ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ОҚУ-АҒАРТУ МИНИСТРЛІГІ

АЛМАТЫ ҚАЛАСЫНЫҢ БІЛІМ БАСҚАРМАСЫ

АЛМАТЫ ҚАЛАСЫНЫҢ ИННОВАЦИЯЛЫҚ ТЕХНИКАЛЫҚ КОЛЛЕДЖІ

**ДИПЛОМДЫҚ ЖОБА**

Жасанды интеллект чатын әзірлеу

ТҮСІНІК ХАТ

|  |  |
| --- | --- |
| Директордың ОІ жөніндегі орынбасары  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ К.Шаймуханбетова  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. | Рецензент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.Қайранбаев  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. |
| ПЦК төрайымы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Р.Нурлибаева  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. | Жетекші  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.Бейсенов  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. |
| Кеңесшілер: |  |
| Экономика бойынша  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ж.Тилемисова  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. | Білім алушы  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.Абас  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. |
| Еңбекті қорғау бойынша  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Э.Рахманова  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. | Нормобақылау  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ұ.Жеңіс  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 ж. |

Алматы, 2024

МАЗМҰНЫ

[КІРІСПЕ 4](#_Toc167800340)

[1 АРНАЙЫ БӨЛІМ 5](#_Toc167800341)

[1.1 Жасанды интеллект саласы 5](#_Toc167800342)

[1.2 Машиналық оқыту өзін-өзі жетілдіру алгоритмі 7](#_Toc167800343)

[1.3 Нейрондық желілер жасанды интеллект миы 9](#_Toc167800344)

[1.4 Терең оқыту технологиясы 11](#_Toc167800345)

[1.5 Табиғи тілді өңдеу, LLM мен LLaMA 13](#_Toc167800346)

[1.6 Ollama платформасы 15](#_Toc167800347)

[1.7 Жасанды интеллект чатының диалог жүйесі 21](#_Toc167800348)

[1.8 Жасанды интеллект чатының веб-интерфейсі 26](#_Toc167800349)

[2 ЭКОНОМИКАЛЫҚ БӨЛІМ 31](#_Toc167800350)

[2.1 Жобаның экономикалық зерттеуі 31](#_Toc167800351)

[2.2 Жұмысшы шығындары мен жұмысшылардың ЖТС-тары 31](#_Toc167800352)

[2.3 Материалдар мен жасақтама шығындары 33](#_Toc167800353)

[2.4 Электр энергиясының шығындары 33](#_Toc167800354)

[2.5 Амортизация шығындары 34](#_Toc167800355)

[2.6 Жасанды интеллект чатына құруға кеткен жалпы шығындар 34](#_Toc167800356)

[2.7 Табыс пен рентабельдікті анықтау 35](#_Toc167800357)

[2.8 Жобаға экономикалық анализ жасау 35](#_Toc167800358)

[3 ЕҢБЕКТІ ҚОРҒАУ БӨЛІМІ 36](#_Toc167800359)

[3.1 Еңбек қауіпсіздігі 36](#_Toc167800360)

[3.1.1 Еңбек режимі 36](#_Toc167800361)

[3.2 Өндіріс санитария және еңбек гигиенасы 37](#_Toc167800362)

[3.2.1 Микроклимат 38](#_Toc167800363)

[3.2.2 Жарықтандыру 39](#_Toc167800364)

[3.3 Қауіпсіздік техникасы 42](#_Toc167800365)

[3.3.1 Электр қауіпсіздігі 43](#_Toc167800366)

[3.4 Өрт қауіпсіздігі 44](#_Toc167800367)

[ҚОРЫТЫНДЫ 46](#_Toc167800368)

[ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ 47](#_Toc167800369)

[А ҚОСЫМШАСЫ 48](#_Toc167800370)

КІРІСПЕ

**Жұмыстың өзектілігі.** Қазіргі уақытта жасанды интеллект негізіндегі чаттарды жасау өзекті болып табылады. Олар түрлі салаларда қолданылып, пайдаланушылармен өзара әрекеттесу, ақпаратты жылдам ұсыну және сұрақтар-

ға нақты жауап беру сияқты көптеген функцияларды орындайды. Чаттар интерактивті көмекші ретінде қызмет атқарып, тұтынушыларды қолдау жүйесін-

де, білім беру платформаларында және коммерциялық қызметтерде кеңінен пайдаланылады. Мұндай чаттар тиімді жұмыс істеуі үшін жасанды интеллект технологияларын терең түсіну және қолдану қажет.

Жасанды интеллект негізіндегі чаттардың өзектілігі олардың қазіргі қоғамдағы кең таралуында және үлкен сұранысқа ие болуында. Мұндай жүйелердің дамуы техникалық талаптармен шектелгенімен, бұл шектеулер бола-

шақта жаңа технологиялардың пайда болуымен және олардың жетілдірілуімен шешімін табады, бұл чаттарды дамытуда үлкен перспективалар ашады.

**Жұмыстың мақсаты** – жасанды интеллект негізіндегі чат құрастыру арқылы пайдаланушыларға ыңғайлы, тиімді және интеллектуалды көмекші жасау. Жұмыста қойылған мақсатқа жету үшін пайдаланушыға ыңғайлы веб-интерфейс әзірлеу міндеті шешілді. Сонымен қатар, жасанды интеллект алгоритмдерін қолдану арқылы чатты оқыту және дамыту, пайдаланушының сұраныстарына сәйкес жауаптар беру үшін чатты оңтайландыру, диалог жүйесі-

нің тиімділігін арттыру бағытында жұмыстар атқарылды.

**Зерттеу нысаны** – жасанды интеллект негізіндегі чат құрастыру және оны пайдаланушыларға арналған веб-интерфейсте іске қосу.

**Зерттеу тақырыбы** – жасанды интеллект негізіндегі чаттарды жобалау және дамыту.

**Жұмыстың практикалық маңыздылығы.** Жасанды интелект негізінде-

гі чаттарды пайдалану арқылы пайдаланушылардың сұраныстарына жылдам және нақты жауап беру, жұмыс тиімділігін арттыру және қызмет көрсету сапасын жақсарту болып табылады. Мұндай чаттар білім беру, коммерция, денсаулық сақтау және басқа да көптеген салаларда қолданыс табады.

**Дипломдық жоба құрылымы мен көлемі.** Дипломдық жоба кіріспеден, арнайы бөлімнен, экономикалық бөлімнен, еңбекті қорғау мен қорытындыдан, пайдаланылған әдебиеттер тізімінен, қосымшалардан тұрады. Жұмыстың жалпы көлемі 60 бетті, 7 кестені, 22 суретті, қосымшаны құрайды.

Бұл жоба кез-келген пайдаланушыға ұсынылып, өз бетінше ақпарат алуда және әр түрлі сұрақтарға жауап алуда жақсы көмекші құрал болып табылады. Сонымен қатар, чаттар арқылы пайдаланушылар өз сұраныстарын жылдам әрі тиімді түрде шеше алады. Қазіргі уақытта жасанды интеллект чаттарына деген қызығушылық мультимедиялық технологиялар мен байланыс құралдарының, интернет желісінің дамуына байланысты артып келеді.

1 АРНАЙЫ БӨЛІМ

1.1 Жасанды интеллект саласы

Жасанды интеллект (ағыл. Artificial Intelligence) – адамның интеллектуал-

ды қабілетін қажет ететін тапсырмаларды орындауға қабілетті жүйелер мен бағдарламаларды құруға арналған информатика саласы. Жасанды интеллект адамға ұқсас ойлауға, әрекет етуге және шешім қабылдауға қабілетті немесе тіпті кейбір салаларда адамның қабілеттерінен асып түсетін машиналар жасауға тырысады. [1]

Жасанды интеллекттің негізгі мақсаттарына мыналар жатады:

- Күрделі есептерді шеше алатын жүйелерді жасау: жасанды интеллект деректердің үлкен көлемін талдай алатын және олардағы үлгілерді таба алатын немесе күрделі есептерді шеше алатын бағдарламалар мен алгоритмдерді жасауға тырысады.

- Бейімделу және білім беру жүйелерін құру: жасанды интеллект сонымен қатар тәжірибе негізінде оқытылатын, жаңа жағдайларға бейімделетін және қабілеттерін біртіндеп жетілдіре алатын жүйелерді құруға бағытталған.

- Адамның интеллектуалды процестерін модельдеу: жасанды интеллект адамның ойлауы мен мінез-құлқына еліктейтін, табиғи тілді түсінетін, бейнелер-

ді танитын, шешім қабылдайтын және модельдер жасауға тырысады.

Жасанды интеллект терминін алғаш рет 1956 жылы Джон МакКарти Дартмут университетінде өткен конференцияда айтқан. Бұл терминнің енгізілуі жасанды интеллект саласындағы зерттеулерді дамытудың бастапқы нүктесі болды. Негізгі идея адамның интеллектін қажет ететін тапсырмаларды орындай алатын машиналар жасау болды.

Мұндай жүйені құру идеясы 1935 жылы Алан Тюрингтің еңбектерінің арқасында пайда болды. Ол өз жазбаларында шексіз жад пен сканерден тұратын абстрактілі есептеу машинасын ұсынды, ол оны алға-артқа жылжытады.

1950 жылы ол жасанды жүйе тұжырымдамасын ұсынды, онда оны адамнан ерекшеленбейтін сөйлесуге қабілетті жүйелер ретінде анықтады. Тьюринг машиналық интеллекттің даму деңгейін бағалау үшін Тьюринг тесті деп аталатын эмпирикалық тест жасады. Бұл тест жасанды жүйенің байланыс қабілеті қаншалықты сәтті жүргізе алатындығын және оны адам деп қателесуге болатындығын анықтауға мүмкіндік береді.

Жасанды интеллекттің алғашқы сәтті бағдарламасы 1951 жылы Кристофер Стрейчи жасаған. Ол адаммен шашки ойнап, көрермендерді болжау қабілетімен таң қалдырды. Осыған байланысты 1953 жылы Тьюринг шахматы бағдарламалау туралы мақала жариялады.

1965 жылы Джозеф Вейзенбаум адаммен диалогты имитациялайтын және жасанды интеллекттің даму тарихындағы маңызды кезеңдерге айналған Eliza бағдарламасын құрды. Элиза сияқты жобалар осы саладағы зерттеулерді қаржы-

ландыруды жалғастырған DARPA сияқты агенттіктерді қызықтырды.

1980 жылдары жаңа алгоритмдер мен қаржыландырудың арқасында терең оқыту мен сараптамалық жүйелерді қоса алғанда, даму қайта жанданды. Жапониядағы бесінші буын компьютері жас зерттеушілердің қызығушылығын арттырды. Алайда өршіл мақсаттардың көпшілігіне қол жеткізілмеді, бірақ жоба жас инженерлер мен ғалымдарды шабыттандырды.

1997 жылы жасанды интеллект өзінің эволюциясындағы ең маңызды оқиғалардың бірі ретінде тарихта қалған оқиғаның арқасында жаңа дамуға ие болды. IBM Deep Blue суперкомпьютері шахматтан әлем чемпионы Гарри Каспаровты жеңе алды. Бұл тарихи сәт компьютерлер бұрын тек адам болып саналатын интеллектуалды тапсырмаларда керемет нәтижелерге қол жеткізе алатынын көрсетті (Сурет-1.1).



Сурет-1.1 Гарри Каспаров пен Deep Blue партиясы

Содан бері жасанды интеллекттің дамуы белсенді түрде жалғасуда. Есептеу ресурстарының қолжетімділігінің артуымен және деректер көлемінің кеңеюімен жасанды нейрондық желілер қарқынды дамып келеді. Машиналық оқыту заманауи алгоритмдері компьютерлерге ақпаратты талдап қана қоймай, ондағы заңдылықтарды анықтауға, болжам жасауға және оқыту деректер негізін-

де шешім қабылдауға мүмкіндік береді.

Жасанды интеллект машиналық оқыту, нейрондық желілер, терең оқыту, табиғи тілді өңдеу және т.б. сияқты әртүрлі тәсілдер мен әдістерді қамтитын көп қырлы сала. Ол көптеген салаларда, соның ішінде медицина, қаржы, автомобиль өнеркәсібі, медиа, ойын индустриясы және адам қызметінің басқа салаларында қолданылады. [2]

Жоба тақырыбын түсіну үшін жасанды интеллекттің тұжырымдамалары мен мақсаттары туралы жалпы түсінікке ие болу маңызды, өйткені олар жасанды интеллект чаты сияқты нақты қосымшалар мен жүйелерді әзірлеуге және жүзеге асыруға негіз болады.

