс таралуы арасындағы квадратталған Вассерштейн (Wasserstein) қашықтығы ретінде анықталады. Атап айтқанда, N(μ, Σ) - генеративті қарсылас желі (GAN) арқылы жасалған кескіндердің нейрондық желідегі ерекшелік (feature) репрезентацияларының таралуы, ал N(μ_w, Σ_w) - GAN-ды үйрету үшін пайдаланылған нақты кескіндердің дәл сол нейрондық желі ерекшеліктерінің таралуы:

$$FID = \|\mu_X - \mu_Y\|^2 - T_r\left(\sum_X + \sum_Y - 2\sqrt{\sum_X \sum_Y}\right) \quad (2.12)$$

мұндағы X және Y — шынайы және генерацияланған ендірмелер (embeddings), олар екі көпөлшемді нормалды таралымдар ретінде қарастырылады. $\mu_x$ және $\mu_\gamma$ - сәйкесінше X және Y векторларының орташа мәндері (модульдері). Tr - матрицаның трассасы (бас диагональ элементтерінің қосындысы), ал $\sum_x$ және $\sum_\gamma$ - X және Y векторларының ковариациялық матрицалары.

2.4.3 GAN оқыту тәжірибелері

Деректер жиынын оқу (train) және тестілеу (test) жиындарына бөлу стратегиямыз әр беттен жолдарды кездейсоқ таңдау әдісіне негізделді. Деректердің 80%-ы оқу үшін, ал 20%-ы тестілеу үшін бөлінді. Мұндай рандомизация әдісі оқу деректерінің әртүрлілігін арттырып, модель құру үшін маңызды болып табылатын деректерді жақсы сипаттауға мүмкіндік береді. Жалпы оқу жиыны 3425 жолдан тұрады, оның құрамында 45 211 сөз және 234

576 таңба (символ) бар. Тестілеу жиыны 685 жолдан тұрады, онда 12 312 сөз және 77 713 таңба қамтылған. Эксперимент келесі сипаттамалары бар дербес компьютерде жүргізілді: Intel Core i7, 12.90GHz × 16, NVIDIA RTX3060 GPU (12GB) және 32GB жедел жады (RAM). Операциялық жүйе ретінде Ubuntu 20.04.5 LTS, ал терең оқытуға арналған фреймворк ретінде PyTorch 1.9.0 пайдаланылды.

*Эксперимент 1.* Осы бөлімде қолжазба мәтіндерге арналған бірнеше GAN-модельдерге талдау жүргізіледі. Атап айтқанда, алғашқы эксперименттік зерттеу ретінде TextStyleBrush моделі таңдалды [60]. Аталған модель 25 қабаттан тұрады. Модельдің негізгі компоненттері ретінде мазмұн кодтаушысы (content encoder) және стиль кодтаушысы (style encoder) қолданылады, олардың екеуі де ResNet34 архитектурасына негізделген. Модельдің маңызды ерекшелігі – мәтіндік деректер пиксельдік кескін түрінде ұсынылады, яғни әріптер сурет форматында стандартты қаріппен ақ фонда бейнеленген. Стильді көрсету торы (style rendering network) – стиль кодтаушысының шығыс векторын бірнеше жеке векторларға түрлендіріп, оларды генератордың түрлі қабаттарына AdaIN (Adaptive Instance Normalization) қабаттарының параметрлері ретінде енгізеді [61].

$$AdaIN(z, \alpha, \beta) = \alpha\left(\frac{z-\mu(z)}{\sigma(z)}\right) + \beta \qquad (2.13)$$

мұндағы z ∈ F (каллиграфиялық стиль атрибуттарының және мәтіндік мазмұн туралы ақпараттың таңбалар бойынша бөлінуінің тіркесімі), μ және σ - арналардың орташа мәні және стандартты ауытқулар. Жаһандық мазмұн туралы ақпарат AdaIN қабаттарын құру процесінде төрт рет (p = 4) енгізіледі. Олардың α және β параметрлері fc-ді төрт жұпқа бөлу арқылы алынады.

$$\bar{x} = H(t, X_i) = G(C(t), S(X_i)) = G(g_1(\tilde{t}), g_2(\tilde{t}), S(X_i)), \qquad (2.14)$$

мұндағы $\tilde{t} = [e(c); \forall c \in t]$, t - жолының кодталуы.

Архитектуралық ерекшеліктеріне байланысты TextStyleBrush тек бұрыннан бар стильдер мен жазу үлгілері бойынша генерация жасай алады және мүлде жаңа стильдерді немесе бар стильдердің "орташа нұсқасын" тудыруға қабілетсіз. Бұл модель кіріс ретінде тек бір кескінді (стиль үлгісі ретінде) және шығысы ретінде сол стильде, бірақ басқа мәтінмен бір ғана кескінді қолдана алады. Генератордың кірісіне мәтін қарапайым қаріппен ақ фонда басылған кескін түрінде беріледі және кейіннен ұлғайтылады (upscale). Сонымен қатар, стандартты баспа қаріптерін ажыратын классификатор да шығын функцияларының бірі ретінде әрекет етеді. Жалпы алғанда, бұл архитектура қолжазбаларға қарағанда баспа мәтіндері бар кескіндерді генерациялау үшін неғұрлым тиімді болып табылады.

*Эксперимент 2.* Бұл эксперимент аясында зерттеу жұмысы GanWriting [62] архитектурасына негізделген генеративті нейрондық желіні оқытуға бағытталды. Бұл архитектура TextStyleBrush моделінен бірқатар маңызды айырмашылықтарымен ерекшеленеді. Мазмұн кодтаушысы екі бөлікке - $g_1$ және $g_2$ атты екі тармаққа (head) бөлінеді. $g_1$ тармағының шығыс векторы стиль кодтаушысының (style encoder) нәтижелерімен біріктіріліп, алынған біріктірілген тензор генератордың кірісіне беріледі. Бұл генератор тензорды жоғары өлшемдегі (high-resolution) кескінге түрлендіреді, яғни масштабын ұлғайтады (upscaling). Ал $g_2$ тармағы арқылы алынған мазмұндық векторлар генератор архитектурасының төрт деңгейіне ендіріліп, AdaIN (Adaptive Instance Normalization) қабаттары арқылы өңделеді. Бұл әдістеме стильдік ақпарат пен мазмұндық ақпаратты генератордың әртүрлі деңгейлерінде нақты түрде біріктіруге мүмкіндік береді. Осы жерде маңызды бір айырмашылықты атап өту қажет: TextStyleBrush архитектурасында генераторға стильдік ақпарат енгізілсе, ал GanWriting моделінде генераторға мазмұндық ақпарат жіберіледі. Бұл айырмашылық архитектуралардың мәліметтермен жұмыс істеу тәсілін өзгертеді және соңғы генерацияланған мәтіннің стиліне әсер етеді. Жалпы алғанда, GanWriting негізіндегі бұл тәсіл кіріс мәтінді неғұрлым құрылымдық түрде өңдеуге және стиль мен мазмұнды нақты түрде бөлек басқаруға мүмкіндік береді. Мұндай тәсіл генерацияланған қолжазба мәтіндерінің сапасын арттыруға септігін тигізеді.

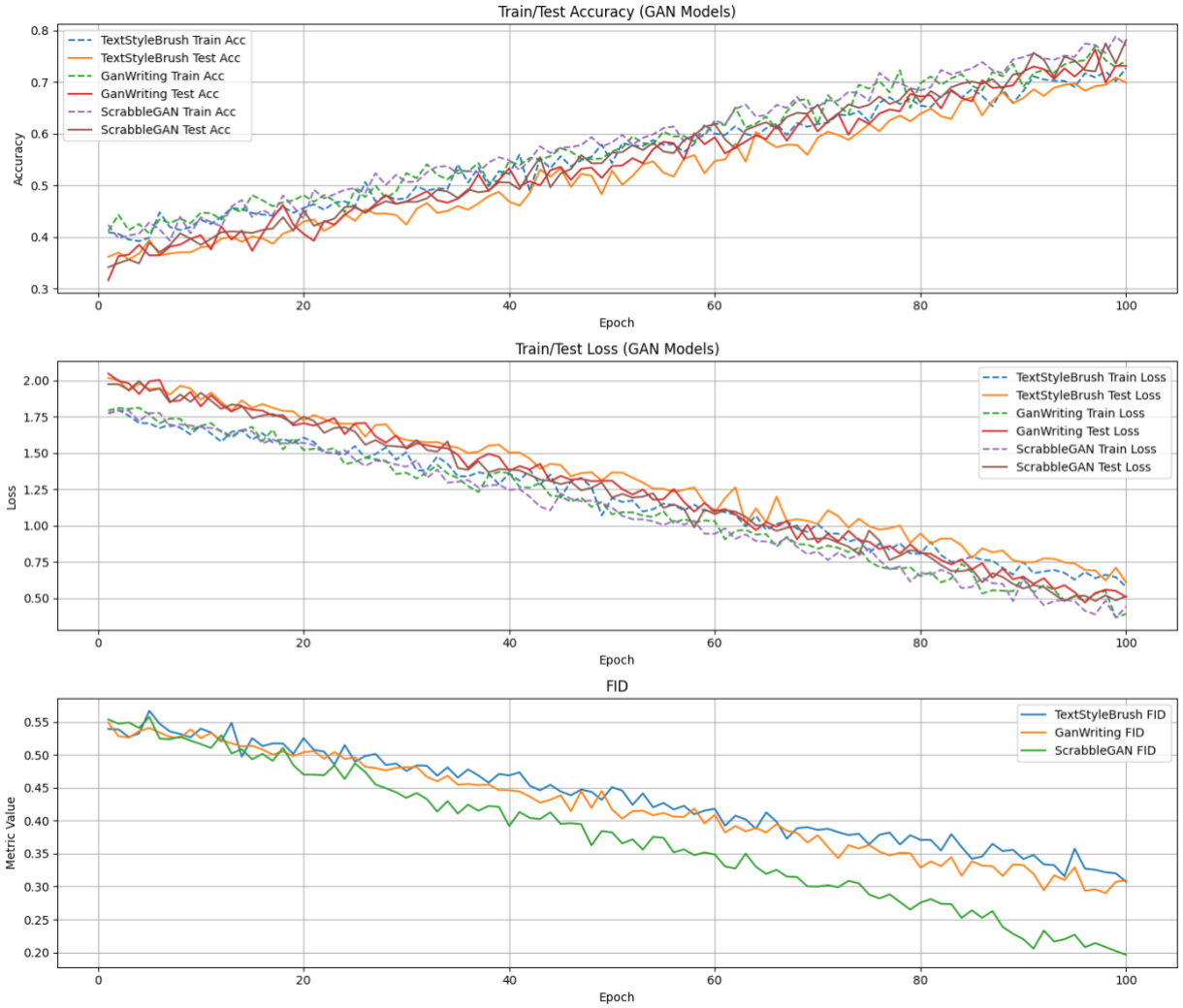
*Эксперимент 3.* Бұл экспериментте ScrabbleGAN [63] архитектурасы қолданылды. Бұл модель құрылымы жағынан алдыңғы бөлімдерде қарастырылған модельдерге қарағанда едәуір жеңіл болып табылады. Архитектурада тек генератор, дискриминатор және OCR-модуль (оптикалық таңу жүйесі) қарастырылған.

Архитектураның жұмыс істеу принципі келесідей: мәтін one-hot форматында ұсынылады және ол кездейсоқ шу (noise) векторымен көбейтіледі. Осыдан кейін нәтижесі толық қосылған (fully-connected) сызықтық қабаттардан өтеді және BigGAN [64] моделіне негізделген генератор арқылы масштабталып, кескінге айналады. Мұндағы шу векторы жазу стилін (қолмен жазылған түрі, әріптердің қалыңдығы, көлбеулігі және т.б.) анықтауға жауап береді.

Генерацияланған кескін екі бағытта қолданылады:

1. Дискриминаторға енгізіледі – бұл модельдің жалпы сапасын арттыруға ықпал етеді;
2. OCR-модульге жіберіледі – мәтінді адам оқи алатындай етіп тануға мүмкіндік береді.

Барлық модельдер үшін біз Adam оптимизаторын қолданамыз $\beta1 = 0$ және $\beta2 = 0{,}8$ оқыту үшін. Үнсіз келісім бойынша, дискриминатор үшін оқу жылдамдығы 0,0004, ал генератор үшін оқу жылдамдығы 0,0001.




Сурет 2.20 – GAN модельдерін оқыту нәтижелері

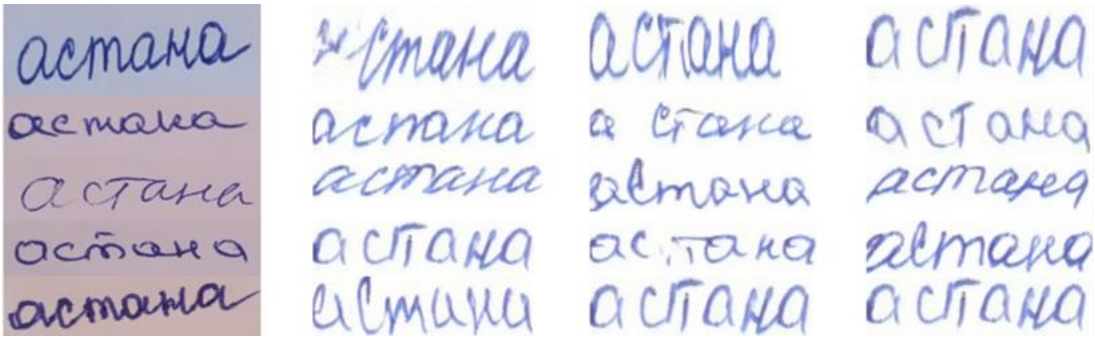
2.4-кестеде көрсетілгендей, SrabbleGAN моделі барлық модельдер арасында FID метрикасы бойынша жақсы нәтижелер көрсетті.

Кесте 2.4 – Әр түрлі жоғалту функцияларының GAN модельдерінің нәтижелеріне әсерін салыстыру.

| **Model** | **FID** |
|---|---|
| TextStyleBrush | 32.11 |
| GanWriting | 29.23 |
| ScrabbleGAN | **20.14** |

TextStyleBrush, GanWriting және ScrabbleGAN модельдері арқылы генерацияланған кескіндерді талдау барысында әрқайсысының артықшылықтары мен шектеулері анықталды. TextStyleBrush моделі жазу стилінің әртүрлілігін шектеулі түрде ғана қамтиды, дегенмен дәл осы қасиеті OCR моделінің тұрақты әрі сенімді жұмыс істеуіне оң әсерін тигізеді. Ал

GanWriting моделінде графикалық тұрғыдан ұқсас әріптерді, мысалы, "м", "ш", "щ" сияқты символдарды бір-бірінен ажырату қиындық тудырады, бұл модельдің әлсіз тұсы ретінде танылды. ScrabbleGAN моделі жазу стиліндегі әртүрлілікті анағұрлым жақсы сақтай отырып, шынайы қолжазба үлгілеріне жақын нәтижелер көрсетті. Алайда, бұл модельде бас әріптерді генерациялау кезінде кейбір кемшіліктер байқалды. Мұндай шектеулердің негізгі себебі – оқыту деректер жиынтығында бас әріптердің аз кездесуі. Жоғарыда келтірілген байқауларға сүйене отырып, келесі кезеңде жүргізілген эксперименттерде кескіндер генерациялау үшін тек ScrabbleGAN моделі пайдаланылды (Сурет 2.21).



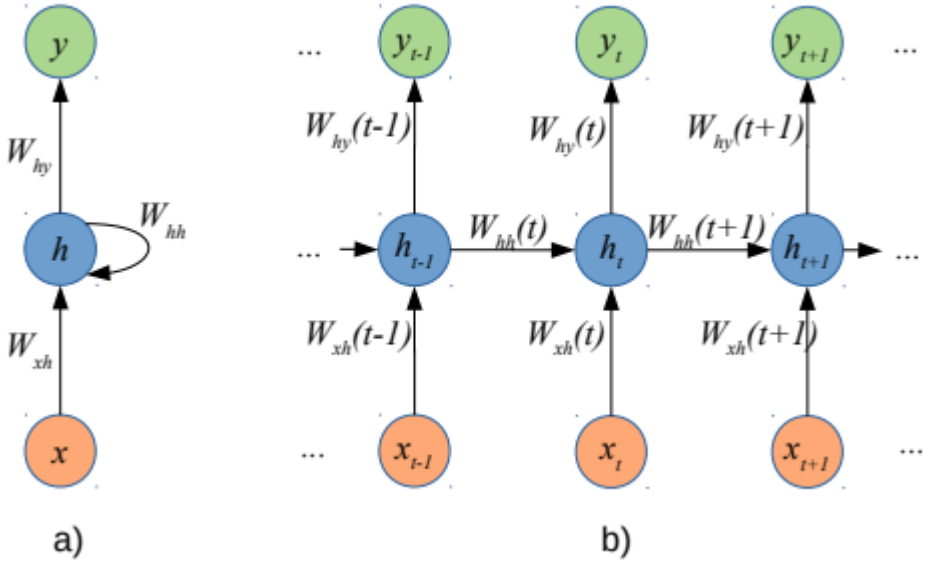
a) b) c) d)

Сурет 2.21 – GAN модельдерінің визуалды салыстыру нәтижелері: (a) нақты деректер; (b) Textstyle стиліндегі щетка; (c) GANWriting; (d) Scrabble GAN.

### 2.5 Сөзді тану алгоритмдері

#### Рекурентті нейрондық желілер (RNN)

Рекуррентті нейрондық желілер (RNN) - бұл тізбекті мәліметтерді ($x_1$, ..., $x_n$) ретімен өңдеуге арналған нейрондық желілердің бір түрі [65]. Мұндай желілер оқу процесінің мақсатына сәйкес болған жағдайда, деректердегі уақыттық немесе дәйектілік байланыстарды тиімді түрде модельдей алады. Бұл желілердің жұмыс істеу принципі - кіріс деректер тізбегі өңделген кезде желінің қайталанатын қабатының әртүрлі түйіндері (нейрондары) арасында параметрлерді ортақ пайдалану идеясын қолдануға негізделеді. Осы құрылымның арқасында, рекуррентті желідегі әрбір шығыс элементі $h_t$ алдыңғы уақыт сәтіндегі шығыс $h_{t-1}$ элементінің функциясы ретінде есептеледі. Жалпы алғанда, бұл байланыс негізінде $x_1, \ldots, x_t$ кіріс тізбегінің функциясы болады. 2.22-суретте негізгі рекуррентті нейрондық желі (RNN) құрылымының сызбасы берілген.




Сурет 2.22 – RNN схемасы: а) ықшам түрі және б) уақыт бойынша ашылатын түрі

2.22-суретке сәйкес негізгі рекуррентті нейрондық желінің (RNN) теңдеулері келесідей түрде анықталады:

$$h_t = tanh\,(W_{hh}h_{t-1} + W_{xh}x_t)$$
$$y_t = W_{hy}h_t \qquad (2.15)$$

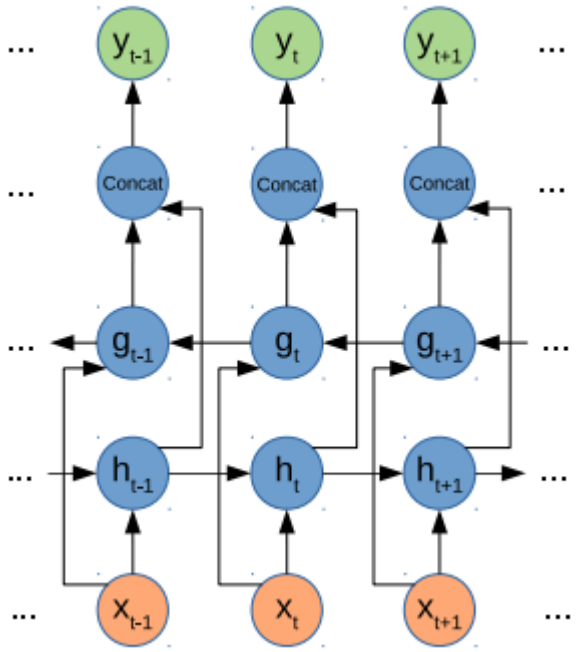
Бұл теңдеулер рекуррентті байланыстарда қайталану механизмін және гиперболалық тангенс белсендіру функциясын қолданудың негізгі стратегиясын сипаттайды. Бұл модель - рекуррентті нейрондық желілердің ең кең таралған түрлерінің бірі болып табылады. Дегенмен, рекуррентті желілердің әртүрлі вариациялары да бар, олардың кейбіреулері келесі бөлімдерде қарастырылады.

RNN моделін оқыту үшін кері таралу алгоритмінің өзгертілген нұсқасы қолданылады, ол уақыт бойынша кері таралу (Backpropagation Through Time - BPTT) деп аталады. Бұл әдіс - уақыт бойынша ашылған RNN құрылымына бейімделген кері таралуды оңтайландыру тәсілі. BPTT әдісі сандық тұрақтылықты сақтау мақсатында градиенттерді есептеуде қысқарту (truncation) стратегиясын қолданады, сондай-ақ тиімді есептеулер жүргізу үшін аралық мәндер сақталады.

Бұрын сипатталған негізгі рекуррентті нейрондық желі себептік құрылымға ие, өйткені t уақыт сәтінде тек өткен $x_1, \ldots, x_{t-1}$ және ағымдағы $x_t$ мәндеріндегі ақпарат қана пайдаланылады. Алайда, кейбір жағдайларда, мысалы, HTR (Handwritten Text Recognition) жүйелеріндегі офлайн тану міндетінде, нақты бір таңбаны дұрыс анықтау үшін алдыңғы және кейінгі контекстті бір мезгілде ескеру қажет болуы мүмкін. Яғни, мәтіннің ортасында орналасқан символды тану кезінде оның алдында және соңында орналасқан символдар туралы ақпарат шешуші рөл атқаруы мүмкін.

Мұндай толық мәтіндік контекстті модельге енгізу үшін қос бағытты рекуррентті нейрондық желілер (Bidirectional Recurrent Neural Networks - BRNN) қолданылады. Бұл архитектура HTR мәселелерінде ұзақ уақыт бойы табысты қолданылып келеді. BRNN құрылымында мәліметтер тізбегі бойымен t уақытының өсу бағытына қарай бір RNN жұмыс істейді, ал екінші RNN t уақытының кему бағытына қарай - яғни тізбектің соңынан басына қарай қозғалады.

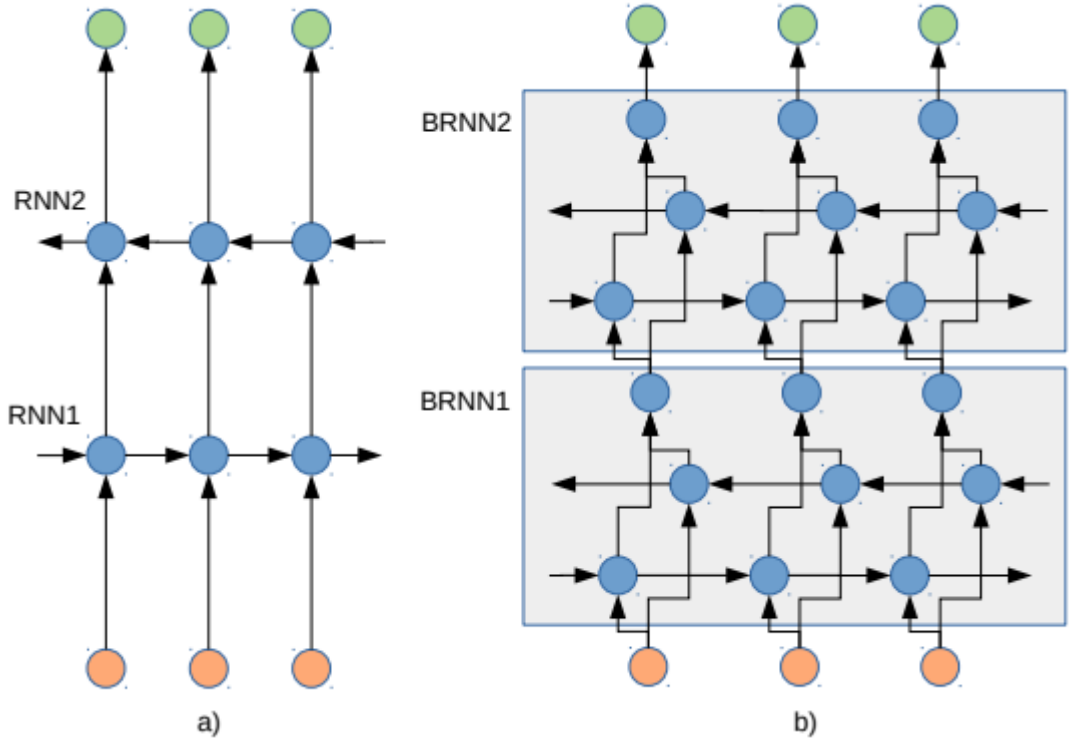
4.6-суретте BRNN архитектурасының типтік сызбасы көрсетілген. Мұндағы *ht* - уақыт бойынша алға бағытталған RNN-нің күйі, ал *gt* - уақыт бойынша кері бағыттағы RNN-нің күйі. Екі бағыттағы RNN-нің шығыс мәндері біріктіру функциясы арқылы әрбір уақыт қадамында біріктіріледі. Ең жиі қолданылатын біріктіру тәсілі - қосатын біріктіру, дегенмен, қосынды немесе элемент бойынша орташа мән әдістері де кейбір жағдайларда қолданылады.




Сурет 2.23 – Қосбағытты рекурентті нейрон сызбасы

Бұл идеяны табиғи түрде екі немесе одан да көп өлшемге кеңейтуге болады. Мұндай кеңейту көпөлшемді рекуррентті нейрондық желі (MDRNN) архитектурасын құрайды және, жоғарыда айтылғандай, бұл тәсіл HTR мәселелерінде жиі қолданылады. MDRNN архитектурасы кескін тәрізді құрылымдық деректерге қатысты контекстті әртүрлі бағыттар бойынша қатар өңдеуге мүмкіндік береді. Көп деңгейлі рекуррентті архитектураларды құру үшін бірнеше қайталанатын қабаттарды қосу - жиі кездесетін тәсіл. Бұл жағдайда әрбір қабаттың шығыс тізбегі келесі қабаттың кірісі ретінде пайдаланылады. Мұндай тәсіл MLP желісіндегі тығыз (dense) қабаттар арасындағы байланысқа ұқсас. 4.7-суреттің сол жағында мұндай жинақталған RNN архитектурасының сызбасы көрсетілген. Бұл құрылымда бірнеше қайталанатын қабаттар тізбекті түрде байланыстырылған.

Сонымен қатар, екі бағыттағы RNN құрылымдарын бір-бірімен тізбектей біріктіру де кең таралған тәсіл болып табылады. Бұл жағдайда әрбір қабат екі бағытта ақпарат өңдейді және бұл тәсіл 4.7-суреттің оң жағында көрсетілген екібағытты жинақталған RNN архитектурасына сәйкес келеді.




Сурет 2.24 – Көпқабатты RNN сызбалары: a) Стектелген RNN. b) Екібағытты стекке салынған RNN

Зерттеулер [66] жұмыста көрсеткендей, рекуррентті желілердің бұл түрлерінде жоғалып кететін градиент (vanishing gradient) мәселесі туындайды. Яғни, уақыт бойынша ұзақ тәуелділіктерді модельдеу барысында, көптеген қабаттар арқылы таратылатын градиенттер шамадан тыс кішірейіп, нейрондық желінің үйрену қабілетін төмендетеді. Бұл мәселе ұзақ қашықтықтағы (ұзақ уақыт интервалындағы) байланыстарды үйренуді қиындатады, тіпті мүмкін емес етеді. Аталған мәселені шешудің ең кең таралған жолы - қақпалы рекуррентті нейрондық желілерді қолдану. Мұндай желілер уақыт бойындағы ақпарат ағынын тиімді басқаратын қақпа механизмдері арқылы градиенттердің жоғалуын болдырмауға мүмкіндік береді.

Қазіргі кезде ең жиі қолданылатын қақпалы архитектуралар - ұзақ қысқа мерзімді жады (LSTM) желілері және жабық қайталанатын бірліктер (GRU) желілері. Бұл архитектуралар ұзақ мерзімді тәуелділіктерді сақтау қабілетіне ие және классикалық RNN-дермен салыстырғанда оқыту кезінде анағұрлым тұрақты нәтижелер көрсетеді.

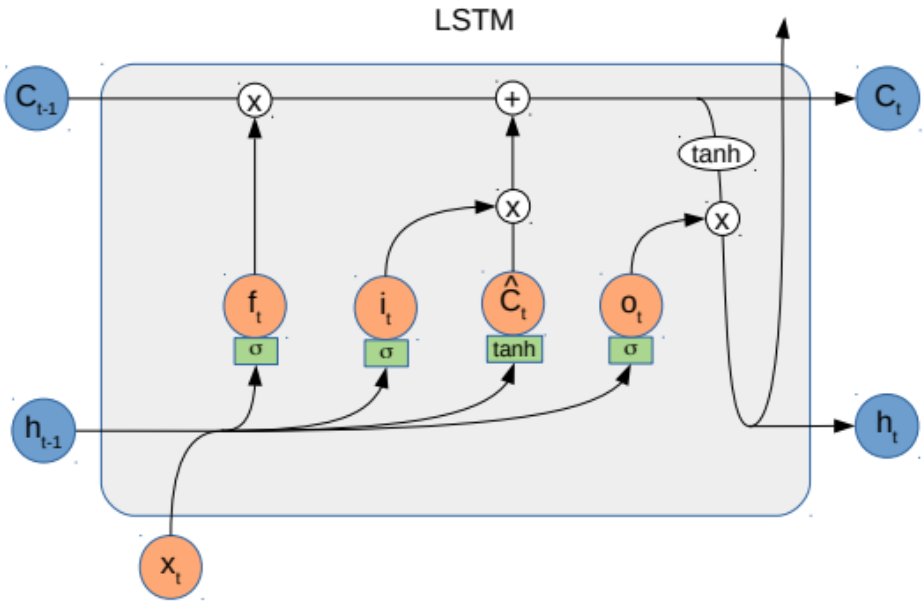
**Ұзақ қысқа мерзімді жады (LSTM)**

Ұзақ қысқа мерзімді жады (LSTM) желілері - ұзақ мерзімді тәуелділіктерді тиімді үйренуге қабілетті рекуррентті нейрондық желілердің (RNN) ерекше түрі

болып табылады. Бұл желі құрылымы ақпаратты ұзақ уақыт бойы сақтау және қажет болған кезде жаңарту немесе өшіру мүмкіндігін қамтамасыз ету үшін арнайы жобаланған.