1.2 Машиналық оқыту өзін-өзі жетілдіру алгоритмі

Машиналық оқыту (ағыл. Machine Learning) – жасанды интеллект әдістері-

нің класы, оның ерекшелігі мәселені тікелей шешу емес, көптеген ұқсас есептер-

дің шешімдерін қолдану арқылы оқыту. Мұндай әдістерді құру үшін математика-

лық статистика, сандық әдістер, Математикалық талдау, оңтайландыру әдістері, ықтималдықтар теориясы, графиктер теориясы, сандық түрде мәліметтермен жұмыс істеудің әртүрлі әдістері қолданылады. [3]

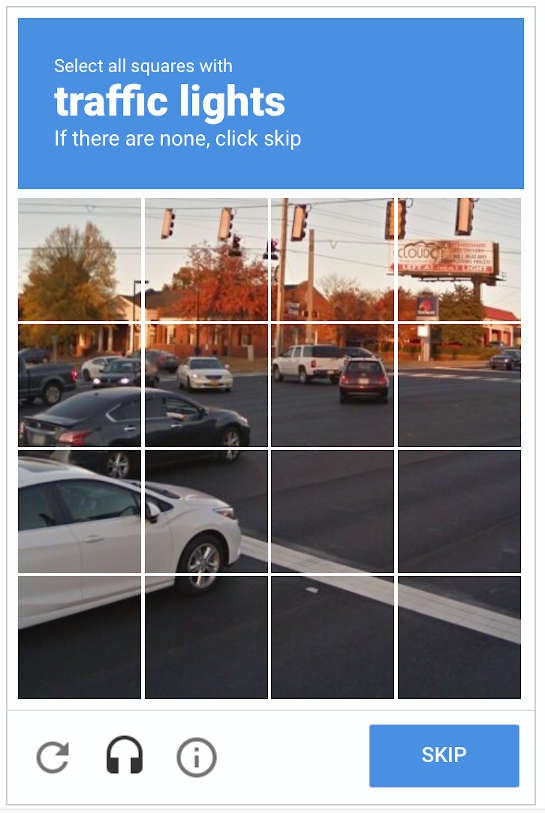
Машиналық оқыту көптеген ұқсас тапсырмаларды өңдеуге негізделген, мұнда кіріс, шарт және тиісті дұрыс жауап немесе мүмкін бірнеше жауаптардың бірі бар. Мысалы, машиналық аударма жағдайында кіріс бір тілдегі мәтін, ал дұрыс жауап оны басқа тілге аудару.

Машиналық оқыту әдістері компьютерлерге нақты бағдарламалауды қажет етпестен өздігінен үйренуге және жұмыс істеуге мүмкіндік береді. Бұл қосымшалар жаңа деректерді алады және өздігінен үйренуге, дамытуға және бейімделуге қабілетті.

Машиналық оқыту заңдылықтарды анықтау және процесте жақсарту үшін алгоритмдерді қолдана отырып, деректердің үлкен көлемін талдайды. Алдын ала берілген теңдеулерді модель ретінде пайдаланудың орнына, машиналық оқыту алгоритмдері есептеу әдістері арқылы деректерден тікелей оқытылады.

Мәселелерді шешу үшін машиналық оқыту қажет:

1. Деректер – бұл шешімдердің мысалдары және статистика, мәтіндік мысалдар, есептеулер, көрсеткіштер және тарихи оқиғалар сияқты оқу процесінде көмектесетін барлық нәрсе. Деректер жылдар бойы жиналады және барлық IT корпорациялары бар үлкен массивтерге – деректер жиынтығына біріктіріледі. Жинаудың мысалы автомобильдермен барлық фотосуреттерді таңдауды және дұрыс жауаптарды есте сақтауды сұрайтын captcha (Сурет-1.2).



Сурет-1.2 CAPTCHA

2. Белгілер – бұл машина оқу процесінде назар аударуы керек деректердің ерекшеліктері немесе сипаттамалары. Мысалы, акциялардың бағасы, жануардың бейнесі, сөз жиілігі немесе адамның жынысы. Белгілер неғұрлым аз болса және олар неғұрлым айқын белгіленіп, безендірілсе, соғұрлым үйрену оңайырақ болады. Күрделі тапсырмалар үшін қазіргі модельдер кірістердің шығысқа қалай айналатынын анықтайтын ондаған миллион параметрлерді ескеруі керек.

3. Алгоритмдер – бұл машиналық оқыту мәселесін шешу жолдары. Олар түпкілік мақсатқа жету үшін деректердің қалай өңделетінін және талданатынын анықтайды. Бір тапсырма үшін көптеген алгоритмдер болуы мүмкін, олардың әрқайсысының күшті және әлсіз жақтары бар. Ең қолайлы алгоритмді таңдау деректердің сипаттамаларына, есептің қойылуына және қажетті дәлдік пен тиімділікке байланысты. [4]

Машиналық оқытудың негізгі тұжырымдамаларына мыналар жатады:

1. Оқыту түрлері:

- Мұғаліммен оқыту: модель әр мысалда тиісті белгі немесе шығыс бар белгіленген деректерде оқытылады.

- Мұғалімсіз оқыту: модель белгіленбеген мәліметтерден үйренеді және олардан жасырын құрылымдарды немесе үлгілерді табуға тырысады.

- Күшейту жаттығулары: модель оқытылады қоршаған ортамен өзара әрекеттесу арқылы және өз әрекеттері үшін сыйақы немесе айыппұл түрінде кері байланыс алады. Тапсырмаларға агенттерді ойындарда немесе робототехникада оқыту кіреді.

2. Оқыту әдістері:

- Нейрондық желілер: көп қабатты архитектуралары бар мидың жұмыс принципі бойынша құрылған нейрондық желілерді модельдеу.

- Шешуші ағаштар мен кездейсоқ ормандар: ережелерге негізделген шешім ағаштарын құру және қайта оқытуды азайту үшін екеуінің тіркесімі.

- Анықтамалық векторлық әдіс: деректер кластары арасында оңтайлы бөлу гипержобасын табу.

- Кластерлеу және өлшемді азайту әдістері: деректерді ұқсас белгілер топтастыру және өнімділік пен интерпретацияны жақсарту үшін өлшемді азайту.

Машиналық оқыту әртүрлі салаларда деректерді талдау және шешім қабылдау тәсілдерін өзгертетін мықты құрал. Бұл жүйелер мен бағдарламаларға деректердің үлкен көлемінен құнды ақпаратты алуға және оны болжау, жіктеу, оңтайландыру және шешім қабылдау үшін пайдалануға мүмкіндік береді.

Машиналық оқыту тек бетті тану және мәтінді автоматты түрде аудару үшін ғана емес, сонымен қатар қаржылық тенденцияларды болжау, жеке медициналық нұсқаулар мен нұсқаулық жүйелерін құру күрделі тапсырмалар үшін де қолданылады. Машиналық оқыту алгоритмдері деректерді өңдеуге мүмкіндік береді, олардағы адам көзіне көрінбейтін заңдылықтарды анықтайды. Бұл өзгермелі жағдайларға бейімделе алатын және өткен деректерді талдау негізінде болашақ оқиғаларды болжай алатын дәлірек және тиімді жүйелерді құруға ықпал етеді. [5]

1.3 Нейрондық желілер жасанды интеллект миы

Нейрондық желі (ағыл. Neural Network) – компьютерлерді интеллектуалды шешімдерді қажет ететін тапсырмаларды орындауға үйрету үшін қолданылатын адам миының жұмысынан шабыттанған математикалық модельдер. Нейрондық желілер ақпаратты өңдеу үшін параллель жұмыс істейтін нейрондар деп аталатын көптеген байланысты элементтерден тұрады. [6]

Негізгі нейрондық желіде жасанды нейрондардың үш қабаты бар:

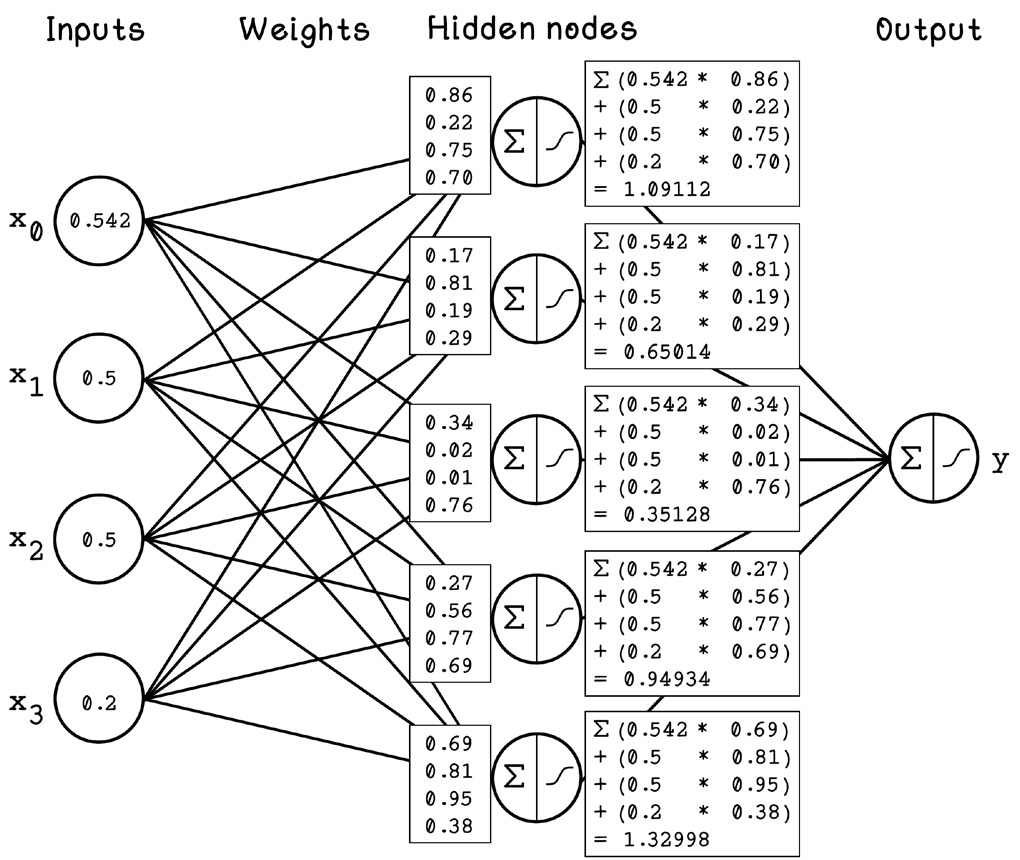
- Кіріс: ақпаратты сырттан өңдейді, талдайды немесе жіктейді және келесі қабатқа жібереді.

- Жасырын: алдыңғы қабаттың шығуын талдайды, өңдейді және келесіге жібереді.

- Шығу: барлық деректерді өңдегеннен кейін соңғы нәтиже береді.

Терең нейрондық желілер ерекшеленеді, өйткені олардағы жасанды нейрондар бір-бірімен байланысады және әрбір осындай байланысқа оның маңыздылығы көрсететін белгілі бір салмақ беріледі. Сонымен қатар, нейрондар арасындағы байланыс белсенді болуы мүмкін. Бұл дегеніміз, деректер олар арқылы тек бір бағытта өтеді. Бұл қосылыстың салмағы мәні берілгеннен төмен болса орын алады.

Нейрондық желіні оқыту кезінде оның барлық салмақтары бастапқыда кездейсоқ мәндермен беріледі. Оқыту деректері төменгі немесе кіріс қабатына беріледі. Содан кейін олар сақтау нүктесі жеткенше келесі қабаттар арқылы өтеді. Оқыту кезінде салмақ және шекті мәндер оқу деректері үнемі бірдей нәтиже бергенге дейін үнемі түзетіліп отырады (Сурет-1.3).