LSTM желілері алдыңғы уақыт сәтіндегі ақпаратты келесі қадамға екі түрлі вектор арқылы береді: *h* - шығыс векторы, *c* - күй векторы.

LSTM архитектурасы үш негізгі қақпа (gates) — ұмыту қақпасы (*forget gate*), енгізу қақпасы (*input gate*) және шығыс қақпасы (*output gate*) арқылы жұмыс істейді. Бұл қақпалар нейрондық желіге қандай ақпаратты сақтау, жаңарту немесе жою қажет екенін анықтауға мүмкіндік береді. Осы механизмдердің арқасында LSTM желілері ұзақ мерзімді тәуелділіктерді сақтау мен тасымалдауды тиімді жүзеге асырады. 2.25-суретте LSTM ұяшығының құрылымдық сызбасы бейнеленген.




Сурет 2.25 – LSTM ұяшығының схемасы

Кіріс қақпасы деп аталатын *i* қақпасы күй векторында қандай мәндер жаңартылатынын біледі. Ұмыту қақпасы деп аталатын *f* қақпасы лақтыруға болатын алдыңғы күйдегі ақпаратты үйренеді. Осы екі қабаттың шығысымен желі жаңа үміткер күй мәндерінің *c* векторын жасайды. Соңында шығыс қақпасы деп аталатын *o* қақпасы *h* шығысына қандай ақпарат түсетінін біледі. Келесі теңдеулер LSTM қақпасын толықтай модельдейді.

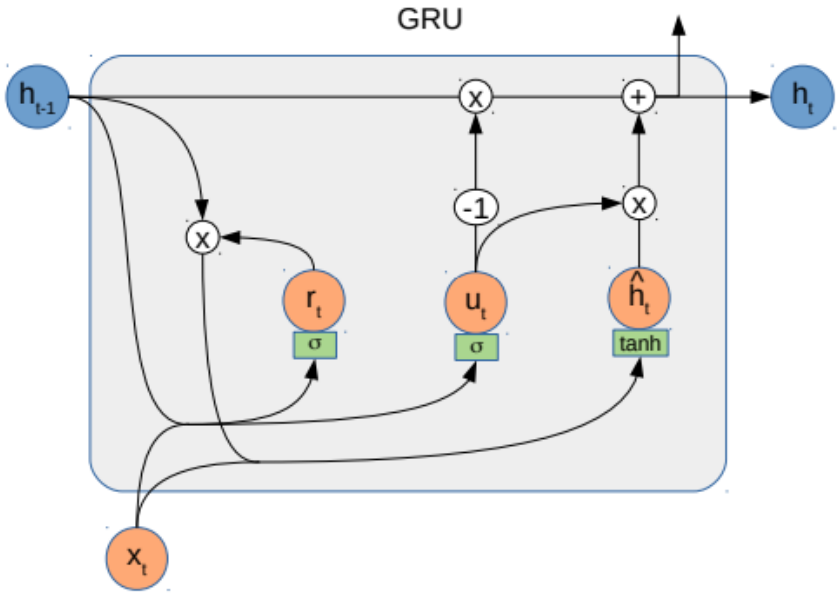
$$
\begin{aligned}
f_t &= (W_f[h_{t-1}, x_t] + b_f) \\
i_t &= (W_f[h_{t-1}, x_t] + b_i) \\
o_t &= (W_o[h_{t-1}, x_t] + b_o) \\
\hat{C}_t &= tan(W_c[h_{t-1}, x_t] + b_c) \\
C_t &= f_t \Theta\, C_{t-1} + i_t\, \Theta\, \hat{C}_t \\
h_t &= o_t tanh\,(C_t)
\end{aligned}
$$

(2.16)

мұндағы $W$ және $b$ – үйретілетін параметрлер, $\sigma$ - сигма тәрізді функция, $x$ - кіріс тізбегі, $i$ - кіріс қақпасы, $f$ - ұмыту қақпасы, $o$ - шығыс қақпасы, $C$ - күй векторы және $h$ - шығыс векторы және Θ элементтік туындыны білдіреді.

**Gated Recurrent Units** (GRU) аз параметрлері бар LSTM архитектурасына балама жабық RNN болып табылады [67]. Архитектуаның ең жақсы түрін анықтау үшін нақты консенсус жоқ. Мәселеге байланысты кейде LSTM өнімділігі жоғары болуы мүмкін, ал басқа сценарийлерде GRU жақсы нәтижелерді ұсынады [68]. Дегенмен, жалпы алғанда, әрбір нақты мәселе үшін екеуін де эксперимент жасау қажет.

GRU желісі мен LSTM желісінің негізгі айырмашылығы, GRU-да бір ғана қақпа бір уақытта ұмыту факторын және күй бірлігін жаңарту шешімін басқарады. 2.26-суретте екі қақпаны пайдаланатын бір GRU желісінің схемасы көрсетілген. Ағымдағы күйдің бөліктерін басқаратын $r$ қалпына келтіру қақпасы жаңа күйді есептеу үшін және $h$ шығысына қандай ақпарат жылжытылатынын басқаратын жаңарту қақпасы $u$ пайдаланылады.




Сурет 2.26 – Gated Recurrent Unit (GRU) ұяшығы

2.26-суретті белгілеу үшін алға қадамға арналған теңдеулер келесідей:

$$\begin{aligned} u_t &= (W_u[h_{t-1}, x_t] + b_u) \\ r_t &= (W_r[h_{t-1}, x_t] + b_r) \\ \hat{h}_t &= tanh(W_u[r_t\ h_{t-1}, x_t] + b_h) \\ h_t &= (1 - u_t)\ \ h_{t-1} + u_t\ \ \hat{h}_t) \end{aligned} \qquad (2.17)$$

мұндағы $W$ және $b$ - жаттықтырылатын параметрлер, $\sigma$ - сигма тәрізді функция, $x$ - кіріс реті, $r$ - қалпына келтіру қақпасы, $u$ - жаңарту қақпасы және $h$ - шығыс векторы және  элементтік өнімді білдіреді.

**Қолжазба тануда RNN қолдану**

Қолжазбаны тану (HTR) мәселесінде нейрондық желілерді бастапқы қолдану олардың жасырын Марков модельдерімен (HMM) біріктірілген гибридті тәсілдерге негізделген [69]. Тек RNN архитектураларын қолдану таңбаларды танумен ғана шектелді, себебі мақсатты функция үшін кіріс тізбегінің әрбір элементіне сәйкес жеке жаттығу сигналы қажет болды. Бұл өз кезегінде деректерді алдын ала сегменттеуді талап етті.

[70] зерттеу HTR мәселесі үшін толық RNN архитектурасын ұсынған алғашқы зерттеу жұмыс болып табылады. Олар жоғарыда аталған шектеуді шешу мақсатында декодер ретінде Connectionist Temporal Classification (CTC) деңгейін қолданды. Бұл қабат модельдің шығысы мен таңбалардың мақсатты тізбегі арасындағы автоматты туралауды (alignment) жүзеге асырады. CTC тек кіріс тізбегі шығыс тізбегінен ұзын немесе тең болуы керек деген шектеуді жүктейді.

Ұсынылған модель 100 жады ұяшығы бар бір қабатты екібағытты LSTM желісі болып табылады. Бұл жұмыстан кейін мұндай RNN құрылымы HTR мәселесіне арналған көптеген ғылыми жарияланымдарда кеңінен қолданыла бастады. CTC әдісінің арқасында RNN негізіндегі тәсілдердің HTR жүйелерінде танымалдығы күрт өсті. Бастапқы кезеңде ең көп қолданылған архитектура - көпөлшемді рекуррентті нейрондық желілер (MDRNN), әсіресе MDLSTM модельдері болды.

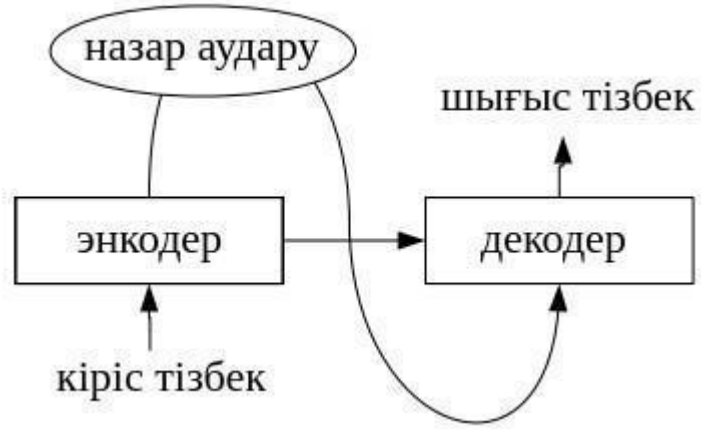
Соңғы жылдары офлайн қолжазбаны тану (HTR) мәселесіне қолданылатын жаңа архитектуралық тәсіл - энкодер-декодер құрылымына негізделген “тізбектен-тізбекке” (sequence-to-sequence, seq2seq) модельдер кеңінен таралуда [71]. Бұл архитектура реттілігі белгісіз (алдын ала тураланбаған) кіріс пен шығыс тізбектерінің арасындағы қатынастарды тиімді модельдеуге бағытталған және қолжазбаны үздіксіз офлайн тану міндеттерінде сәтті қолданылып келеді.

Seq2seq архитектурасының негізгі модификациясы 2014 жылы ұсынылған, ол ұзақ тізбектермен жұмыс істеудің тиімділігін арттыру үшін "назар аудару" (attention) механизмін қамтиды. Бұл нейрондық архитектура ақпарат тізбегін басқа бір тізбекке түрлендіруді талап ететін есептерге, атап айтқанда тізбекті түрлендіру (sequence transduction) мәселелерін шешуге бағытталған.

Мұндай архитектуралар сөйлеуді тану, машиналық аударма, мәтінді табиғи кескіндерден тану, тіпті бейнелердегі ерін қимылы арқылы сөйлеуді тану сияқты күрделі мәселелерде де кеңінен қолданылуда. Осы салалардағы тиімділігі seq2seq модельдерінің әмбебаптығын және бейімделгіштігін көрсетеді. 2.27-суретте seq2seq архитектурасының негізгі компоненттері мен олардың өзара әрекеттесуі бейнеленген. Бұл компоненттер келесі негізгі элементтерден тұрады:

- Энкодер - тұрақты ішкі кеңістіктегі айнымалы ұзындықты енгізу ретін кодтауға мүмкіндік береді. Ол қайталанатын қабаттардан тұрады, егер кіріс кескін немесе бейне болған жағдайда конволюциялық қабаттарды қамтиды.
- Декодер - шығысынан шығыс реті элементін элемент бойынша құрастыратын декодер. Ол әдетте бір бағытты RNN-ден тұрады.

● назар аудару механизмі - әрбір декодер қадамына сол қадам үшін ең сәйкес кодтаушы ақпаратқа қол жеткізуге мүмкіндік береді.




Сурет 2.27 – Seq2seq архитектурасының негізгі схемасы.

**Seq2seq энкодері**

Энкодер seq2seq архитектурасында айнымалы ұзындықта болуы мүмкін әрбір кіріс тізбегін бекітілген ұзындықтағы белгілер кеңістігіндегі тізбекке кодтауға жауапты. Осылайша, $x = (x_1, ..., x_{nx})$ айнымалы ұзындықтағы кіріс элементтер тізбегі берілген, ол кодтауыш арқылы, әдетте кіріс реті арқылы $h = (h_1, ..., h_T)$ тұрақты ұзындық тізбегіне түрлендіріледі, табиғатына байланысты алдыңғы түрлендіру және одан кейінгілері RNN *f* арқылы келесідей есептеледі:

$$(s, \dots, s_T) = g(x_1, \dots, x_{n_x})$$
$$h_t = f(s_t, h_{t-1}) \qquad (2.18)$$

мұндағы *g* ұзындықты теңестіру үшін кіріс тізбегінің үстіне жай толтыру немесе бір немесе бірнеше кескіндер (бір немесе бірнеше арналардың) кіріс болған жағдайда күрделі конволюционды архитектура немесе мәселенің сипаты мен кодтау деректеріне байланысты басқа опциялар болуы мүмкін. Сонымен қатар, $h_t \in IR_n - t$ уақытындағы жасырын күй.

Соңында $h = (h_1, ..., h_T)$ шығыс тізбегі дешифраторға жіберілетін *c* контекстік векторына түрлендіріледі. Модельде назар аудару механизмі болған жағдайда, ол әрбір дешифратор қадамына, $h = (h_1, ..., h_T)$ шифрлаушының толық шығу ретіне қол жеткізуге мүмкіндік береді.

Seq2seq үлгісіндегі энкодер архитектурасы өте бейімделгіш және оның нақты құрылымы мәселенің сипаты мен кіріс деректерінің түріне байланысты өзгеріп отырады. Қолжазбаны тану (HTR) міндетіне арналған seq2seq модельдеріндегі тәсілдердің негізгі айырмашылықтары көбіне энкодер бөлігінде, сондай-ақ кейде назар аудару (attention) компонентінде шоғырланған.

Энкодер құрылымы әртүрлі болуы мүмкін болғанымен, оның соңғы қабаты көбінесе рекуррентті нейрондық желі (RNN) болып табылады. Атап айтқанда,

LSTM немесе GRU сияқты ұзақ реттіліктердегі тәуелділіктерді тиімді модельдей алатын архитектуралар жиі қолданылады.

Декодер де әдетте сол энкодермен бірдей RNN типін қолданады, өйткені энкодердің соңғы күйі (ccc) декодердің бастапқы күйі ретінде пайдаланылады - бұл seq2seq архитектурасында кең таралған тәсіл.

Егер энкодер ретінде екібағытты RNN (bidirectional RNN) қолданылса, онда декодерді инициализациялау үшін: алдыңғы бағыттағы қабаттың соңғы күйін, кері бағыттағы қабаттың соңғы күйін, немесе екі бағыттағы күйлердің біріктірілген комбинациясын (мысалы, қосындысы, конкатенациясы) пайдалану мүмкін.

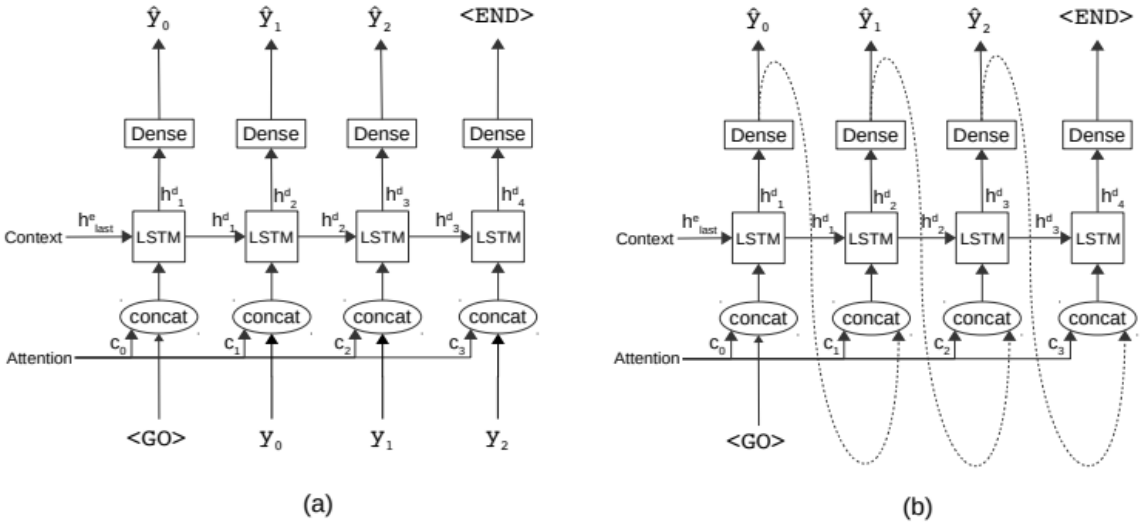
**Seq2seq декодері**

seq2seq архитектурасында декодер әдетте кіріс реттілігінен әртүрлі ұзындықтағы модельдің шығыс тізбегін генерациялауға жауапты. CTC жағдайынан айырмашылығы, шығыс реті кіріс реттілігінен ұзағырақ болуы мүмкін. Декодер $y_t$ шығыс тізбегінің әрбір элементін $y = (y_1, y_{t-1})$ тізбегіндегі алдыңғы элементтердің және назар аудару механизмі қамтамасыз ететін $c$ контекстінің функциясы ретінде болжайды. Осылайша, дешифратор $y$ шығыс тізбегіндегі ықтималдықты теңдеуде көрсетілгендей реттіліктің әрбір алдыңғы элементінің реттелген шартты ықтималдықтарының көбейтіндісі ретінде анықтайды:

$$p(y) = \prod_{t=1}^{T} \quad p(y_t|\{y_1, \dots, y_{t-1}\}, c) \tag{2.19}$$

Декодер әдетте бірбағытты RNN ретінде іске асырылады, әдетте LSTM немесе GRU түрлерін айтады. RNN шығысы оның өлшемін $y_t$ мүмкін болатын мәндер жиынының кардиналына сәйкестендіру үшін тығыз $d$ қабатымен қосымша түрленеді. Соңында $p(y_t)$ ықтималдығының таралу бағасын алу үшін белсендіру *softmax* функциясы қолданылады .

RNN дешифраторының $h_0$ бастапқы күйі әдетте соңғы кодтаушының соңғы күйінің мәнімен инициализацияланады. Бұл 2.28-суретте энкодер мен декодер арасында анықталған тікелей байланыс арқылы көрсетілген. Оны кездейсоқ инициализациялауға да болады.




Сурет 2.28 – LSTM негізіндегі декодер схемасы: (a) оқыту кезеңі (мұғалім мәжбүрлеу әдісі) және (b) бағалау.

Шығыс тізбегінің бастапқы мәндерін декодерге кіріс ретінде беру арқылы seq2seq моделін оқыту әдісі "мұғалім мәжбүрлеуі" (teacher forcing) деп аталады. Бұл әдісте декодер әрбір уақыт қадамында өз алдыңғы болжамын емес, нақты мақсатты таңбаны (яғни, t−1 сәтіндегі дұрыс шығысты) кіріс ретінде алады. Дегенмен, мұндай оқыту стратегиясы бағалау (инференс) кезеңінде қиындықтар туғызуы мүмкін, өйткені модель енді алдыңғы нақты шығыстардың орнына өз болжамдарын пайдаланады. Егер генерациялау процесінің бір қадамында қате орын алса, ол шартты ықтималдықтар тізбегі арқылы жинақталып, келесі қадамдарға әсер етеді. Бұл модельдің шығыс реттілігінің нақты реттіліктен біртіндеп ауытқуына және, сәйкесінше, жалпы өнімділіктің төмендеуіне алып келуі мүмкін.

Мұғалім мәжбүрлеу (teacher forcing) әдісінің кейбір жағымсыз жақтары да бар, олардың бірі - модельдің шамадан тыс үйренуіне (overfitting) бейімділігі. Бұл әсіресе декодтау процесінің әрбір $t$ қадамында кіріс ретінде $y_{t-1}$ символының нақты мәні пайдаланылған кезде байқалады. Аталған мәселені жеңілдету үшін бірнеше балама тәсілдер ұсынылған. Солардың бірі -келесі қадамға кіріс ретінде нақты мән мен алдыңғы қадамдағы модельдің бағалауын біріктіру тәсілі [72]. Бұл әдіс модельге өз қателіктерінен үйренуге мүмкіндік береді және бағалау кезеңіне неғұрлым жақын жағдай жасайды.

**seq2seq назар аудару механизмі**

Бұл компонент бастапқы seq2seq мәселесімен бетпе-бет келеді, өйткені ол кодтауыш пен дешифраторды тек $c$ күй векторымен байланыстырды. Бұл ұзын көлемді тізбектері бар seq2seq үлгілерін пайдалануды шектейді, себебі

тізбектердің ұзындығы ұлғайған кезде бастапқы үлгінің өнімділігі тез нашарлайды [73].

Назар аудару механизмі дешифратордың әрбір қадамын осы қадамды декодтау үшін өзекті болуы мүмкін кодтаушы ретінің барлық элементтерімен байланыстыру арқылы алдыңғы мәселені шешеді. Содан кейін модель алдыңғы элементтердің болжамдарына және назар аудару механизмі қамтамасыз ететін осы контекстке негізделген шығыс ретінің әрбір элементін болжайды.

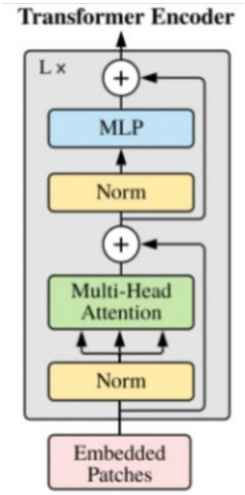
Неғұрлым формальды түрде және алдыңғы бөлімдердің белгілеулерінен кейін, энкодер шығысының ұзындығы $N_e$ бойынша әрбір шығыс $t$ уақытындағы $c_t$ назар аудару векторы келесідей есептеледі:

$$e_{t,i} = \omega^T tanh\,(Wh_i^e + Vh_{t-1}^d + b)$$
$$a_t = softmax(e_t)$$
$$c_t = \sum_{i=1}^{N_e} \quad a_{t,i}h_i \qquad (2.20)$$

мұндағы $w$ және $b$ - жаттықтырылатын салмақ және ығысу векторлары, $W$ және $V$ - жаттықтырылатын матрицалар, $h_{i-i}^e$ $he_{i-i} \in 0, 1, ...N_e$ уақытындағы энкодердің жасырын күйі, және $h_{t-1}^d$ $t-1$ уақытындағы декодердің жасырын күйі. Назар аудару механизмі нейрондық желі деңгейі ретінде іске асырылады, ол энкодердің және декодердің алдыңғы күйлерін белсендіру функциясы ретінде гиперболалық тангенспен біріктіреді. $e_t$ векторы $i$ позициясында ағымдағы дешифратор қадамын $t$ $i$-ші энкодер жасырын күйге қоюға қаншалықты көңіл бөлу керектігін көрсететін баллды қамтиды. Содан кейін біз оны қалыпқа келтіру үшін softmax функциясын қолданамыз және кодтаудың барлық күйінде назар аудару маскаларын жасаймыз. Соңғы $c_t$ назар аудару векторы 2.28-суретте көрсетілгендей декодер қабатының кірісімен біріктірілген.

**Трансформерлер**

Соңғы жылдары назар аудару механизмдері жоғары өзектілікке ие болды, себебі олар қазіргі таңда кеңінен қолданылатын трансформер (Transformer) деп аталатын нейрондық желілер архитектурасының негізін құрайды [74]. Бұл архитектура мәтіндік ақпаратты ғана емес, сонымен қатар кескіндерді модельдеуде де елеулі жетістіктерге қол жеткізді. Трансформерлер құрылымы жағынан seq2seq архитектурасына ұқсас энкодер-декодер типіндегі модельдер болып табылады. Алайда, негізгі айырмашылығы - трансформер архитектурасында энкодер мен декодер бөліктері дәстүрлі нейрондық желілер сияқты CNN немесе RNN-ге негізделмеген, олардың орнына барлық есептеулер толықтай назар аудару механизмдері мен позициялық ендірулерге (positional encodings) сүйенеді (сурет 2.29).

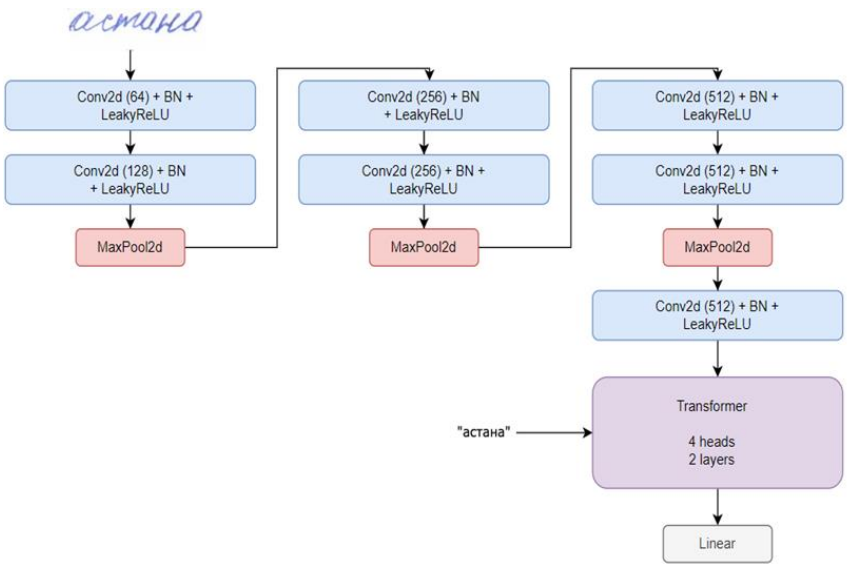



Сурет 2.29 – Трансформер энкодер архитектурасы

Трансформер архитектурасының RNN негізіндегі энкодер-декодер үлгілерімен салыстырғанда екі негізгі артықшылығы бар. Біріншіден, трансформерлердің орындалуы параллельді түрде жүзеге асады, бұл үлкен көлемдегі параметрлерге қарамастан, есептеу құны мен оқыту уақытының RNN-мен салыстырғанда едәуір төмен болуына мүмкіндік береді. Ал RNN құрылымдарына тән тізбекті өңдеу тәсілі параллельді есептеулерді шектейді және оқыту уақытын ұзартады. Екіншіден, трансформерлер ұзақ мерзімді тәуелділіктерді тиімдірек үйрене алады. Бұл олардың құрылымында ақпараттың алға немесе кері бағытта таралу жолдарының ұзындығы тұрақты болуымен түсіндіріледі. Яғни, трансформерде кез келген екі элемент арасындағы байланыс бір ғана назар аудару қабаты арқылы жүзеге асады. Керісінше, RNN негізіндегі seq2seq архитектураларында ақпараттың таралу жолдарының ұзындығы O(n) тәртібінде болады, мұндағы n - кіріс тізбегінің ұзындығы. Жолдар неғұрлым ұзақ болса, модельдің ұзақ мерзімді тәуелділіктерді үйрену қабілеті соғұрлым шектеледі. Сондықтан қысқа жолдар модельдің үйрену тиімділігін арттыра түседі.

2.5.1 Нейрондық желілерді оқыту және нәтижелері

Бұл бөлімде қазақ қолжазба мәтіндерін тануға арналған трансформер негізіндегі архитектура құрылды. Бұл архитектура жеңіл салмақты конволюциялық кодтаушымен жабдықталған (2.30-сурет).

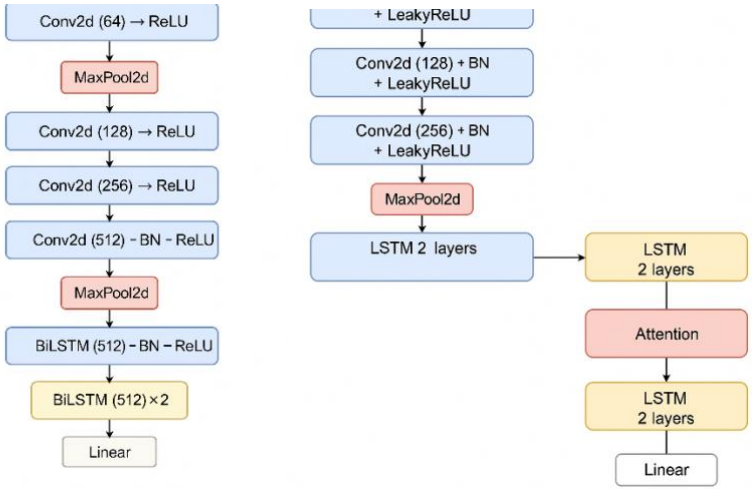



Сурет 2.30 – OCR жүйесіне арналған трансформер негізіндегі тану моделі архитектурасы

Архитектурада екі негізгі бөлек модуль бар: біріншісі бейнеден керекті белгілерді алады (CNN), екіншісі сол белгілердің реттілігін анықтайды (Transformer). Модельге берілетін дерек – әр пиксельге бөлінген бір сөздің суреті (мысалы, астана деген сөздің суреті). Ол суретті алдын ала керекті мөлшерге келтіріп аламыз (1×64×160). CNN бөлімінде бірнеше қабат бар: үйірткі қабаты (Conv2d), нормализация қабаты (BatchNorm) және LeakyReLU активация қабаты. Осылардың арқасында суреттен керекті белгілерді алып, оның кеңістіктік мөлшерін кішірейтіп, мағынасын тереңдетеміз.

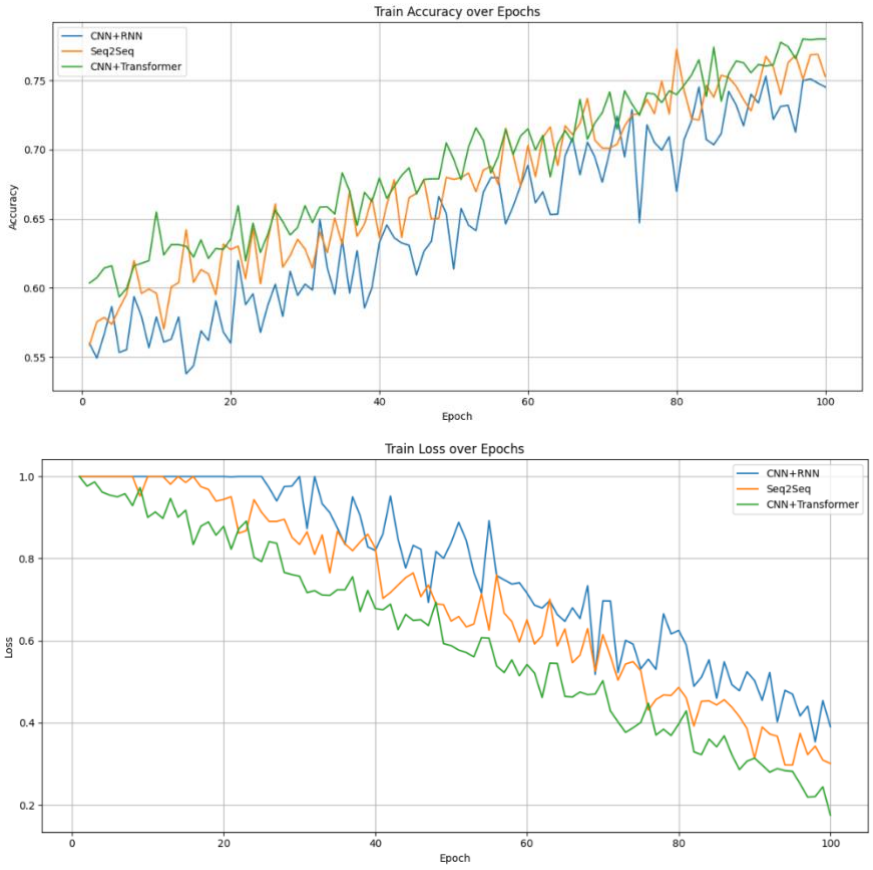
CNN-нен кейін алынған белгілер тізбегі Transformer Encoder-ге беріледі. Ол ұзақ тізбектік байланыстарды анықтап, әр белгінің мағынасын ескереді. Модель алдымен input sequence-ті positional encoding арқылы толықтырады. Сосын self-attention механизмін пайдаланып, әр позициядағы белгінің басқа позициялармен байланысын қарастырады. Transformer-ден шыққан тізбек әр позиция үшін Linear қабат арқылы символға (әріп, сан, т.б.) айналады. Әр позициядағы вектор сызықтық қабат арқылы softmax функциясына беріліп, ең ықтимал символ таңдалады.