Сурет-1.3 Қарапайым нейрондық желі

Нейрондық желілерді оқыту түрлері:

- Бақыланатын оқыту: деректер мамандары нейрондық желіні оқыту үшін алдын ала дұрыс жауап беретін белгілі бір деректер жинағын жүктеп алады. Оқыту процесінде нейрожелі білімді жинақтайды, содан кейін оның болжамдар-

ын құру үшін жаңа мәліметтер алады.

- Терең оқыту: деректер маманы нейрондық желіге тек шикі деректерді жібереді, ал ол функцияларды өздігінен шығарады және оларды өздігінен зерттейді. Егер нәтиже қанағаттанарлықсыз болса, нейрондық желі дұрыс жауап бергенше оқу циклі қайтадан қайталанады. [7]

Нейрондық желілерді оқыту процесі:

- Ақпаратты жіберу: нейрондық оқытуды бастау үшін деректерді таңдау қажет. Нейрондық желі неғұрлым көп тапсырмаларды орындауы керек болса, соғұрлым бұл үлгі үлкенірек болады. Оқу процесінде алгоритмдер қойылған сұраққа формулалар мен сандық коэффициенттер арқылы жауап береді.

- Ақпаратты түрлендіру: нейрондық желінің кіріс нейрондары үлгіні алған кезде, олар оны түрлендіріп, әрі қарай жібереді. Бұл жағдайда ақпараттың мазмұны санды коэффициенттерге айналады. Ол үшін оқу авторлары белгілейтін коэффициенттері бар формулалар қолданылады. Олар әр оқу белгісінің салмағы қанша болатынын және т.б. шешеді.

- Ақпаратты өңдеу: әрбір нейронға оның жауаптарының қаншалықты маңызды екенін көрсететін салмақ беріледі. Жаттығу кезінде салмақтар автомат-

ты түрде өзгереді және теңгеріледі.

- Нәтиже алу: нейрондық желі жауап ретінде түрлендірілетін формулалар мен сандар жиынтығын шығарады. Бұл жағдайда ол нақты емес, керісінше ықтималдығын көрсетеді. Мысалы, егер ер адамның бейнесі үшін 0, ал әйел үшін 1 болса, онда 0,67 нәтижесі бұл әйел болуы мүмкін дегенді білдіреді.

Нейрондық желілердің артықшылықтары:

- Үлкен көлемдегі деректерді оқыту мүмкіндігі: нейрондық желілер үлкен көлемдегі деректерді тиімді өңдей алады және олардан күрделі үлгілерді шығара алады.

- Белгілерді автоматты түрде алу: нейрондық желілер кірістерден маңызды белгілерді автоматты түрде анықтай алады, бұл оларды әртүрлі машиналық оқы-

ту тапсырмалары үшін тиімді етеді.

- Әмбебаптық және икемділік: нейрондық желілерді әртүрлі деректер түрлеріне қолдануға және жіктеу, регрессия, сегментация және генерацияны қоса алғанда, әртүрлі тапсырмаларды шешуге болады. [8]

Нейрондық желілер сурет пен мәтін генерациялау, сөйлеуді тану және т.б. сияқты әртүрлі заттарға кеңінен қолданылады. Машиналық оқыту мен жасанды интеллект саласындағы ең белсенді зерттелген және қолданылатын әдістердің біріне айналды.

Нейрондық желілерді суреттерді генерациялау немесе жаңа графикалық элементтер жасау үшін және мәтінді генерациялау немесе мәтіндік деректердің үлкен көлемін талдау үшін пайдалануға болады.

1.4 Терең оқыту технологиясы

Терең оқыту (ағыл. Deep Learning) – деректерді өңдеу және шешім қабыл-

дау үшін пайдаланылатын үлгілерді жасау процесінде адам миының жұмысын имитациялауға негізделген машиналық оқыту әдістерінің кең тобының жиынтығы. Терең оқыту үлкен көлемдегі деректермен жұмыс істеуге арналған және модельді оқыту үшін күрделі алгоритмдерді қолданады. Үлкен деректер жиынтығында терең оқыту дәстүрлі машиналық оқытумен салыстырғанда нәтижелердің жоғары дәлдігін көрсетеді. [9]



Сурет-1.4 Терең оқыту жасанды интеллекттің соңғы бөлігі

Терең оқытудың негізгі компоненттері:

1. Нейрондар: кіріс қабылдайтын, өңдейтін және шығыс жасайтын нейрондық желінің негізгі бірліктері.

2. Байланыстар: сигналдар берілетін нейрондар арасындағы байланыстар.

3. Қабаттар: бір-бірімен байланысқан нейрондар тобы.

4. Активтендіру функциялары: нейронның шығуы оның кірісіне байланыс-

ты анықтайтын математикалық функциялар.

5. Оқыту алгоритмдері: деректерде нейрондық желіні оқыту үшін қолданы-

латын әдістер.

6. Жоғалту функциясы: модель қатесін бағалайтын және оны оңтайланды-

ру үшін қолданылатын функция.

7. Оптимизаторлар: оқу процесінде нейрондық желінің салмағын жаңарту үшін қолданылатын алгоритмдер.

8. Параметрлерді инициализациялау: оқытуды бастамас бұрын нейрондық желі салмағының бастапқы мәндерін орнату процесі.

9. Желілік архитектуралар: әр түрлі мәселелерді шешу үшін арнайы жасалған нейрондық желілердегі қабаттар мен байланыстар ұйымдары.

Кесте-1.4 Терең оқыту мен машиналық оқытуды салыстыру

|  |  |  |
| --- | --- | --- |
| Параметр | Терең оқыту | Машиналық оқыту |
| Деректер саны | өте үлкен | үлкен |
| Қажетті есептеу ресурстары | жоғары | төмен |
| Оқу уақыты | ұзақ | тез |
| Дәлдік | өте жоғары | жақсы |
| Декодтау | өте қиын | қарапайым |
| Шығу деректері | сан, ұпай, сынып, мәтін, сурет, дыбыс | сан, ұпай, сынып |
| Бағасы | өте қымбат | қымбат емес |

Терең оқыту компьютерлік көру, табиғи тілді өңдеу, автономды жүйелер, медицина және биоинформатика, қаржылық аналитика, ойын индустриясы және т.б салаларда қолданылады. Компьютерлік көруде терең нейрондық желілер объектілерді тану және кескіндерді жіктеу үшін, табиғи тілді өңдеуде машиналық аударма және мәтінді талдау үшін, автономды жүйелерде қоршаған ортаны қабылдау және шешім қабылдау үшін қолданылады. Медицинада терең оқыту медициналық бейнелер талдауға және ауруларды болжауға, ал қаржылық аналитикада акциялардың бағасын болжауға және алаяқтықты анықтауға көмектеседі. Ойын индустриясында интеллектуалды ойын кейіпкерлерін құру және графиканы жақсарту үшін терең оқыту қолданылады. [10]

Терең оқыту жеткілікті ресурстарды қажет ететін технология. Ол неғұрлым мықты графикалық процессорларды, өнімділігі жоғары графикалық карталарды, модельдерді оқыту үшін үлкен жадты және т.б. қажет етеді.

Барлық кемшіліктерге қарамастан, тереңдетіп оқытудың жетілдірілген әдістері құрылымдалмаған деректердің үлкен көлемін тиімді талдауға жаңа мүмкіндіктер ашады. Өз міндеттерінде терең оқытуды қолданатын компаниялар жүйені оқыту көп уақыт жұмсамай-ақ аналитиканың дәл нәтижелерін ала алады.

1.5 Табиғи тілді өңдеу, LLM мен LLaMA

Табиғи тілді өңдеу (ағыл. Natural Language Processing) – жасанды интелект пен математикалық лингвистика бағыты, онда компьютерлер адам тілімен талдайды және өзара әрекеттеседі. NLP ағылшын, испан және қытай сияқты табиғи тілдердегі мәтіндік деректермен жұмыс істеу үшін әртүрлі әдістер мен әдістерді қолданады. Бұл мәтінді түсіну мен түсіндіруді ғана емес, сонымен қатар компьютерлерді қолдана отырып жаңа мазмұн құруды да қамтиды. [11]

Смартфондар мен ноутбуктерге енгізілген сөйлеуді тану жүйелері ауызша сөйлеуді мәтінге айналдыру үшін NLP әдістерін қолданады. Бұл жүйелерді әртүрлі тілдерді, диалектілерді және екпіндерді тануға және түсінуге үйретуге болады. NLP машиналық аударма, құжаттарды автоматты түрде жинақтау, мәтіндердегі көңіл-күйді талдау, сұрақтарға жауап беру және т.б. сияқты көптеген қосымшаларда қолданылады.

LLM – бұл мәтінді генерациялауға арналған жасанды нейрондық желінің бір түрі. Бұл модельдер табиғи тілдегі мәтіндерден тұратын деректердің үлкен көлемінде оқытылады және сөздер мен сөйлемдер арасындағы статистикалық байланыстарды анықтауға үйренеді. Осының арқасында адам жазған мәтінге ұқсас мәтін генерациялауға мүмкіндік береді.

OpenAI GPT сериясы, Google Gemini, Meta LLaMA сияқты үлкен тілдік модель нейрондық желілердің озық архитектураларына негізделген. Бұл модельдер әр түрлі NLP тапсырмаларына тез бейімделе алады, бұл әр нақты тапсырма үшін дәл баптауды қажет ететін ерте тәсілдерден ерекшеленеді.

OpenAI ChatGPT-3 модель мәтін құра алады, сұрақтарды түсінеді және жауаптар ұсына алады, сонымен қатар NLP-дің басқа тапсырмаларын әрқайсысы үшін дәл баптауды қажет етпестен орындай алады. Бұл LLM синтаксис, семантика және басқа лингвистикалық аспектілер туралы білім алады, сонымен қатар олар оқытылатын мәтіндік деректердегі нюанстар мен жағымсыздықтарды қабылдай алады. [12]

LLM негізгі сипаттамалары:

1. Мәтінді құру: жаңалықтар мақалалары, әдеби шығармалар немесе техникалық құжаттар болсын, әртүрлі дереккөздердің стилі мен мазмұнына елік-

тейтін мәтін жасай алады.

2. Автоматты аударма: мәтінді бір тілден екінші тілге аударғанда, мағына-

сы мен контекстін сақтайды.

3. Мазмұнды құру: өлеңдер, сценарийлер, музыкалық шығармалар және шығармашылық мазмұнның әртүрлі түрлерін жасай алады.

4. Сұрақтарға жауаптар: сұрақтарға ашық, күрделі немесе пайымдауды қажет етседе, оқу деректеріндегі ақпаратты пайдалана отырып жауап бере алады.

5. Диалогты құру: адамдар арасындағы диалогты имитациялайтын мәтін жасай алады, бұл оларды чаттарды, виртуалды көмекшілерді және табиғи тілде пайдаланушымен өзара әрекеттесетін басқа жүйелерді дамыту үшін пайдалануға мүмкіндік береді.

LLaMA – қол жетімді, ашық LLM құруға бағытталған жоба. Meta әзірлеген LLaMA өзінің ауқымдылығы мен тиімділігімен басқа LLM-ден ерекшеленеді. Ол аз қуатты құрылғыларда жұмыс істей алады, бұл оны пайдаланушылардың кең ауқымына қол жетімді етеді. [13]

Ең қуатты LLM әдетте шектеулі API арқылы ғана қол жетімді болғанымен, Meta коммерциялық емес лицензиямен зерттеу қауымдастығы үшін Llama моделін шығарды. Meta AI LLaMA-ның жаңа моделі әлі де дамып келеді.

LLaMA қарқынды дамып келе жатқан зерттеу саласы және алдағы жылдары жаңа және қуатты модельдер пайда болады деп күтуге болады. LLaMA-ның дамуы машиналық аударма, мәтін генерациялауға және сұрақ-жауап жүйелері сияқты әртүрлі салаларға маңызды үлес. Бұл пайдаланушылармен жақсы түсінетін және өзара әрекеттесетін ақылды қосымшаларды құрудың негізгі құралы бола алады.