Салыстыру үшін CNN + RNN және Sequence2sequence архитектуралар ды қарастырылды (сурет 2.31)

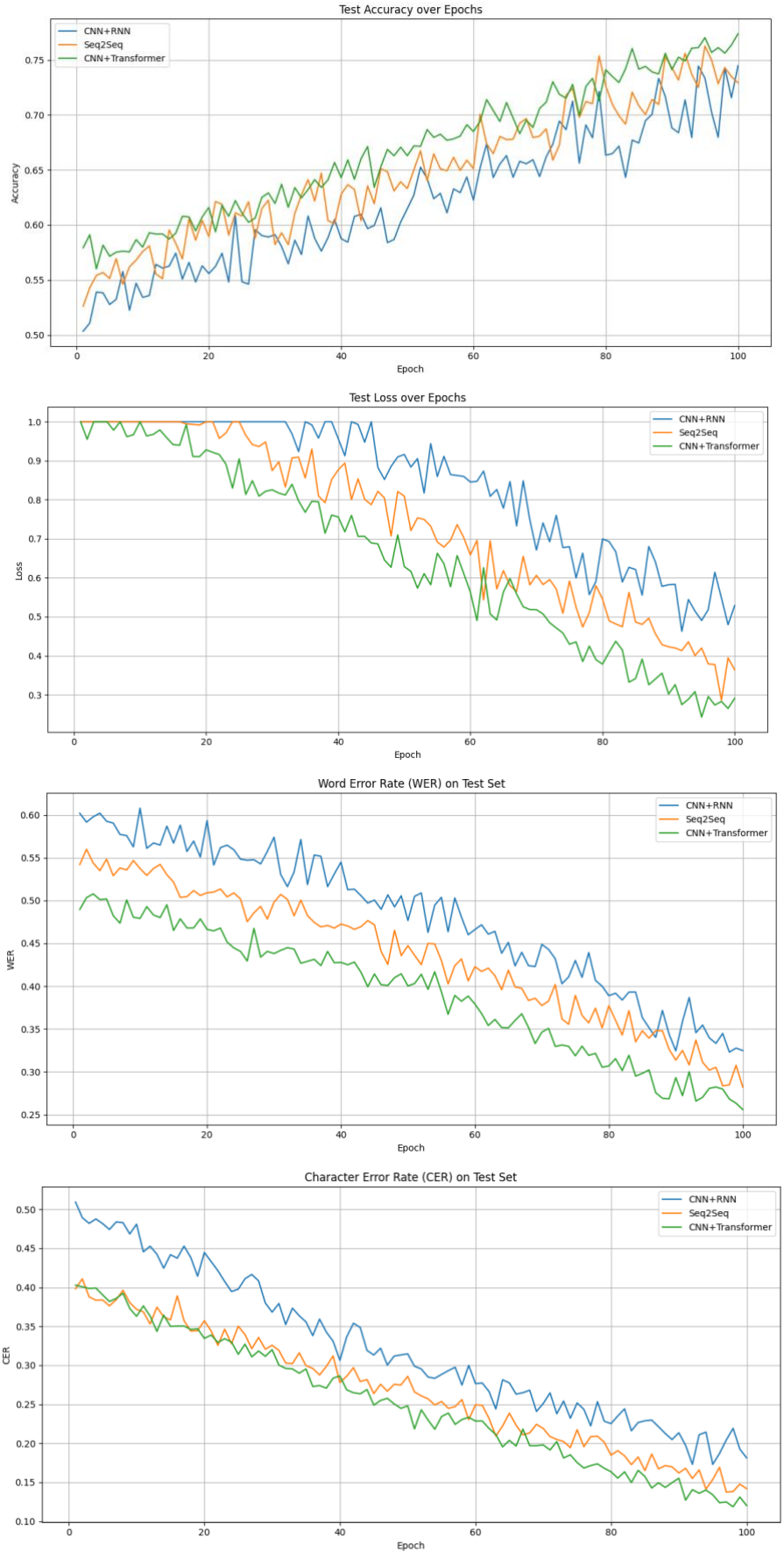



Сурет 2.31 – CNN + RNN және Sequence2sequence архитектуралары

Оқыту нәтижелері 2.32 – 2.34 суреттерде көрсетілген:



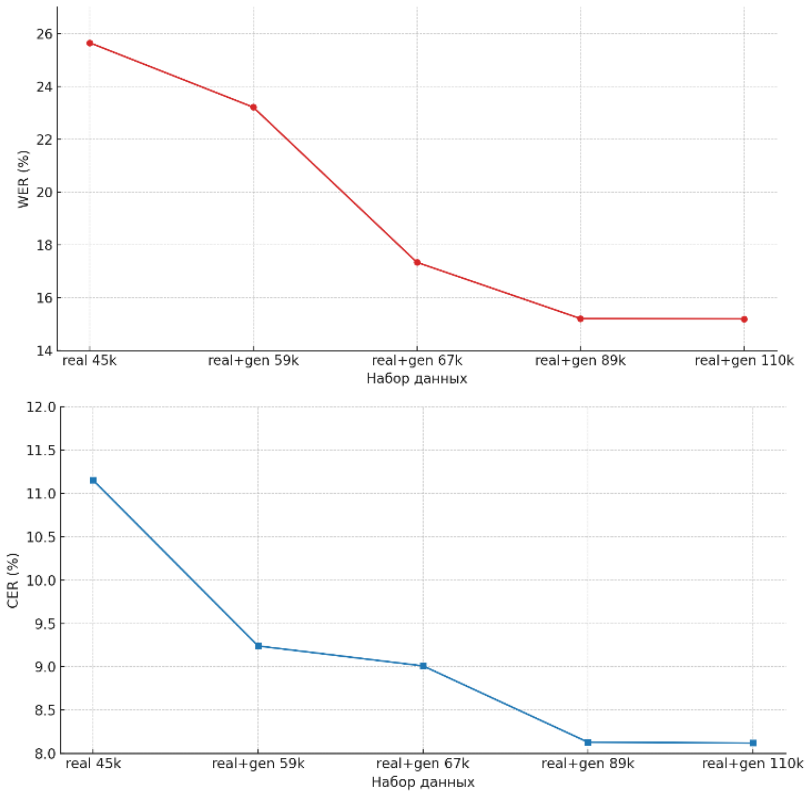

Сурет 2.32 – Моделдерің оқыту кезіндегі дәлдігі мен қателігі




Сурет 2.33 – Моделдерің метрикаларының көрсеткіштері

Трансформер архитектурасына негізделген модельдер бастапқы деректер жиынтығын әртүрлі түрлендіру (data augmentation) әдістерін қолдана отырып,

оны оқытуға арналған кеңейтілген деректер жиынына айналдырады. Мұндай тәсіл модельдің артық үйрену (overfitting) ықтималдығын төмендетіп, оқыту үлгілерінің әртүрлілігін арттыру арқылы тану дәлдігін жақсартуға мүмкіндік береді. Айта кету керек, трансформер негізіндегі модельдердің тиімділігіне тікелей әсер ететін негізгі факторлардың бірі – деректер жиынының көлемі мен сапасы. Сондықтан бұл эксперимент деректердің генерациялау әдісі арқылы тану сапасына нақты қаншалықты әсер етуге болатынын бағалауға бағытталған. Модельдің тиімділігін бағалау үшін біз Character Error Rate (CER) және Word Error Rate (WER) метрикалары қолданылды [75]. Генерацияланған деректердің бұл метрикаларға қаншалықты әсер ететінін бағалау мақсатында, нақты деректер жиынына әртүрлі көлемдегі синтетикалық деректерді қосу арқылы бірнеше эксперимент жүргізілді. Алдымен модель тек нақты деректерге негізделіп оқытылды, бұл жиынтықта жалпы саны 45 211 сөз қамтылған. Одан кейін біз әрбір нақты сөзге сәйкес қосымша 30%, 50% және 100% көлемінде синтетикалық кескіндер генерацияладық. Нәтижесінде сәйкесінше 14 мың, 22 мың және 44 мың сөз кескіндері жасалды. Әрбір деректер комбинациясына сәйкес модель жеке-жеке оқытылып, оның нәтижелері төмендегі кестеде көрсетілді. 3.2-кестеде әртүрлі деректер жиынтықтарында оқытылған модельдердің орташа WER (%) және CER (%) көрсеткіштері берілген. Тек нақты деректер негізінде сынақтан өткізілген жағдайда, Word Error Rate (WER) - 25.65%, ал Character Error Rate (CER) - 11.15% құрады [79, 80].




Сурет 2.34 – Деректер санына сай метрикалардың өзгерісі

Кесте 2.5 - Деректер жиынтығының түрлі комбинацияларына байланысты модельдің WER және CER нәтижелері

| Dataset | WER (%) | CER (%) |
|---|---|---|
| real 45k | 25.65 | 11.15 |
| real + generated 59k | 23.21 | 9.24 |
| real + generated 67k | 17.34 | 9.01 |
| real + generated 89k | 15.21 | 8.13 |

2.5-кестеден байқағанымыздай, генерацияланған деректермен оқытылған модельдің нәтижелері бойынша WER және CER көрсеткіштері тиісінше 10.44% және 3.02%-ға айтарлықтай жақсарғаны байқалады. Модель генерацияланған деректер негізінде оқытылғаннан кейін, оның нақты деректердегі тану дәлдігі тексерілді.

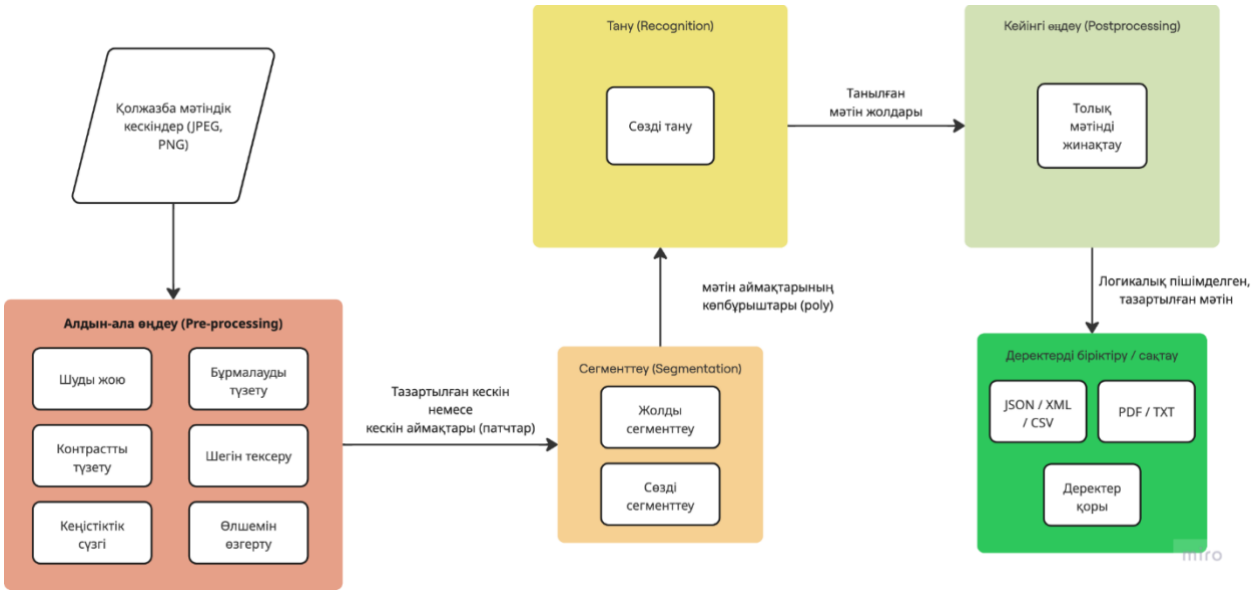
Сынақ деректер жиынтығында модельдің CER (Character Error Rate) көрсеткіші 17.11% деңгейінде тіркелді. Алынған нәтижелер абсолютті дәлдікті талап етпейтін жобаларда бұл модельді практикалық қолдануға жеткілікті екенін көрсетеді. Сонымен қатар, нейрондық желі арқылы алынған мәтін жолдарына постпроцессинг әдістерін қолдану арқылы дәлдікті бірнеше пайызға арттыруға болатынын ескеру қажет. Әсіресе, сөздіктермен жұмыс істеу бұл жерде маңызды рөл атқарады, себебі көптеген сөздер тілдік корпусқа белгілі және әріптер тізбегі көбіне кездейсоқ болмайды.

Модельдің дәлдігін басқа зерттеу жұмыстарымен объективті түрде салыстыру мүмкін болмады, өйткені қазақ тіліне арналған қолжазба мәтіндерді тану бойынша ғылыми жұмыстар әлі де аз. Осы себепті салыстыру кирилл әліпбиіндегі орыс тілді қолжазбаларды тану бойынша жасалған зерттеулермен жүргізілді. Бұл бағыттар ұқсас болғанымен, қазақ тіліне тән "ә", "ң", "і", "ғ", "ү", "ұ", "h", "ө" сияқты әріптердің болуына байланысты айырмашылықтар бар.

# 3 ЖҮЙЕНІҢ АҚПАРАТТЫҚ МОДЕЛІН ҚҰРУ ЖӘНЕ ЖҮЗЕГЕ АСЫРУ

## 3.1 Қолжазбаны тану жүйесінің архитектуралық компоненттері

Бұл бөлімде қазақ тіліндегі жазылған қолжазба мәтінді тануға арналған жүйенің ақпараттық моделі мен оның жүзеге асырылуы туралы жазылады. Жүйе қолданушыға арналғаннан кейін end-to-end моделін қолданамыз. Ол өз кезегінде қолданушыға соңғы нәтижені алу үшін барлық машиналық оқыту моделдерінің белгілі бір тізбек бойынша жұмыс істеуін білдіреді. Төменде 3.1-суретте көрсетілгендей жүйе архитектурасы бірнеше бөліктерден құралған.