Кесте-1.5 LLM мен LLaMA-ды салыстыру

|  |  |  |
| --- | --- | --- |
| Сипаттама | LLM | LLaMA |
| Әзірлеуші | Google AI | Meta AI |
| Модель өлшемі | өте үлкен | үлкен |
| Деректер жиынының өлшемі | 1,56 Т сөздер | 600 В сөздер |
| Қол жетімділік | ақылы | ашық |
| Күрделілігі | жоғары | төмен |
| Қажетті есептеу ресурстары | үлкен | орташа |

Meta AI өзінің үлкен тілдік модельдерін дамытуды жалғастыруда және Llama3 моделінің жаңа нұсқасын ұсынды. Бүгінгі күнге дейін ең мықты ашық LLM, бұл Llama2-нің алдыңғы нұсқасынан айтарлықтай жақсаруды білдіреді.

Llama3 алдыңғы Llama моделінің масштабтау және жақсартылған өнімділік негізгі ерекшеліктері сақтады, сонымен бірге бірқатар жаңа мүмкіндік-

тер мен функцияларды қосады. Llama3-тің басты ерекшеліктерінің бірі оның логикалық қорытындыны қажет ететін күрделі сұрақтарды жақсы түсіну және оларға жауап беру қабілеті.

Llama3 ашық лицензиямен қол жетімді, бұл оны ғылыми және коммерция-

лық мақсатта пайдалануға мүмкіндік береді. Болашақта модель мәтіннен басқа, суреттер мен аудионы қоса алғанда, әртүрлі деректер түрлерімен жұмыс істеуді қолдайды.

1.6 Ollama платформасы

Ollama – үлкен тілдік модельдерді жергілікті іске қосуға арналған құрал. Бұл құрал зерттеушілер мен әзірлеушілерге бұлтқа негізделген шешімдерге немесе шектеулі API-ге сүйенудің орнына жергілікті компьютерде немесе серверде қуатты тілдік модельдермен жұмыс істеуге мүмкіндік береді. [14]

Ollama-ның негізгі ерекшеліктерінің бірі – оның жергілікті құрылғыда үлкен көлемдегі деректер мен есептеулерді өңдеу қабілеті, бұл модельдерді оқыту мен пайдаланудың жылдам әрі тиімді процесін қамтамасыз етеді. Бұл әсіресе сезімтал деректермен жұмыс істейтін немесе модельдері мен есептеулер-

ін бақылаудың жоғары дәрежесін қажет ететін зерттеушілер мен компаниялар үшін өте маңызды.

Ollama тілдік модельдерді басқару және өзара әрекеттесу үшін ыңғайлы интерфейсті ұсынады, соның ішінде модельдерді жүктеу, оқыту және пайдалану, модельдердің нәтижелерін талдау мүмкіндігі. Бұл құрал LLM тәжірибесін зерттеушілерді, әзірлеушілерді және қызықты энтузиастарды қоса алғанда, пайдаланушылардың кең ауқымы үшін қол жетімді және ыңғайлы етеді.

Ollama қуатты тілдік модельдерге қол жеткізуді қамтамасыз ететін және олармен жұмыс істеу процесін жеңілдететін жасанды интеллект саласындағы зерттеулер мен әзірлемелер инфрақұрылымындағы маңызды құрал болып табылады.

Ollama пайдаланған кезде құпиялылықты, шығындарды, бақылауды, қолжетімділік, пайдаланудың қарапайымдылығы, әмбебаптығы, интеграциясын және жергілікті жұмыс мүмкіндігін ескеретін маңызды артықшылықтарға ие:

1. Құпиялылық: деректер компьютерде қалады, ақпараттың құпиялылығы мен қауіпсіздігін қамтамасыз етеді, әсіресе құпия деректермен жұмыс істеу үшін маңызды.

2. Құны: жергілікті LLM пайдалану бұлтты шешімдермен салыстырғанда үнемді болуы мүмкін, әсіресе ресурстарды қажет ететін тапсырмаларды орындау кезінде, бұл есептеу шығындарын айтарлықтай төмендетуі мүмкін.

3. Бақылау: LLM және деректерді толық басқара аласыз, бұл оларды жоба-

ның қажеттіліктері мен талаптарына сәйкес реттеуге мүмкіндік береді.

4. Қол жетімділік: Ollama Linux, macOS және Windows сияқты әртүрлі операциялық жүйелерде жұмыс істейді, бұл оны көптеген пайдаланушыларға қол жетімді етеді.

5. Пайдаланудың қарапайымдылығы: Ollama командалық жолының қарапайым интерфейсі оны пайдалану мен басқару жеңілдететін бағдарламалау тәжірибесі жоқ пайдаланушылар үшін де қол жетімді етеді.

6. Әмбебаптық: Ollama әртүрлі LLM үлгілерімен үйлесімді, бұл Сіздің тапсырмаңызға немесе жобаңызға ең қолайлы үлгіні таңдауға мүмкіндік береді.

7. Интеграция: Ollama-ны басқа қосымшалармен оңай біріктіруге болады, бұл сізге күрделі және қуатты чаттар немесе жасанды интеллектті қолданатын басқа қосымшалар жасауға мүмкіндік береді.

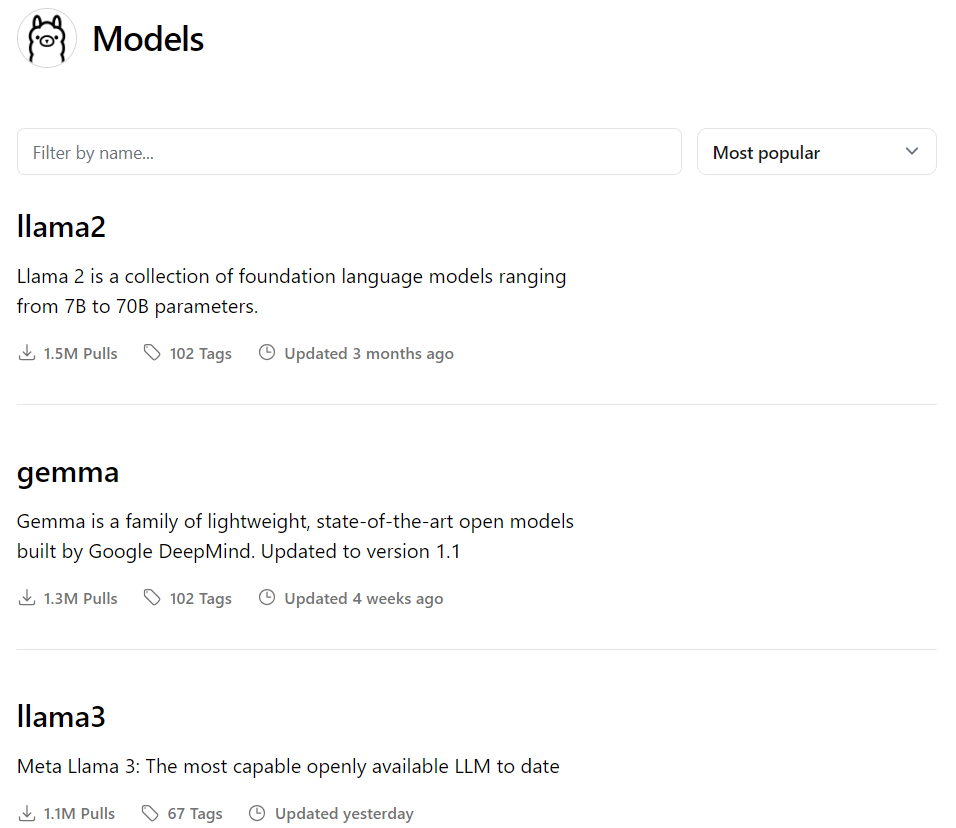
8. Жергілікті жұмыс: Ollama LLM-мен жергілікті, тіпті интернетке қол жеткізбестен жұмыс істеуге мүмкіндік береді, бұл әсіресе шектеулі немесе сенімсіз интернет жағдайында пайдалы. Бұл жұмыстың үздіксіздігін және кез-келген жағдайда модельдерді пайдалану мүмкіндігін қамтамасыз етеді. [15]

Ollama-ны қолдануды бастау қарапайым және барлық операциялық жүйелер үшін қол жетімді. Linux пайдаланушылары қарапайым орнату сценарийін пайдалана алады, ал macOS және Windows пайдаланушылары арнайы орнатушыларды пайдалана алады. Ollama Docker ресми көрінісі де бар, ол контейнерлеуді білетіндер үшін процесті жеңілдетеді.

Ollama кез-келген тапсырма үшін дұрыс құралдардың болуын қамтамасыз ететін оның кең модельдік кітапханасы. Llama әмбебап моделінен Gemma және Dolphin Phi сияқты мамандандырылған модельдерге дейін нақты қажеттіліктері қанағаттандыру үшін көптеген нұсқалар болады. Платформаның құжаттамасын-

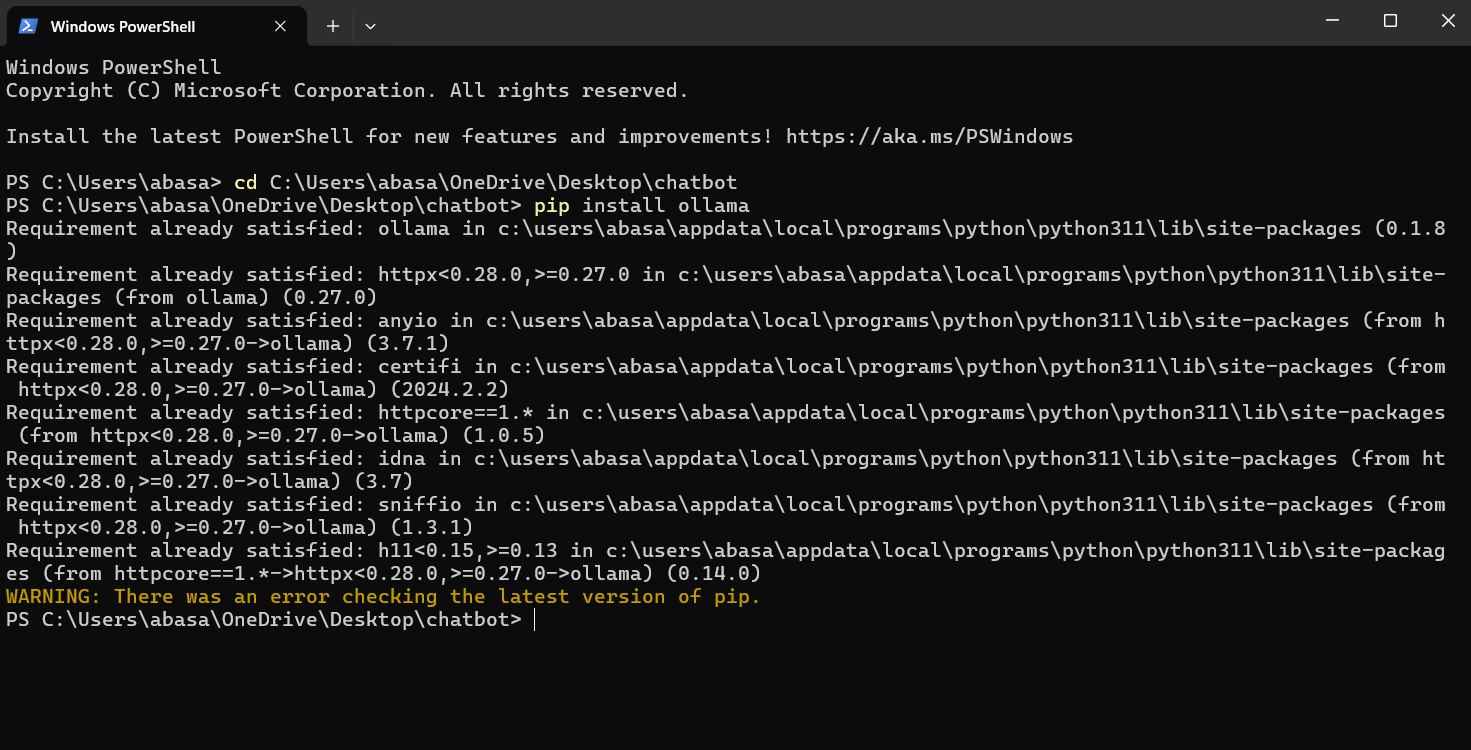
да осы модельдерді жүктеу және іске қосу туралы егжей-тегжейлі нұсқаулар бар, бұл Ollama-ны бастауды қарапайым және қарапайым етеді.

Python-да жұмыс істейтін пайдаланушылар үшін Ollama қарапайым интеграцияны ұсынады, бұл жергілікті тілдегі модельдерді өз жобаларына қосуды жеңілдетеді. Бұл күрделі чатботтарды жасаудан бастап, деректерді талдау құралдарын жетілдіруге дейінгі көптеген мүмкіндіктерді ашады, сонымен бірге жергілікті компьютерде деректерді сенімді сақтауды қамтамасыз етеді.



Сурет-1.6 Ollama-ның ең танымал модельдері

Ollama орнату жергілікті тіл модельдерін оңай бастауға мүмкіндік беретін қарапайым және жылдам процесс. Орнатуды бастамас бұрын компьютерде Python 3.11 немесе одан жоғары нұсқасы бар екеніне көз жеткізу керек. Ollama орнату үшін терминалды ашып, келесі кодты орындау керек (Сурет 1.6.1).



Сурет 1.6.1 Ollama-ны орнату процессі

Орнату процесі аяқталғаннан кейін Ollama-ны жергілікті жасанды интеллект модельдерімен жұмыс істеу үшін пайдалануға дайын. Бұл интернет болмаған кезде де деректеріңізді, құпиялылығыңызды және қолжетімділігіңізді бақылауды қамтамасыз етеді.

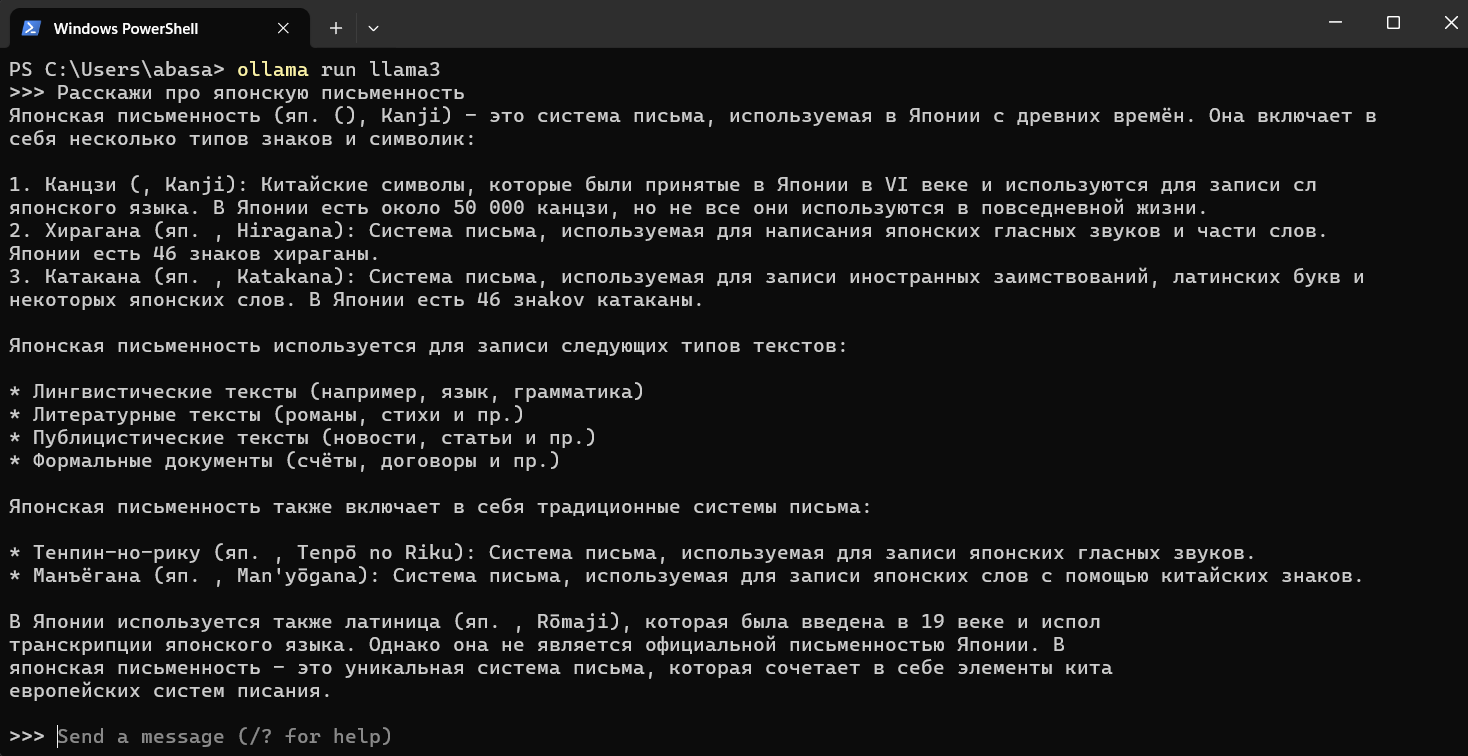
Терминалда ollama командасын пайдаланған кезде қол жетімді командалар және мүмкіндіктер тексеруге болады. Бұл команданың орындаудың нәтижесі қол жетімді командалардың тізімі, сондай-ақ олардың сипаттамалары болады, бұл осы бағдарламаның қандай функциялары бар екенін және оларды қалай пайдалану керектігін жақсы түсінуге көмектеседі.

Ollama функционалдығын сәтті тексергеннен кейін Llama3 моделін орнатуды бастауға болады. Llama3 моделін орнату үшін терминалға ollama run Llama3 командасын енгізу керек. Бұл Llama3 моделін компьютерге жүктеу және орнату процесін бастайды.

Llama3 моделін орнатуды аяқтағаннан кейін модельді сәтті орнату және пайдалануға дайын болу туралы хабарламасы шығады. Енді мәтіндік деректерді өңдеуге және табиғи тілді өңдеуге қатысты әртүрлі тапсырмаларды орындауға арналған қуатты моделін қолдануға болады.

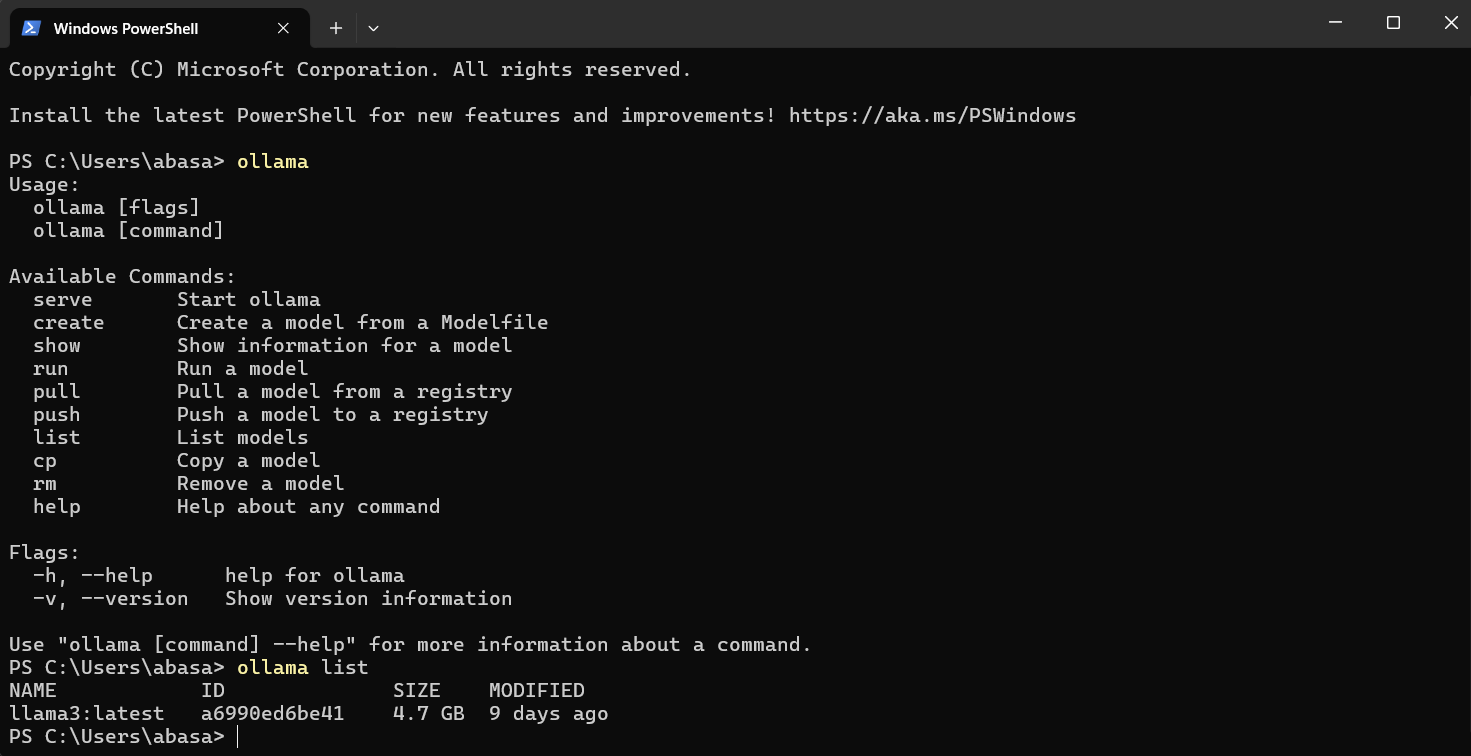
Llama3 моделін орнатқаннан кейін онымен өзара әрекеттесуді бастауға болады. Ол үшін терминалда сұраулар жіберуге және Llama3 моделінен жауап алуға болады. Терминалға кез келген мәтін жолын енгізейік. Мысалы, жапон тілінде қалай жазу керектігін сұрағым келеді.

Llama3 моделі сұрағанды талдайды және тиісті жауап немесе шешім береді. Бұл сұрақтарға жауап беруден бастап мәтін құруға немесе табиғи тілді өңдеуге қатысты басқа мәселелерді шешуге дейін болуы мүмкін (Сурет 1.6.2).



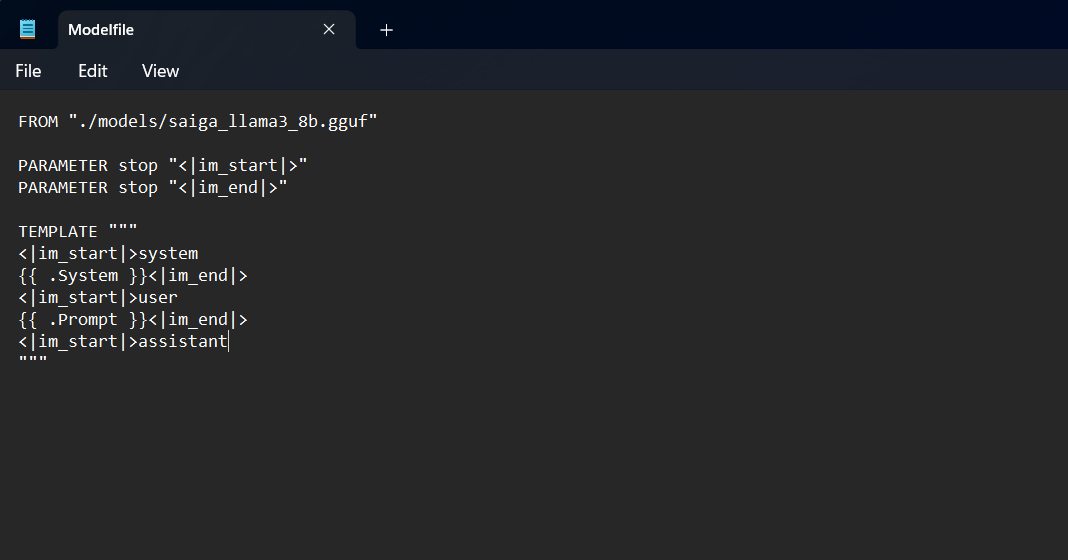
Сурет 1.6.2 Llama3 моделінің жауабы

Llama3 моделі бар-жоғын білу үшін ollama list командасы арқылы барлық қол жетімді және орнатылған модельдердің тізімін зерттеуге болады. Команда орындау нәтижесінде барлық орнатылған модельдердің тізімі мен олардың сипаттамалары мен нұсқалары көрсетіледі (Сурет 1.6.3).