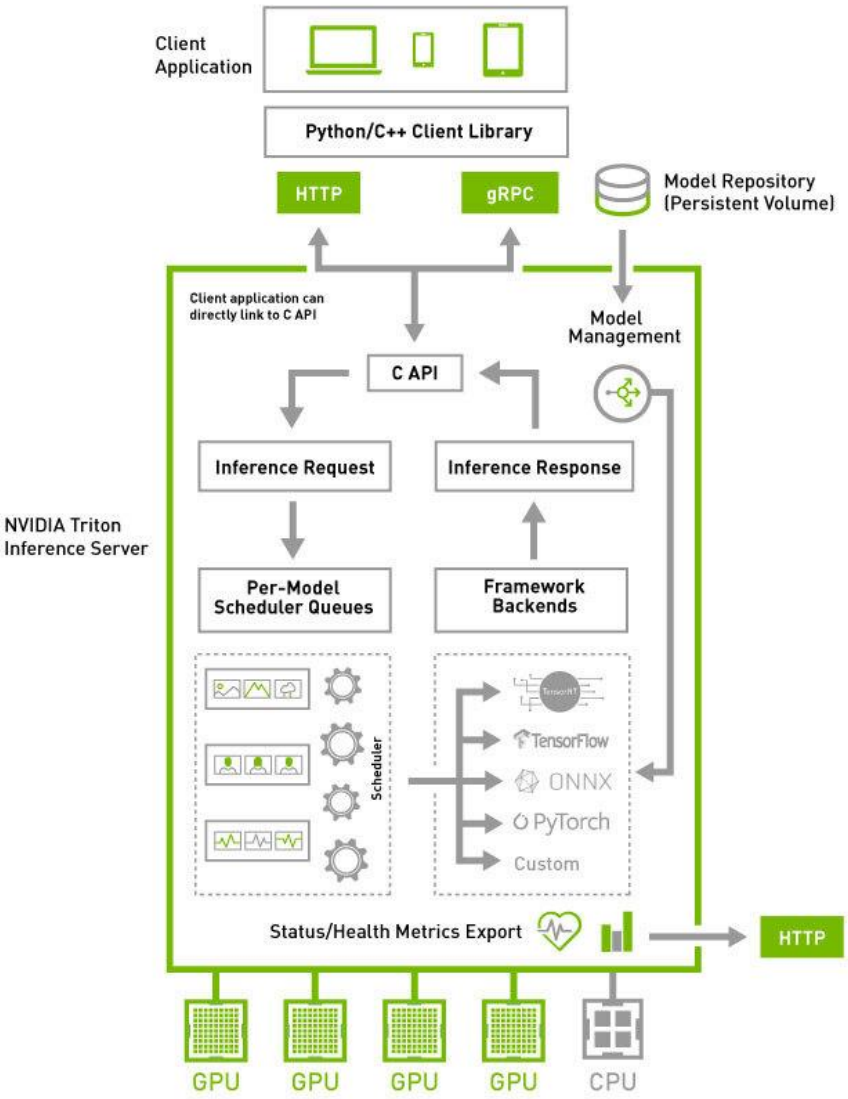

Сурет 3.1 – Жүйенің ақпараттық моделі

### 3.1.1 Модельдерді Nvidia Triton платформасында әзірлеу

Бірнеше модельдерді бір нәтиже алу үшін тиімді пайдалануға арналған бірнеше құралдар бар. Мысалы, PyTorch және Tensorflow фреймворктарының TorchServe және TensorflowServe сияқты платформалары модельдерді қызмет көрсетуге арналған. BentoML ML қолданбаларын оқытуға және орналастыруға арналған толық жинақты ұсынады.

NVIDIA Triton Inference Server немесе Triton — орналастыру кезеңін қарастыратын ашық кодты бағдарламалық құрал [76]. Ол өзінің толық құжаттамасымен, икемділігімен және аппараттық құралды оңтайландыру опцияларымен ерекшеленеді.

Triton көптеген машиналық оқыту (ML) құрылымдарына, сондай-ақ C++ және Python тілдерінде серверлерде қолдауды ұсынады. Бұл машиналық оқыту инфрақұрылымын жеңілдетуге мүмкіндік беретін әртүрлі фреймворктар үшін бірнеше қорытынды серверлерінің қажеттілігін азайтады. Ол NVIDIA графикалық процессорларымен үздіксіз жұмыс істейді және аппараттық құралдарды пайдалануды арттыра отырып, бірнеше GPU-да үлгіні бір уақытта орындауға мүмкіндік береді және Azure, AWS, GCP және басқа да көптеген шешімдерде үлгіге қызмет көрсету үшін пайдаланылады.(3.2-сурет)




Сурет 3.2 – Triton Server архитектурасы [76]

Машиналық оқыту моделдерін Тритон серверінде пайдалану үшін алдымен оларды оңтайландыру қажет. Тритон TensorRT, TensorFlow, PyTorch, ONNX, OpenVINO, Python, RAPIDS FIL және т.б. терең және машиналық оқыту моделдерін орналастыруға мүмкіндік береді. Ол NVIDIA GPU, x86 және ARM CPU немесе AWS Inferentia жүйесінде бұлт, деректер орталығы, жиек және ендірілген құрылғыларда жұмыс істеу мүмкіндігі бар. Тритон сервері нақты уақытты, топтаманы, ансамбльдерді және аудио/бейне ағынын қоса алғанда, көптеген сұрау түрлері үшін оңтайландырылған өнімділікті ұсынады. Оның негізгі қолдану жағдайлары:

- Бір серверде бірнеше модельдерге қызмет көрсету.
- Серверді қайта қосусыз динамикалық модельдерді жүктеу және түсіру.
- Нәтижелерге жету үшін бірнеше модельдерді бірге пайдалануға мүмкіндік беретін ансамбльдік қорытынды.
- A/B сынағы мен модельдердің нұсқаларын басқару.

TensorRT – бұл тек Nvidia графикалық процессорларында жұмыс істейтін CUDA негізінде жасалған қорытынды жеделдету кітапханасы. Torch пішімі арқылы әртүрлі терең оқыту шеңберлерінің үлгілерін TensorRT үлгілеріне түрлендіруге болады. Модельдерді түрлендіру кезінде TensorRT сонымен қатар кванттау және ядроны автоматты баптау арқылы қорытынды өнімділікті оңтайландырады. Ядроны автоматты баптауда CUDA ядроларының саны қарастырылады. Нәтижесінде, TensorRT үлгісі әртүрлі TensorRT нұсқалары және/немесе әртүрлі GPU түрлері арасында алмастырылмайды.

Модельді Тритон жүйесінде қолданбас бұрын, оны Torch пішіміне түрлендіру керек. Torch – әртүрлі терең оқыту фреймворктары арасында үлгілерді тасымалдауға мүмкіндік беретін формат.

Алдымен берілген бейнеден қолжазба сөздерді табуға арналған моделді конвертация жасаймыз. Ол үшін оқытылған модельді Torch фраймворкы арқылы шақырамыз:

```
import torch
import torchvision.models as models
model = torch.load("handwritten_detection.pt")
model.eval()
```

PyTorch-тан ONNX-ке түрлендіру процесі келесілерді талап етеді:

- Модель кіріс өлшемі - пакет саны, кескін арналарын және оның пішінін көрсететін [patch_size, channels, image_size, image_size] пішімінде болады. Біздің жағдайда модельдің кіріс өлшемі: [4, 3, 640, 640]

```
model_input = torch.rand(4,3,640,640)
```

- Модель шығыс өлшемі - қолжазба мәтін класы, сөздің координаталары, болжау ықтималдығы[class, bbox, prob]

```
input_names = [ "actual_input" ]
```

```
output_names = [ "output" ]
```

Келесі қадам модельді ONNX түріне түрлендіру үшін `torch.onnx.export` функциясын пайдалану болып табылады. Бұл функция келесі деректерді қажет етеді:

- *Үлгі*
- *Жалған енгізу*
- *Экспортталған файлдың аты*
- *Енгізу атаулары*
- *Шығару атаулары*
- *Үйретілген параметр салмақтарының үлгі файлында сақталатынын немесе сақталмайтынын анықтайтын `export_params`*

```
torch.onnx.export(model,
                  dummy_input,
                  "handwritten_detection.onnx",
                  verbose=False,
                  input_names=input_names,
                  output_names=output_names,
                  export_params=True,
                  )
```

Onnx форматына ауысқан модельдердің дұрыстығын тексеру үшін onnxruntime арқылы жүзеге асырып көреміз:

```
import onnxruntime as onnxrt
onnx_session= onnxrt.InferenceSession("handwritten_detection.onnx")
onnx_inputs= {onnx_session.get_inputs()[0].name:to_numpy(img)}
onnx_output = onnx_session.run(None, onnx_inputs)
img_label = onnx_outputort_outs[0]
```

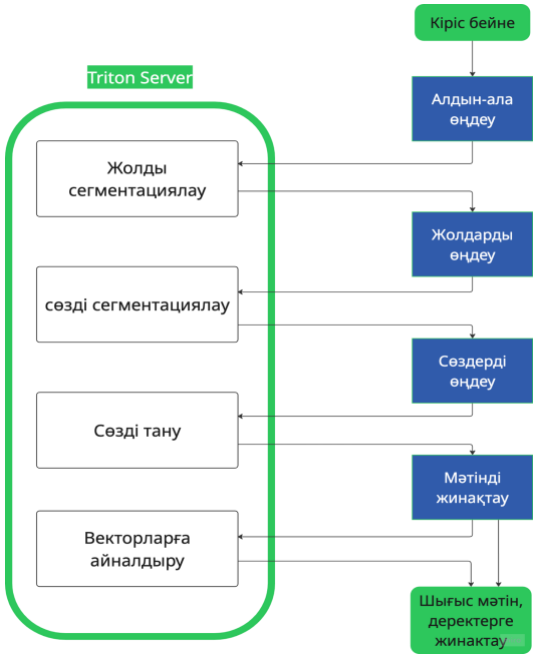
Осылай қалған барлық модельдерді түрлендіреміз. Дайындалған модельдерді Тритон серверіне жүктеу үшін келесі қадамдарды орындау қажет. Модель нұсқасы бар қалтаға сақтаймыз. Тритон барлық үлгілердің нұсқа нөмірі бар бөлек қалтада болуын талап етеді. Әрі қарай, белгілі бір нұсқаға сұрау жіберуге болады. Сондай-ақ, осы механизмді пайдалана отырып, үлгі нұсқаларын тікелей жұмыс өніміне ауыстыруға болады.

```
model_path = (
       Path(__file__).resolve().parents[1]
       / "model_repository"
       / "stn"
       / "1"
       / "handwritten_detection.onnx"
   )
traced_model.save(model_path)
```

Модель конфигурацияны орнатамыз. Анықтауды қажет ететін басты нәрсе - сервер, партиялар (Тритон пакеттерде өңдей алатындай), модельге кірістер мен шығыстар. Біздің жағдайда ол:

```
backend: "onnxruntime"
max_batch_size: 32
input [
    {
        name: "input__0"
        data_type: TYPE_FP32
        dims: [ 4,3, 640, 640 ]
    }
]
output [
    {
        name: "output__0"
        data_type: TYPE_FP32
        dims: [ 4, 16128, 1]
    }
]
```

Осы қадамдар барлық модельдерге орындалғаннан кейін Тритон жүйесі дайын болды.




Сурет 3.3 – Тритон серверіндегі модельдер байланысы

3.1.2 Жүйенің нәтижелерін тестілеу және бағалау

Жүйенің жұмысын толыққанды тексеру үшін Тритон серверге grpc протоколымен қолжазба мәтін жазылған қажетті суретті жіберсе жеткілікті. Ол үшін Python бағдарламалау тілін қолданса жеткілікті (сурет 3.4).

```python
triton_client = httpclient.InferenceServerClient(url="localhost:8000")
model_name = "plate_recognition"
image = cv2.imread(img_path)

inputs, outputs = [], []

inputs.append(httpclient.InferInput("input__0", image.shape, "UINT8"))
inputs[0].set_data_from_numpy(image)

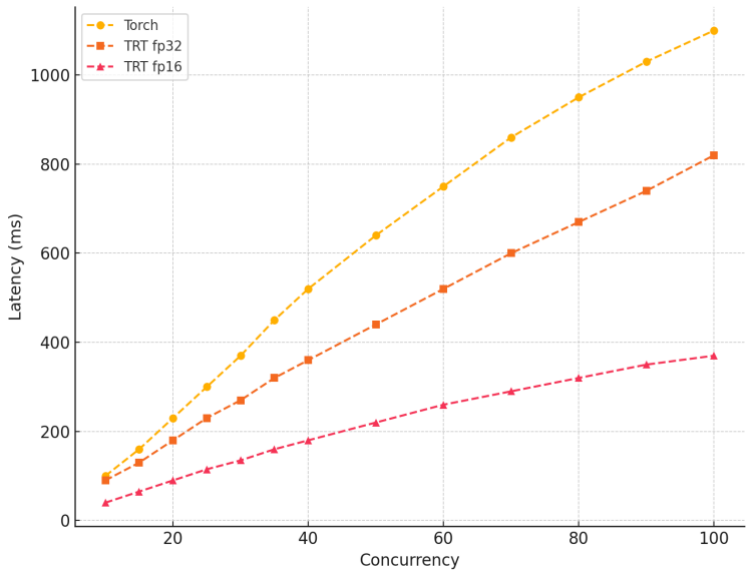
outputs.append(httpclient.InferRequestedOutput("coordinates", binary_data=False))
outputs.append(httpclient.InferRequestedOutput("texts", binary_data=False))

results = triton_client.infer(
    model_name=model_name,
    inputs=inputs,
    outputs=outputs,
)

coordinates = results.as_numpy("coordinates")
texts = results.as_numpy("texts")
```

Сурет 3.4 – Тритон серверіне сұраныс жіберу

Төменде көрсетілген графикте модельдердің Tensorrt және Onnx жеделдету технологияларының жұмыс істеу уақыттары салыстырылған (сурет 3.5). TensorRT үлгілері оңтайландырылмаған Torch моделіне қарағанда жоғары өткізу қабілеттілігін қамтамасыз ететінін көреміз. Қабатты біріктіру және ядроны автоматты баптау арқылы TensorRT fp32 моделі Torch үлгісіне қарағанда біршама жоғары өткізу қабілетіне ие болады. Біз fp16 кванттауын қабылдау өткізу қабілеттілігін айтарлықтай жақсартатынын және динамикалық топтаманы қабылдау жоғары сәйкестік режимінде өткізу қабілеттілігін шамамен 33%-ға одан әрі жақсартатынын байқаймыз.




Сурет 3.5 – Torch үлгісінің, динамикалық топтамасы жоқ TensoRT fp32 моделінің, динамикалық топтамасы жоқ TensorRT fp16 моделінің және динамикалық пакеттеуі бар TensorRT fp16 моделінің өткізу мүмкіндіктері

Torch және TensorRT модельдерінің кідірістері параллельдіктерге пропорционалды өсетінін көреміз. TensorRT fp32 моделі оңтайландырылмаған Torch үлгісінен 24,6%-ға жылдамырақ жұмыс істейді. Сондай-ақ, біз fp16 кванттауын қабылдау қорытындыны күрт жеделдететінін көреміз.

### 3.2 Құжаттардан қажетті ақпаратты іздеу жүйесін әзірлеу

RAG (Retreival Augmented Generation) дегеніміз - ақпарат іздеу мен мәтін жазуды бірге қосатын жаңа технология. Оны 2020 жылы Facebook AI Research (FAIR) зерттеушілері ойлап тапты [77]. RAG-тың басты плюсы - үлкен тіл модельдерінің (LLM) білімін сыртқы деректермен байытып, нақты жауаптар беруі. RAG жүйесінде екі бөлік бар:

- Іздеуші (Retriever) – векторлық кеңістіктегі ұқсастық арқылы керекті мәліметті тауып береді.
- Генератор (Generator) - табылған мәліметтерді пайдаланып, сұраққа жауап жазады.