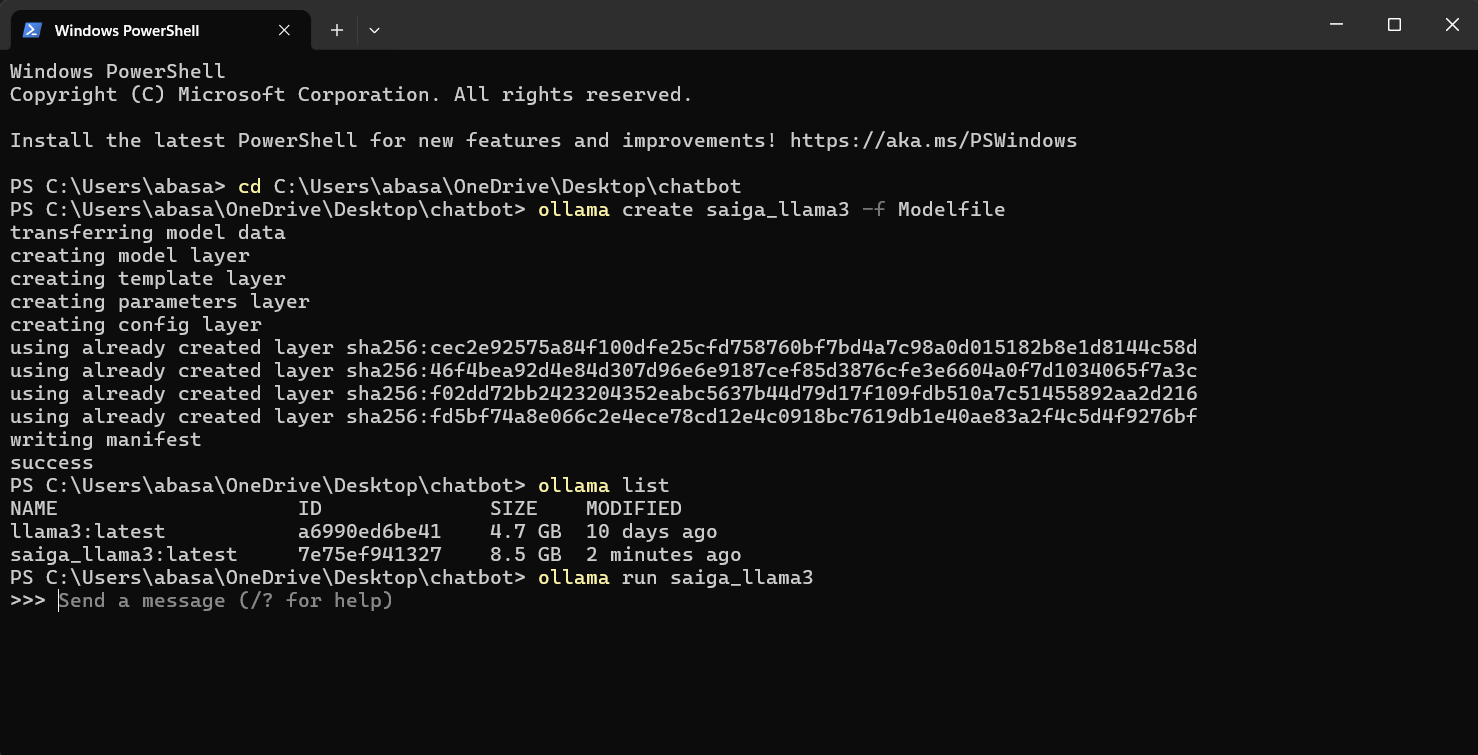
Сурет 1.6.3 Ollama командаларының нәтижесі

Жаңа модель қосу үшін Ollama-ның Modelfile форматын пайдалану қажет. Бұл файл модель мінез-құлқы мен функционалдығын анықтайтын параметрлер мен конфигурацияларды анықтау үшін қолданылады. Modelfile-де модельдің негізгі архитектурасы, пайдаланушының өзара әрекеттесу механизмі, тоқтату ережелері және т.б. жазылады. Бұл опциялар жобаның талаптарына немесе шешкін келетін нақты тапсырмаға сәйкес реттеуге мүмкіндік береді. Modelfile деп аталатын жаңа файл керек. Оны мәтіндік редакторда ашып, ішіне келесі кодты жазамыз (Сурет 1.6.4).



Сурет 1.6.4 Жаңа модель параметрлері

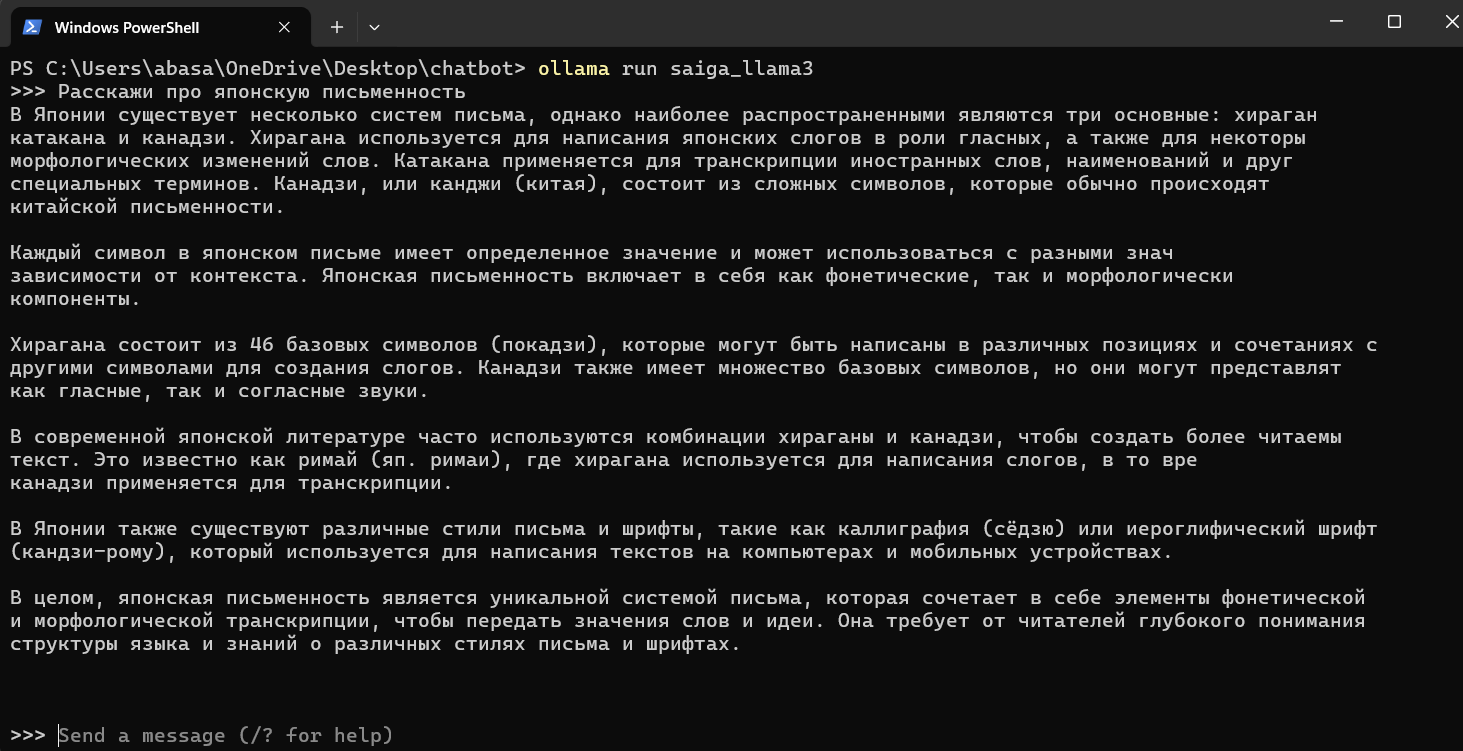
Modelfile файлын жасағаннан кейін ollama create командасын пайдаланып жаңа моделін қосуға болады. Терминалда ollama create saiga llama3 -f Modelfile командасын жазамыз. Бұл Modelfile файлындағы параметрлер негізінде saiga llama3 деп аталатын жаңа модель жасайды. Модельдің бар-жоғын білу үшін ollama list командасын пайдаланып тексеруге болады (Сурет 1.6.5).



Сурет 1.6.5 Saiga моделін қосу процесі

Saiga – бұл орыс тілі үшін арнайы жасалған үлкен тілдік модель. Бұл Mistral және Llama сияқты жоғары сапалы деректерді қайта оқытудың нәтижесі. Saiga Llama3 модель деректер жиынтығында өңделген көптеген параметрлерді қамтиды, бұл алдыңғы Saiga Mistral нұсқадан төрт есе үлкен. Бұл ауқымды оқыту Saiga Llama3-ке кодты құру, мәтіндерді аудару және сұрақтарға жауап беру сияқты әртүрлі тапсырмаларды сәтті орындауға мүмкіндік береді. Осының арқасында Saiga LLM орыс тіліндегі сұраныстар мен тапсырмаларды өңдеуде жоғары дәлдік пен тиімділікке ие.

Saiga Llama3 моделін сәтті орнатқаннан кейін, оны ollama run saiga llama3 командасының көмегімен қолдануға болады. Модельдің функционалдығын көрсету үшін, мен Llama3 моделінен сұраған жапон жазуы туралы ақпаратты тексеру болады (Сурет 1.6.6).



Сурет 1.6.6 Saiga моделінің жауабы

Екі модельден де жауаптар алғаннан кейін мен олардың сапасын, толықтығын және дәлдігін бағалауға мүмкіндік беретін салыстырмалы талдау жасадым. Салыстыру нәтижелері Saiga Llama3 моделі, ағылшын тілінде оқытылатын көптеген басқа LLM-ге қарағанда, негізінен орыс тіліндегі мазмұн деректерінде оқытылғанын көрсетті. Бұл оған осы тілдің ерекшеліктері мен нюанстарын ескере отырып, орыс тіліндегі сұраныстарды жақсы түсінуге және өңдеуге мүмкіндік береді.

Менің салыстырмалы талдауым көрсеткендей, Saiga Llama3 жапон тілінде қалай жазу керектігі деген сұрауда жақсы жұмыс көрсетті, өйткені ол орыс тілін практикалық қолдануға бағытталған. Оның негізгі орыс тіліндегі контентте оқуы, оған орыс тіліндегі сұраныстарды жақсы түсінуге және өңдеуге мүмкіндік береді, бұл оны осындай сұраныстар үшін таңдаулы етеді.

Llama3 жапон жазуын үйренудің пайдалы құралы бола алады, әсіресе бұл жазу жүйесімен таныс емес адамдар үшін. Оның табиғи тілдегі сұраулар мен жауаптарды өңдеу мүмкіндіктері жапон жазуы мен оның ерекшеліктері туралы негізгі түсінік алғысы келетіндер үшін пайдалы болуы мүмкін.

Llama3 және Saiga моделдері жасанды интеллект чатында қолданылатын болады. Llama3 жалпы ақпарат алу және сұраулардың кең ауқымын өңдеу үшін көп тілдерді қолданады. Оның ағылшын және басқа тілдерде мәтінді өңдеудің үлкен мүмкіндіктері бар, бұл оны әртүрлі көздерден ақпарат алуға мүмкідік береді. Saiga Llama3 нақты тапсырмалар үшін, әсіресе орыс тіліндегі ақпаратпен жұмыс істеу контекстінде қолданылады. Орыс тілінен қазақ тіліне аудару және қазақ деректерімен жұмыс істеу үшін жасанды интеллект чатында пайдалану үшін таптырмас модель.

1.7 Жасанды интеллект чатының диалог жүйесі

Жасанды интеллект диалог жүйесі – адамдармен табиғи сөйлесуге қабілет-

ті, адамның қарым-қатынасына еліктейтін бағдарламалық құрылым. Ол келесі мақсаттарға жету үшін табиғи тілді өңдеу, машиналық оқыту және жасанды интеллект әдістерін қолданады:

- Пайдаланушы енгізуін түсіну: жүйенің пайдаланушының ниеттері мен сұрауларын, тіпті олар түсініксіз немесе құрылымдалмаған болса да, дұрыс түсіндіре алуы керек. Жүйе сұрақтардың, командалардың және мәлімдемелердің әртүрлі формаларын танып, тиісті жауап беру үшін контекстті ескеруі керек.

- Тиісті жауаптар жасау: жүйенің тек ақпараттандыратын ғана емес, сонымен қатар қазіргі сөйлесу контекстіне және пайдаланушының талаптарына сәйкес келетін жауаптар жасауды қамтиды. Жауаптар берілген сұрақты немесе сұрауды ескере отырып, пайдаланушыға сәйкес және пайдалы болуы керек.

- Мәтінмәнді сақтау: жүйе пайдаланушымен сөйлесудің алдыңғы қадамдары мен мәліметтерін қадағалап, есте сақтай алуы керек. Алдыңғы сұрақтар мен жауаптар, негізгі ойлар мен нақтылаулар сияқты контексттік ақпарат диалогтың логикасы мен дәйектілігін сақтау үшін пайдаланылады, осылайша жоғары сапалы және өнімді өзара әрекеттесуді қамтамасыз етеді.

- Қарым-қатынастың табиғи болуын қамтамасыз ету: пайдаланушымен қарым-қатынас кезінде жүйе өзін табиғи ұстауы керек. Бұған ыңғайлы және түсінікті тілді пайдалану, сыпайы өрнек формалары сияқты әртүрлі қарым-қатынас стильдерін қолдау және мүмкіндігінше табиғи және ыңғайлы қарым-қатынас тәжірибесін жасау үшін пайдаланушының өзара әрекеттесуінің жеке ерекшеліктеріне бейімделу мүмкіндігі кіреді. [16]

Жасанды интеллект диалог жүйесінің міндеттері:

- Клиенттерге қызмет көрсету: жиі қойылатын сұрақтарға жауаптарды автоматтандыруды, компанияның өнімдері туралы анықтамалық ақпарат беруді, тапсырыстарды өңдеуді және шағымдарды шешуді қамтамасыз етеді.

- Виртуалды көмекшілер: пайдаланушыларға кездесулерді жоспарлау, билеттерді брондау, маңызды оқиғалар туралы еске салғыштар және т.б. сияқты күнделікті тапсырмаларды орындауға көмектесетін көмекшілер. Олар көптеген мүмкіндіктерге ие және пайдаланушыларға барынша ыңғайлылықты қамтамасыз ету үшін әртүрлі платформаларға біріктірілген.

- Білім: диалогтық жүйелер жеке оқыту, оқу материалдарын ұсыну, сондай-ақ әрбір нақты пайдаланушының оқу стиліне бейімделу үшін қолданылады. Бұл әр оқушының ерекшеліктері мен қажеттіліктерін ескере отырып, жеке білім беру тәжірибесін құруға мүмкіндік береді.

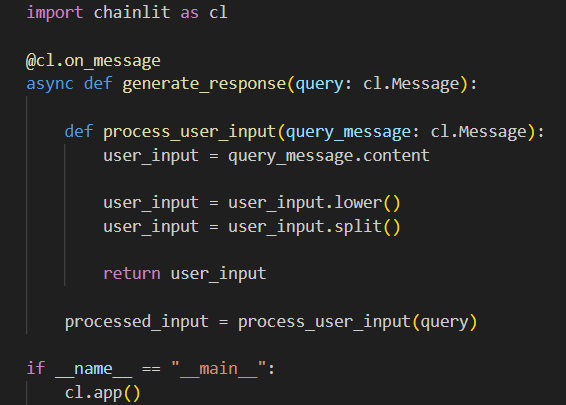
- Ойын-сауық: ойындарға, викториналарға және интерактивті әңгімелер құруға арналған чаттар пайдаланушылармен өзара әрекеттесу үшін диалогтық жүйелерді қолданатын ойын-сауық қосымшалары. Олар қызықты ойын оқиғалар жасауға, әртүрлі тақырыптағы викториналарға қатысуға және әртүрлі ойын-сауықтардан ләззат алуға мүмкіндік береді.

- Инклюзивтілік: диалогтық жүйелер есту немесе сөйлеу қабілеті нашар адамдар үшін ақпарат пен қызметтерге қол жеткізуде маңызды рөл атқарады. Олар жеке қажеттіліктеріне қарамастан барлық пайдаланушылар үшін ақпарат пен қызметтерді қолжетімді ету үшін мәтіндік чаттар, дауыстық интерфейстер сияқты балама байланыс әдістерін ұсынады.

Жасанды интеллект диалог жүйесі жасанды интеллект функцияларын жүзеге асыру үшін қажетті әртүрлі компоненттердің үйлесімді жұмысын қамтамасыз ететін ойластырылған архитектураға негізделген. Диалогты жүйенің архитектурасы әдетте келесі модульдерден тұрады:

1. Енгізу модулі – пайдаланушыдан келетін ақпаратты жинауға және өңдеуге жауап береді. Ол тиімді және өнімді өзара әрекеттесуді қамтамасыз етуге және пайдаланушының оң тәжірибесін құруға жауап береді. Енгізу модулі икемді және әртүрлі енгізу түрлерін, соның ішінде мәтіндік хабарларды, сөйлеуді, суреттерді немесе басқа деректер түрлерін дұрыс өңдеуге бейімделеді. Бұл компонент пайдаланушының кірісті тиімді өңдеуі жүйенің басқа компонент-

терінің одан әрі жұмыс істеуіне негіз болады және оның жалпы өнімділігі мен функционалдығын анықтайды (Сурет 1.7.1).



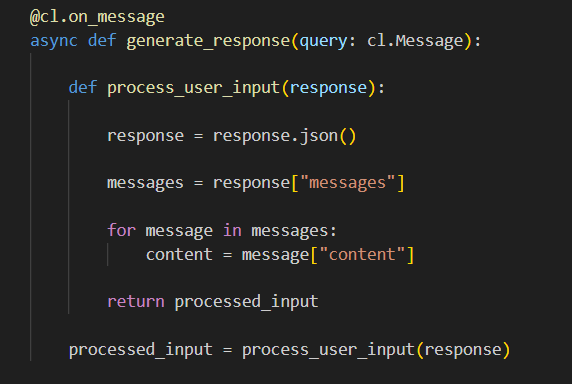
Сурет 1.7.1 Енгізу модулін жүзеге асыру коды

Process user input функциясы жасанды интеллект диалогтық жүйесінде одан әрі өңдеуге дайындау үшін пайдаланушыдан алынған мәтінді алдын ала өңдеуді орындайды. Ол cl Message және query message нысанын параметр ретінде қабылдайды және query message мазмұнынан хабарлама мәтінін шығарады. Мәтін кіші әріпке ауысады және сөздерге бөлінеді. Функцияға тоқтату сөздерін жою, лемматизация, стемминг сияқты қосымша өңдеу қадамдарын қосуға болады. Process user input функциясын шақыру нәтижесі диалогтық жүйенің компоненттерінде қолданылатын processed input айнымалысына тағайындалады. Өңделген мәтін дұрыс өңделген пайдаланушы кірісі қажет болған жағдайда қолданылады.

Енгізу модулі кодтың қарапайымдылығы мен икемділігін, оның диалогтық жүйенің қалған кодымен оңай интеграциялануын қамтамасыз етеді. Chainlit көмегімен енгізу модулін енгізу ұсынылған қосымша функцияларды қосу және оны белгілі бір тапсырмаларға бейімдеу үшін кодты өзгертуді жеңілдетеді.

2. Өңдеу модулі – пайдаланушының кірісін талдау мен түсінуге, ақпаратты алуға және диалогтың келесі кезеңдері үшін деректерді дайындауға жауап береді. Ол мәтінді талдауды, нысандар мен параметрлер сияқты негізгі сұрау элементтерін және одан әрі өңдеу мен жауап беру үшін деректерді шығаруды дайындайды. Бұл компонент жүйенің басқа компоненттерімен өзара әрекеттесе-

ді, жауап беру және диалогтың барысын басқару үшін қажетті ақпараттың берілуін қамтамасыз етеді (Сурет 1.7.2).



Сурет 1.7.2 Өңдеу модулін жүзеге асыру коды

Process user input функциясы response нысанын параметр ретінде қабылдайды. Ол жауап беруден json жауабын response json және хабарламалар тізімін response[messages] ішінен алады. Ол хабарламалар тізімінен өтіп, әр хабарламаның мазмұнын message[content] ішінен шығарып, оны өңделген мәтін-

ге қосады. Функцияға мәтінді өңдеуге қосымша тоқтату сөздерін жою, леммати-

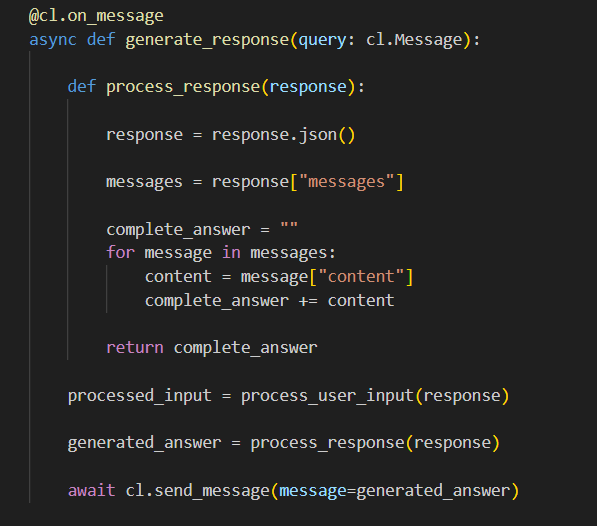
зация, стемминг қадамдарын қосуға болады. Мәтінді өңдегеннен кейін функция оны processed input ретінде қайтарады.

Алынған өңделген мәтін содан кейін модуль жауаптарын талдау және түсіндіру үшін диалогтық жүйенің келесі компоненттерінде қолданылатын processed input айнымалысына тағайындалады. Бұл өңдеу модулінің қарапайым-

дылығын, икемділігін және диалогтық жүйенің қалған кодымен интеграциялану-

ын қамтамасыз етеді. Chainlit көмегімен өңдеу модулін іске асыру жасанды интеллект диалогтық жүйесіндегі жауаптарды оңай және икемді талдауға және түсіндіруге мүмкіндік береді.

3. Жауап генерациялау модулі – пайдаланушының кірісіне табиғи және ақпаратты жауаптар генерациялауға жауап береді. Ол пайдаланушы тәжірибесін қалыптастыруды және тиімді чат қарым-қатынасын қамтамасыз етеді. Бұл компонент әртүрлі контексттерге бейімделе алатын және пайдаланушылардың сұраныстарына сәйкес жауап бере алатын күрделі алгоритмдер мен модельдерді қолданады (Сурет 1.7.3).



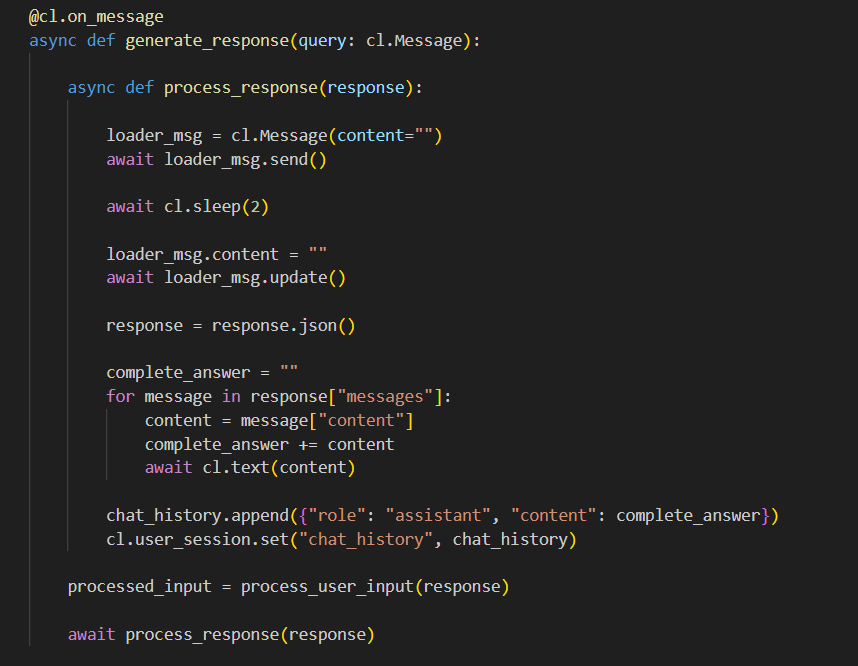
Сурет 1.7.3 Жауап генерациялау модулін жүзеге асыру коды

Process response функциясы серверден json жауабын өңдейді. Ол response нысанын қабылдайды, json жауабын және хабарламалар тізімін шығарады, хабарламалар тізімінен өтеді, әрқайсысының мазмұнын шығарады және оларды бір жолға біріктіреді, сол жолды генерацияланған жауап ретінде қайтарады. Process response шақырылғаннан кейін генерацияланған жауап generated answer айнымалысына тағайындалады, пайдаланушыға await cl send message функциясы арқылы жіберіледі. Ол серверден жауапты өңдейді және оны әрі қарай жұмыс істеу үшін ыңғайлы форматқа айналдырады.