Кәдімгі тіл модельдерінен айырмашылығы, RAG тек өзінде бар білімді пайдаланады. Ол әр сұрақ сайын ақпарат көзінен керекті деректерді іздейді. Осылайша, модельдің жауабы дәлірек болады, және қате (галлюцинация) жіберу ықтималдығы азаяды. RAG жүйесінде іздеу бөлігі векторлық бейнелеу арқылы істейді. Мәтіндер алдымен трансформер модельдерімен өңделіп, сандарға айналдырылады. Ұқсас мағынадағы мәтіндер бұл сандық кеңістікте бір-біріне жақын тұрады, мұны косинус ұқсастығы немесе Евклид арақашықтығы арқылы білуге болады. Біз мәтіндерді векторға түрлендіру үшін Sentence-BERT деген модельді қолданамыз [78]. Бұл модель BERT-ке ұқсайды және мағынасы бірдей сөйлемдерді жақын қылып көрсетеді. Sentence-BERT кез келген мәтінді 384 өлшемді векторға өзгерте алады және ол қазақ, орыс тілдерін де қолдайды.

Іздеу жүйесін құру үшін алдымен біз архивтегі қолжазбаларды таныған кейін бірден векторлық форматқа ауыстырамыз. Мұны істеу үшін әр құжаттағы мәтінді Sentence-BERT архитектурасына негізделген sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2 моделімен өңдейміз. Осылайша, біз 384 өлшемді вектор аламыз. Бұл модель қазақ, орыс тілдерінде және тағы басқа тілдерде жұмыс істейді, сондықтан мұрағат құжаттарының әртүрлі тілдерге бейімделе алады. Содан кейін векторларды FAISS (Facebook AI Similarity Search) жүйесі арқылы индекстеп, арнайы векторлық мәліметтер базасында сақтаймыз. FAISS көмегімен семантикалық ұқсас құжаттарды косинустық ұқсастық немесе евклид арақашықтығы арқылы тез арада табамыз. Жалпы жүйеде 105 архив құжаты өңделді. Құжаттардан алынған мәтіндерді жеке-жеке векторға ауыстырып, RAG (Retrieval-Augmented Generation) жүйесі үшін семантикалық іздеу базасын жасадық. Бұл база арқылы пайдаланушының сұранысына ең жақын құжаттарды табуға болады.

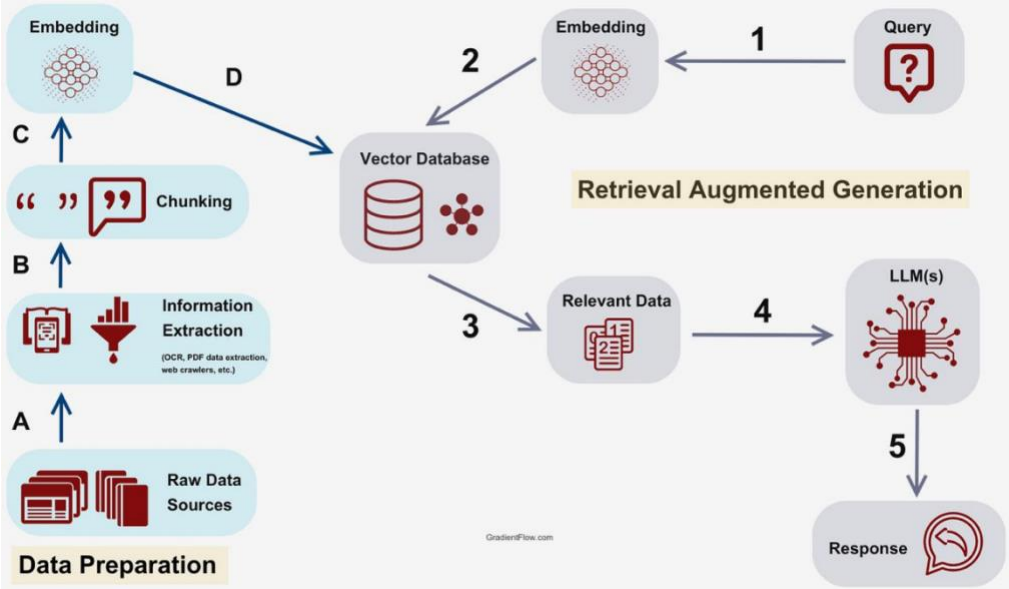
Одан әрі промт жасау (prompt engineering), яғни жасанды интеллектің тілді түсіну жүйелеріне, әсіресе үлкен тілдік модельдерге (LLM) дұрыс сұрақ қою арқылы керекті нәтижеге жету. Қарапайым тілмен айтқанда, бұл - модельге не

сұрап тұрғаныңды жақсылап түсіндіру. RAG жүйесінде промпт инжинирингі екі жағдайда керек. Бірінші - іздеу кезеңі: модельге дұрыс ақпарат табу үшін ақылды сұрақтар қойылады. Сұрақ пайдаланушының сұранысына сай етіліп, дерекқордағы мәліметтермен дәлме-дәл сәйкес келуі керек. Екінші – жауап беру кезеңінде: табылған мәліметтерге сүйене отырып, сұрақ жаңаша құрастырылады. Бұл кезде промптке сұрақтың түсініктемесі, табылған мәліметтер және нақты сұрақ кіреді. Мысалы, «мына мәліметке қарап, мына сұраққа жауап бер...» деген сияқты.

Моделмен жұмыс істеу үшін арнайы prompt шаблоны дайындалды:

*Қазақ қолжазбалары корпусы негізінде сұраққа жауап беріңіз:*
*Контекст: {retrieved_documents}*
*Сұрау: {user_question}*
*Нұсқаулар:*
*- Тек берілген контекст ақпаратын пайдаланыңыз*
*- Қазақ тілінде жауап беріңіз*
*- Дереккөздерді көрсетіңіз*
*- Ақпарат жеткіліксіз болса, мұны айтыңыз*

Құрылған құжаттардан іздеу жүйесінің схемасы 3.6 суретте көрсетілген.




Сурет 3.6 – Құжаттардан мәліметті іздеу жүйесі

# ҚОРЫТЫНДЫ

Бұл диссертациялық жұмыста қазақ тіліндегі қолжазба мәтіндерді тану жүйелерінің өзекті мәселелері жан-жақты қарастырылды. Қолжазба мәтіндердің әртүрлілігі, жазу стилінің бірегейлігі, қазақ графикасының морфологиялық және синтаксистік ерекшеліктері сияқты қиындықтар ескеріліп, осы бағыттағы қазіргі заманғы шешімдер мен технологиялар қолданылды. Жүргізілген зерттеу жұмыстары нәтижесінде құрылған алгоритмдер мен нейрондық желі модельдері қазақ тіліндегі қолжазбаларды жоғары дәлдікпен тануға мүмкіндік беріп, осы саладағы ғылыми және практикалық мәселелерді шешуге үлес қосты.

Диссертациялық жұмысты орындау барысында келесідей нәтижелерге қол жеткізілді:

- Қазақ тіліндегі қолжазба мәтіндерді қамтитын деректер жиынтығы жасақталды. Бұл жиынтықта түрлі жазу стильдеріндегі жолдар мен сөздер қамтылып, жүйелі түрде алдын ала өңдеуден өткізілді;
- Мәтінді жолдарға және сөздерге дәл және тиімді түрде бөлетін сегментация модельдері әзірленді. Бұл модельдер DeepLabV3 негізіндегі сегментациялық нейрондық желілер арқылы жүзеге асырылды;
- Деректердің көлемін арттыру және модельдің тану дәлдігін арттыру мақсатында Generative Adversarial Network (GAN) негізіндегі жасанды деректер генерациялау әдістері қолданылды. Бұл әдіс сирек кездесетін жазу түрлерін үлкейтуге мүмкіндік берді;
- Қолжазба сөздерді тануға арналған гибридті CNN және Transformer архитектурасына негізделген тану моделі жасалды. Бұл архитектура мәтіннің визуалды ерекшеліктерін тиімді үйреніп, күрделі тілдік тәуелділіктерді ескеруге мүмкіндік береді;
- Қолжазбаны тануға арналған жалпы жүйенің құрылымдық-логикалық және ақпараттық моделі жасалды. Бұл модель жүйенің әрбір компонентінің өзара байланысын анықтап, тану үдерісін формализациялауға мүмкіндік берді;
- Құжаттардан мәліметтерді іздеу және нақты жауаптар ұсыну үшін Retrieval-Augmented Generation (RAG) архитектурасы енгізілді. Бұл тәсіл танылған мәтін негізінде нақты әрі релевантты ақпаратты іздеуге және генерациялауға мүмкіндік берді.

Жалпы, диссертациялық жұмыста ұсынылған модельдер мен әдістер қазақ тіліндегі қолжазба мәтіндерді өңдеудің заманауи тәсілдерін пайдаланып, олардың дәлдігін арттыруға бағытталды. Құрылған жүйелерді болашақта мемлекеттік архивтерде, білім беру саласында, тарихи құжаттарды цифрландыруда, сондай-ақ цифрлық басқару мен электронды үкімет жүйелерінде кеңінен қолдануға болады.

## ПАЙДАЛАНҒАН ӘДЕБИЕТТЕР ТІЗІМІ


1. Villena Toro J, Wiberg A and Tarkian M. Optical character recognition on engineering drawings to achieve automation in production quality control // Front. Manuf. Technol. – 3:1154132. – 2023. https://doi.org/10.3389/fmtec.2023.1154132

2. P. Kunekar, K. Vayadande, O. Kulkarni, K. Ingale, R. Kadam and S. Inamdar. OCR based Cheque Validation using Image Processing // – 2023. 5th Biennial International Conference on Nascent Technologies in Engineering (ICNTE) – 2023. – pp. 1-5. https://doi.org/10.1109/ICNTE56631.2023.10146687.

3. Қазақстан Республикасының Мұрағаты http://archive.president.kz/ru/

4. Интернет сілтеме Цифрландыру мемлекетік архив.

5. Marti, U.-V & Bunke, H.The IAM-database: An English sentence database for offline handwriting recognition // International Journal on Document Analysis and Recognition – 5. – pp. 39-46. https://doi.org/10.1007/s100320200071 .

6. E. Grosicki, M. Carré, J.M. Brodin and E. Geoffrois. Results of the RIMES Evaluation Campaign for Handwritten Mail Processing // – 2009. 10th International Conference on Document Analysis and Recognition . – 2009. – pp. 941-945. https://doi.org/10.1109/ICDAR.2009.224 .

7. F. Kleber, S. Fiel, M. Diem and R. Sablatnig. CVL-DataBase: An Off-Line Database for Writer Retrieval, Writer Identification and Word Spotting // – 2013 12th International Conference on Document Analysis and Recognition . – 2013. – pp. 560-564. https://doi.org/10.1109/ICDAR.2013.117 .

8. O. Vovchuk, M. Kyrychenko, Y. Kazan, Recognition of Handwritten Cyrillic Letters // Passive Cutaneous Anaphylaxis. – 2019.

9. Nurseitov D., Bostanbekov K., Kurmankhojayev. Handwritten Kazakh and Russian (HKR) database for text recognition // Computer Vision and Pattern Recognition. – 2020.

10. Гороыжин А. Н., Обработка и распознавание рукописного текста в системах электронного документооборота // автореферат . – 2008.

11.Интернет сілтеме https://en.wikipedia.org/wiki/Computer_Vision_Annotation_Tool

12.Интернет сілтеме https://viso.ai/computer-vision/labelimg-for-image-annotation/.

13. Russell, B.C., Torralba, A., Murphy, K.P. *et al.* LabelMe: A Database and Web-Based Tool for Image Annotation // Int J Comput Vis – **77.** – pp. 157–173. – 2008. https://doi.org/10.1007/s11263-007-0090-8 .

14. Russ, J. The Image Processing Handbook, Fifth Edition // CRC Press, Boca Raton .– 2007.

15. Tarsha-Kurdi F., Landes T., Grussenmeyer P. Hough-Transform and Extended RANSAC Algorithms for Automatic Detection of 3D Building Roof Planes from Lidar Data // ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007 – vol. XXXVI – pp. 407– 412.

16. O'Gorman, L., Kasturi, R., Govindaraju, V., & B. U. Pal, S. Document image analysis: A primer // In Document Image Analysis – pp. 1-19 – Springer, London.

17. Gonzalez, R. C., Woods, R. E., & Eddins, S. L. Digital Image Processing Using MATLAB // Prentice Hall. – 2004.

18. Tanner, S. Deciding whether Optical Character Recognition is Feasible // King's Digital Consultancy Services, http://www.odl.ox.ac.uk/papers/OCRFeasibility_final.pdf.

19. Otsu, N. A Threshold Selection Method from Gray-Level Histograms // IEEE Transactions on Systems, Man, and Cybernetics – 9(1) – pp. 62 - 66.

20. A. Asi, R. Cohen, K. Kedem, and J. Elsana. Simplifying the reading of historical manuscripts // in Proceedings of the 13th International Conference of Document Analysis and Recognition – pp. 826–830. – 2015.

21. M. Agrawal and D. Doermann. Voronoi++: A dynamic page segmentation approach based on voronoi and docstrum features // in Proceedings of the 10th International Conference on Document Analysis and Recognition – pp. 1011–1015. – 2009.

22. M. Baechler and R. Ingold. Multi resolution layout analysis of medieval manuscripts using dynamic mlp // in Proceedings of the 11th International Conference on Document Analysis and Recognition – pp. 1185–1189. – 2011.

23. A. Garz, R. Sablatnig, and M. Diem. Layout analysis for historical manuscripts using sift features // in Proceedings of the 11th International Conference on Document Analysis and Recognition – pp. 508–512. – 2011.

24. S. S. Bukhari, T. M. Breuel, A. Asi, and J. El-Sana. Layout analysis for arabic historical document images using machine learning // in Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition – pp. 639–644. – 2012.

25. G. Nagy, M. Viswanathan, and S. Seth. A prototype document image analysis system for technical journals // in Proceedings of the IEEE – vol. 25. – pp. 10-22. – 1992.

26. S. Uttama, J. M. Ogier, and P. Loonis. Top-down segmentation of ancient graphical drop caps: lettrines // in Proceedings of the International Workshop on Graphics Recognition – pp. 87–96. – 2005.

27. N. Ouwayed and A. Belad. Multi-oriented text line extraction from handwritten arabic documents // in Proceedings of the Eighth Iapr International Workshop on Document Analysis Systems – pp. 339–346. – 2008.

28. N. Journet, J. Y. Ramel, R. Mullot, and V. Eglin. Document image characterization using a multiresolution analysis of the texture: application to old documents // in Proceedings of the International Journal on Document Analysis and Recognition – vol. 11. – pp. 9–18. – 2008.

29. Mao, S., Rosenfeld, A. Document structure analysis algorithms: A literature survey // In Proc. SPIE, Document Recognition and Retrieval X – vol. 5010 – pp. 197-207 – International Society for Optics and Photonics.

30. Saba, T., Rehman, A., Hussain. M. A Comprehensive Review of Handwritten Word Segmentation // .– 2011

31. Droby, A., Shkolnisky, Y., & Elad, M. Unsupervised Line Segmentation of Historical Documents via Feature Clustering // arXiv preprint. https://arxiv.org/abs/2101.07487.

32. Plaksyvyi, O., Tyshchenko, V., & Borysov, D. A Comparative Analysis of Image Segmentation Using Classical and Deep Learning Approach // ResearchGate. https://www.researchgate.net/publication/376127494. – 2023.

33. Long, J., Shelhamer, E., & Darrell, T. Fully Convolutional Networks for Semantic Segmentation // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 3431–3440. https://arxiv.org/abs/1411.4038 . – 2015.

34. Ronneberger, O., Fischer, P., & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation // In Medical Image Computing and Computer-Assisted Intervention (MICCAI) – Vol. 9351 – pp. 234–241. – 2015. https://arxiv.org/abs/1505.04597.

35. Chen L. C., Papandreou, G., Kokkinos I., Murphy K., Yuille A. L. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs // arXiv preprint. https://arxiv.org/abs/1706.05587.

36. A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition // In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. – 2014.

37. Michael J., Puigcerver J., Labahn, R. Evaluating the performance of a CNN-LSTM hybrid model for handwritten text recognition // Pattern Recognition Letters – Vol. 129 – pp.205–211 .– 2020.

38. Anuj Dutt, Aashi Dutt, Handwritten Digit Recognition Using Deep Learning, //International Journal of Advanced Research in Computer Engineering & Technology (IJARCET). – 2017. – vol. 6 – issue 7. – pp. 990-997.

39. M. Kayed, A. Anter and H. Mohamed. Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture // 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE), Aswan, Egypt – 2020. – pp. 238-243. doi: 10.1109/ITCE48509. – 2020.9047776.

40. Chenghao Cai. Fast Learning of Deep Neural Networks via Singular Value Decomposition // PRICAI 2014 – LNAI 8862. – pp. 820–826. – 2014.

41. Gualtiero Piccinini. The First Computational Theory of Mind and Brain: A Close Look at Mcculloch and Pitts's “Logical Calculus of Ideas Immanent in Nervous Activity” // Synthese. – 141(2). – 2014.

42. Minsky M., Papert. Review of 'Perceptrons: An Introduction to Computational Geometry // IEEE Xplore, IEEE Transactions on Information Theory – 1969. – 15(6) – pp. 738 – 739. DOI:10.1109/TIT. – 1969.1054388

43. Dr. Paul Werbos: Artificial Neural Networks IEEE Neural Network. – 2011.

44. Minsky ML. Theory of neural-analog reinforcement systems and its application to the brain model problem doctoral dissertation (PhD thesis) // Princeton, NJ: Princeton University. – 2016. – OCLC 3020680

45. David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors // Nature. – 1986.

46. Kurt Hornik. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function // Neural Networks, Multidisciplinary. – 1991.

47. Cortes C., Vapnik V. Support-vector networks // Machine Learning. – *20*(3). – pp.273–297. – 1995.

48. Martin Anthony and Peter L Bartlett. Neural network learning: Theoretical foundations // Cambridge university press. – 2009.

49. Schölkopf B., Smola A. J. Learning with Kernels // Support Vector Machines, Regularization, Optimization, and Beyond.MIT Press. – 2002.

50. LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. Gradient-based learning applied to document recognition. // Proceedings of the IEEE. – 86(11). – pp. 2278–2324. – 1998.

51. Dewi Kusumawati, Amil Ahmad Ilham, Andani Achmad, Ingrid Nurtanio. Vgg-16 And Vgg-19 Architecture Models In Lie Detection Using Image Processing. // 2022 6th International Conference on Information Technology, Information Systems and Electrical Engineering (ICITISEE). – 2022. IEEE.

52. C. Lamprecht, P. Bekhrad, H. Ivanov and E. Leitgeb, "Modelling the Refractive Index Structure Parameter. // A ResNet Approach," 2020 International Conference on Broadband Communications for Next Generation Networks and Multimedia Applications (CoBCom), Graz, Austria. – 2020. – pp. 1-4.

53. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. // In Advances in Neural Information Processing Systems (NIPS). – pp. 1097–1105. – 2012.

54. Hitawala, Saifuddin. Evaluating ResNeXt Model Architecture for Image Classification. – 2018.

55. M Tan, Q Le, Efficientnet.// Rethinking model scaling for convolutional neural networks International conference on machine learning. – 2019.

56. Ankan Dash, Junyi Ye, and Guiling Wang. " A review of Generative Adversarial Networks (GANs) and its applications in a wide variety of disciplines // From Medical to Remote Sensing". – 2021.

57. Ngoc-TrungTran, Tuan-Anh Bui, Ngai-Man Cheung. Dist-GAN: An Improved GAN using Distance Constraints // Book chapter. ECCV – 2018.

58. Ghojogh, B., Karray, F., Crowley, M. Theoretical Insights into the Use of Structural Similarity Index in Generative Models and Inferential Autoencoders // Image Analysis and Recognition. ICIAR – 2020.

59. Jonathon Shlens. Notes on Kullback-Leibler Divergence and Likelihood // Computer Science, ArXiv. – 2014. – pp. 1-4.

60. P.Krishnan, R.Kovvuri, G. Pang, B. Vassilev, T. Hassnerhttps. TextStyleBrush: Transfer of Text Aesthetics from a Single Example. // Computer Science,Computer Vision and Pattern Recognition. – 2021. – pp. 1-18.

61. Xun Huang, Serge Belongie. Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization // Computer Science> Computer Vision and Pattern Recognition, – 2017. pp. 1-11.

62. Tero Karras, Samuli Laine, Timo Aila. A Style-Based Generator Architecture for Generative Adversarial Networks // Computer Science> Neural and Evolutionary Computing. – 2019. pp. 1-112.

63. Sharon Fogel, Hadar Averbuch-Elor, Sarel Cohen, Shai Mazor, Roee Litman. ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation // Computer Science> Computer Vision and Pattern Recognition. – 2021, pp. 1-12.

64. Andrew Brock, Jeff Donahue, Karen Simonyan. Large Scale GAN Training for High Fidelity Natural Image Synthesis // Computer Science > Machine Learning. – 2019, pp. 1-35.

65. Mienye, I.D.; Swart, T.G.; Obaido, G. Recurrent Neural Networks // A Comprehensive Review of Architectures, Variants, and Applications. Information . – 2024, pp. *15*, 517. https://doi.org/10.3390/info15090517.

66. Jozefowicz, R., Zaremba, W., & Sutskever, I. An empirical exploration of recurrent network architectures // In Proceedings of the 32nd International Conference on Machine Learning (ICML) – pp.2342–2350. – 2015.

67. Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation // arXiv:1406.1078 https://arxiv.org/abs/1406.1078

68. Zhou C., Sun C., Liu Z., Lau F. A C-LSTM neural network for text classification // Neural Networks – V *88*. – pp. 105–113 – 2014. https://doi.org/10.1016/j.neunet.2016.04.006

69. Fischer, A., Keller, A., Frinken, V., Bunke, H. (2012). Lexicon-free handwritten word spotting using character HMMs. // Pattern Recognition Letters – 33(7). – pp. 934–942.

70. Graves, A., Schmidhuber, J. Offline handwriting recognition with multidimensional recurrent neural networks. // Advances in Neural Information Processing Systems. – Vol. 21. – pp. 545–552.

71. Sutskever I., Vinyals, O., Le Q. V. Sequence to sequence learning with neural networks // Advances in Neural Information Processing Systems. – Vol. *27*. – pp.3104–3112. – 2014.

72. Bahdanau D., Cho K., Bengio Y. Neural machine translation by jointly learning to align and translate. // In Proceedings of the 3rd International Conference on Learning Representations (ICLR) - 2015.

73. Serrano N., Vigueras M. Handwritten text recognition with encoder-decoder networks. // In 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). – 2019. – Vol. 2. – pp. 37–42.

74. Mager M., Scheidl T., Fink G. A. A subword-based transformer approach for handwritten text recognition. // In Proceedings of the International Conference on Document Analysis and Recognition (ICDAR) – pp. 179–194 – 2021.

75. Morris A. C., Maier V., Green P. From WER and RIL to MER and WIL: Improved evaluation measures for connected speech recognition. // Proceedings of the

8th International Conference on Spoken Language Processing (INTERSPEECH). – 2004. – pp.2765–2768.

76. NVIDIA. NVIDIA Triton Inference Server: A tool to deploy, run, and scale AI models. NVIDIA Developer Documentation. https://developer.nvidia.com/nvidia-triton-inference-server

77. Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., Riedel S. Retrieval-augmented generation for knowledge-intensive NLP tasks. // In Advances in Neural Information Processing Systems. – 2020. – vol. 33. – pp. 9459–9474.

78. Reimers N., Gurevych I. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. // In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP). – 2019. – pp. 3982–3992.

79. Yeleussinov A., Amirgaliyev Y., Cherikbayeva L. Improving OCR Accuracy for Kazakh Handwriting Recognition Using GAN Models. // Applied Science. – 2023. – vol.13. – pp.5677. https://doi.org/10.3390/app13095677

80. Amirgaliyev B., Yeleussinov A., Taizo M. Kazakh handwritten recognition. // Journal of Theoretical and Applied Information Technologies. – 2020. – vol.98(14). – pp. 2744–2754.

81. Yeleussinov A., Islamgozhayev T., Kozhirbayev Z.. Creation and Annotation of a Handwritten Text Database for the Kazakh Language: Methodologies and Preliminary Results. // 5th International Conference on Electrical, Communication and Computer Engineering ICECCE. – 2024.

82. Амиргалиев Е.Н., Миачи Т., Елеусинов А.И. Методы и алгоритмы формирования базы и обработки рукописных текстов на казахском языке. // Вестник КазНИТУ – 2020г. – №138. – сс. 653-660.

83. Елеусинов А.И., Бурибаев Ж. Сегментация рукописных символов с исправлением наклона текста // Конференция "V-международная конференция «Информатика и прикладная математика»"– 2020.

ҚОСЫМША А – Авторлық куәлік



ҚАЗАҚСТАН РЕСПУБЛИКАСЫ

РЕСПУБЛИКА КАЗАХСТАН

АВТОРЛЫҚ ҚҰҚЫҚПЕН ҚОРҒАЛАТЫН ОБЪЕКТІЛЕРГЕ ҚҰҚЫҚТАРДЫҢ МЕМЛЕКЕТТІК ТІЗІЛІМГЕ МӘЛІМЕТТЕРДІ ЕНГІЗУ ТУРАЛЫ

КУӘЛІК

2019 жылғы «11» қараша № 6358

Автордың (лардың) жөні, аты, әкесінің аты (егер ол жеке басын куәландыратын құжатта көрсетілсе):
**АМИРГАЛИЕВ ЕДИЛХАН НЕСИПХАНОВИЧ, ЕЛЕУСИНОВ АРМАН ИНАБАТОВИЧ**

Авторлық құқық объектісі: **ЭЕМ-ге арналған бағдарлама**

Объектінің атауы: **"Kazakh HandWritten OCR"**

Объектіні жасаған күні: **01.10.2019**

Құжат түпнұсқалығын http://www.kazpatent.kz/ru сайтының "Авторлық құқық" бөлімінде тексеруге болады https://copyright.kazpatent.kz

Подлинность документа возможно проверить на сайте kazpatent.kz в разделе «Авторское право» https://copyright.kazpatent.kz

Подписано ЭЦП

Оспанов Е. К.

# ҚОСЫМША Б – Қолжазба мәтінді тану алгоритмінің бағдарлама листингі

```
import cv2
import torch
import numpy as np
from utils.segmenter import load_deeplabv3, segment_lines_and_words
from utils.recognizer import load_recognizer_model, recognize_words

DEVICE = 'cuda' if torch.cuda.is_available() else 'cpu'

# Модельдерді жүктеу
seg_model = load_deeplabv3("deeplabv3_model.pth", device=DEVICE)
rec_model = load_recognizer_model("cnn_transformer_model.pth", device=DEVICE)

# Кіріс бейне
img = cv2.imread("kaz_image.jpg", cv2.IMREAD_GRAYSCALE)

# Сегменттеу
lines_with_words = segment_lines_and_words(seg_model, img, device=DEVICE)

# Тану
for i, words in enumerate(lines_with_words):
    print(f"Line {i+1}:")
    for word_img in words:
        text = recognize_words(rec_model, word_img, device=DEVICE)
        print(text, end=' ')
    print("\n")

import torch
import torchvision.transforms as T
import numpy as np
import cv2

def load_deeplabv3(path, device='cpu'):
    from torchvision.models.segmentation import deeplabv3_resnet50
    model = deeplabv3_resnet50(pretrained=False, num_classes=3)
    model.load_state_dict(torch.load(path, map_location=device))
    model.to(device).eval()
    return model

def segment_lines_and_words(model, img, device='cpu'):
    transform = T.Compose([
        T.ToPILImage(),
        T.Resize((512, 512)),
        T.ToTensor(),
        T.Normalize(mean=[0.5], std=[0.5])
    ])
    input_tensor = transform(img).unsqueeze(0).to(device)
    with torch.no_grad():
        output = model(input_tensor)['out']
    mask = torch.argmax(output.squeeze(), dim=0).cpu().numpy()

    # 1 = line, 2 = word
    lines = extract_regions(mask, class_id=1)
    all_words = []
    for line_box in lines:
        line_crop = crop_from_box(img, line_box)
        words = extract_regions(mask, class_id=2, in_box=line_box)
```

```
        word_crops = [crop_from_box(line_crop, box) for box in words]
        all_words.append(word_crops)
    return all_words

def extract_regions(mask, class_id=1, in_box=None):
    if in_box:
        x, y, w, h = in_box
        mask = mask[y:y+h, x:x+w]
    contours, _ = cv2.findContours((mask == class_id).astype(np.uint8), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    boxes = [cv2.boundingRect(cnt) for cnt in contours]
    return sorted(boxes, key=lambda b: b[0])

def crop_from_box(img, box):
    x, y, w, h = box
    return img[y:y+h, x:x+w]

import torch
import torchvision.transforms as T
from utils.model_defs import CNNTransformerOCR

def load_recognizer_model(path, device='cpu'):
    model = CNNTransformerOCR()
    model.load_state_dict(torch.load(path, map_location=device))
    model.to(device).eval()
    return model

def recognize_words(model, word_img, device='cpu'):
    transform = T.Compose([
        T.ToPILImage(),
        T.Resize((32, 128)),
        T.ToTensor(),
        T.Normalize([0.5], [0.5])
    ])
    tensor = transform(word_img).unsqueeze(0).to(device)

    with torch.no_grad():
        logits = model(tensor)  # [batch, seq_len, vocab_size]
        preds = torch.argmax(logits, dim=-1).squeeze(0)
    return decode_prediction(preds)

kazakh_charset = [
    'а', 'ә', 'б', 'в', 'г', 'ғ', 'д', 'е', 'ё', 'ж', 'з', 'и', 'й',
    'к', 'қ', 'л', 'м', 'н', 'ң', 'о', 'ө', 'п', 'р', 'с', 'т', 'у',
    'ұ', 'ү', 'ф', 'х', 'һ', 'ц', 'ч', 'ш', 'щ', 'ы', 'і', 'э', 'ю', 'я',
    'А', 'Ә', 'Б', 'В', 'Г', 'Ғ', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й',
    'К', 'Қ', 'Л', 'М', 'Н', 'Ң', 'О', 'Ө', 'П', 'Р', 'С', 'Т', 'У',
    'Ұ', 'Ү', 'Ф', 'Х', 'Һ', 'Ц', 'Ч', 'Ш', 'Щ', 'Ы', 'І', 'Э', 'Ю', 'Я'
]

def decode_prediction(preds):
    return ''.join([kazakh_charset[i] for i in preds if i < len(kazakh_charset)])
```



inprotected.com