Process user input функциясы пайдаланушыдан алынған мәтінді алдын-ала өңдеуді қамтамасыз етеді, бұл оны әрі қарай талдауға және жүйеде қолдануға дайындауға мүмкіндік береді. Бұл мәтінді кіші әріпке түрлендіруді, жеке сөздерге бөлу және қажет болған жағдайда тоқтату сөздерін жою, лемматизация немесе стемминг сияқты қосымша өңдеу қадамдарын қосуға болады.

4. Шығару модулі – пайдаланушыға генерацияланған жауапты ұсынуға жауап береді. Ол тиімді қарым-қатынасты және пайдаланушының тәжірибесін қамтамасыз етеді. Генерация жасалып, пайдаланушыға жіберу дайын болғанда, шығару модулі бұл жауапты алады және оны пайдаланушыға ыңғайлы форматта көрсетеді. Бұл экран бетіне шығаруды, электрондық поштаны жіберуді, веб-бетте жариялауды және т.б. қамтуы мүмкін. Бұл компонент қосымша енгізу немесе қосымша ақпарат қажет болған жағдайда пайдаланушының өзара әрекет-

тесуін қамтамасыз ете алады (Сурет 1.7.4).



Сурет 1.7.4 Шығару модулін жүзеге асыру коды

Process response функциясы response нысанын параметр ретінде қабылдайды. Алдымен cl Message(content) көмегімен күту хабары жасалады. Күту хабары await loader msg send арқылы жіберіледі, пайдаланушы жүктеу көрсеткішін көреді. Await cl sleep көмегімен жүктеу процесін модельдеу үшін 2 секундқа кідірту жасалады. Жүктеу аяқталғаннан кейін күту хабары loader msg content және await loader msg update көмегімен жойылады. Содан кейін json жауабы response json ішінен алынады. Жауап фрагменттері толық жауап айнымалысына қосылады. Әр үзінді үшін await cl text(content) көмегімен жауап біртіндеп көрсетіледі. Соңында диалог тарихы жаңартылып, толық жауап жіберіледі. Cl text пайдалану жауап беруді біртіндеп көрсетуге мүмкіндік береді, бұл пайдаланушының тәжірибесін жақсартады, пайдаланушыға сұранысты өңдеу барысы туралы хабарлайды және тірі байланыс сезімін тудырады.

1.8 Жасанды интеллект чатының веб-интерфейсі

Веб-интерфейс – пайдаланушының веб-қосымшамен немесе веб-сайтпен визуалды өзара әрекеттесуін қамтамасыз ететін бағдарламалық жасақтама компоненті. Бұл пайдаланушыларға қолданбаның мазмұны және функционалды-

ғымен өзара әрекеттесуге мүмкіндік беретін веб-шолғыш арқылы қол жетімді графикалық пайдаланушы интерфейсі (GUI). Жасанды интеллект чатының веб-интерфейсі пайдаланушыларға мәтіндік хабарламасы арқылы жасанды интелект-

пен байланысуға мүмкіндік беретін веб-қосымша болып табылады. [17]

Веб-интерфейстің негізгі функциялары:

- Деректерді енгізу: пайдаланушылар мәтіндік ақпарат, суреттер, аудио және бейне файлдар, сондай-ақ деректерді енгізудің басқа түрлері жасанды интелект жүйесімен өңдеуге қажетті деректерді енгізе алады. Бұл пайдаланушы-

ларға талдау және өңдеу үшін жүйеге ақпараттың әртүрлі түрлерін беруге мүмкіндік береді.

- Сұранысты өңдеу: веб-интерфейс пайдаланушы енгізген деректерді деректерді талдайтын және жауаптар немесе нәтижелер шығаратын жасанды интеллект моделін немесе жасанды интеллект алгоритмін өңдеуге жібереді. Бұл кезең пайдаланушының кірісін өңдеуді, деректерді талдау үшін машиналық оқыту модельдерін немесе басқа алгоритмдерді қолдануды және нәтижелерді қалыптастыруды қамтиды.

- Нәтижелерді визуализациялау: жасанды интеллект жүйесінен алынған нәтижелерді пайдаланушыларға графиктер, диаграммалар, мәтіндік жауаптар немесе кескіндер сияқты түсінікті және көрнекі түрде көрсетуге және көрсетуге болады. Бұл пайдаланушыларға нәтижелерді жақсы түсінуге және түсіндіруге мүмкіндік береді.

- Жасанды интеллект модельдерімен өзара әрекеттесу: пайдаланушылар жасанды интеллект модельдерін басқара алады, олардың жұмысын бастайды және тоқтатады, модель параметрлерін реттейді және жасанды интеллект жүйесімен өзара әрекеттесудің басқа операцияларын орындай алады. Бұған оқу процестерін басқару, модель параметрлерін өзгерту және олардың жұмысын бақылау мүмкіндігі кіреді.

- Диалогтық өзара әрекеттесу: веб-интерфейс жасанды интеллект агенттерімен немесе чатботтармен диалогтық өзара әрекеттесу мүмкіндігін қамтамасыз ете алады, бұл пайдаланушыларға сұрақтар қоюға, жауап алуға және табиғи тілде сөйлесуге мүмкіндік береді. Бұл пайдаланушыларға жасанды интеллект жүйесімен өзара әрекеттесуін жақсарту арқылы табиғи және интерактивті тәжірибе жасайды.

Графикалық пайдаланушы интерфейстерін құруға арналған құралдар:

- Bootstrap: HTML, CSS және JavaScript көмегімен веб-қосымшаларды әзірлеуге арналған ақысыз және ашық құралдар жиынтығы. Ол түймелер, пішін-

дер, модальды терезелер және т.б. дайын интерфейс компоненттерін ұсынады, бұл әдемі және жауап беретін интерфейстерді құруды жеңілдетеді.

- Gradio: машиналық оқытуға арналған интерактивті веб-интерфейстерді жылдам құруға арналған Python кітапханасы. Бұл көптеген дайын басқару элементтерін және нәтижелерді визуализациялау мүмкіндігін қамтамасыз ете отырып, аз күш жұмсай отырып, машиналық оқыту модельдері үшін пайдалану-

шы интерфейстерін жасауға мүмкіндік береді.

- Streamlit: деректерге негізделген интерактивті веб-қосымшаларды құруға арналған Python фреймворкі. Деректерді өзгерткен кезде интерфейсті автоматты түрде жаңартуды қамтамасыз ететін бірнеше код жолдары арқылы жылдам прототиптеуге және веб-қосымшаларды орналастыруға мүмкіндік береді.

- LangChain: жасанды интеллектке негізделген тілдік модельдерді құруға және орналастыруға арналған интеграцияланған даму ортасы. Ол тілдік модель-

дерді әртүрлі қолданбаларға, веб-интерфейстерге үйретуге, бағалауға және бірік-

тіруге арналған құралдарды ұсынады.

- Chainlit: интерактивті веб-қосымшаларды тез және оңай құруға арналған Streamlit негізіндегі Python кітапханасы. Ол жасанды интелектпен сөйлесу жүйе-

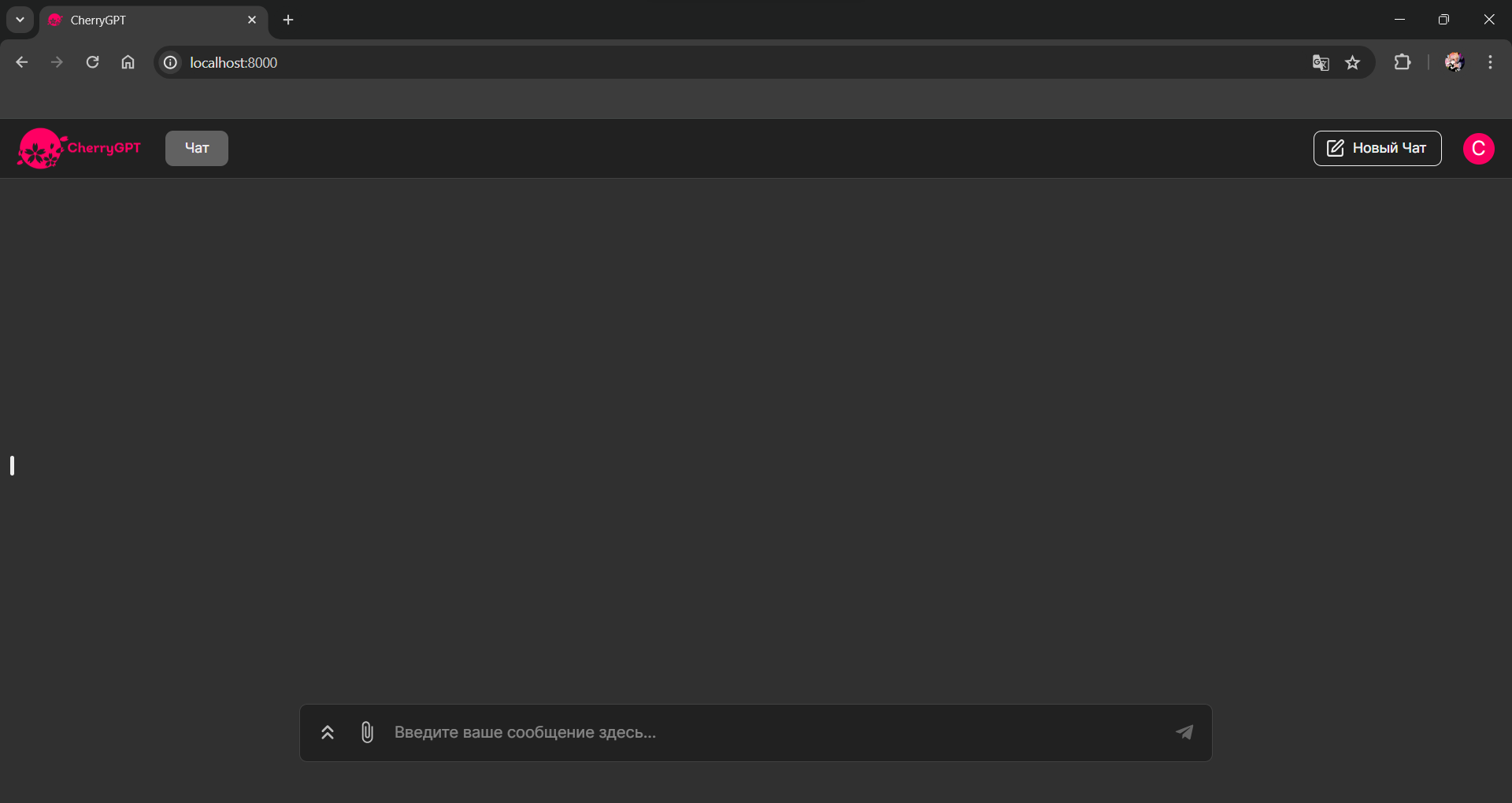
лерінің веб-интерфейстерін жасау үшін тартымды құралға айналдыратын көпте-

ген мүмкіндіктерге ие. [18]

Жасанды интеллект чат веб-интерфейсінің негізгі функциялары:

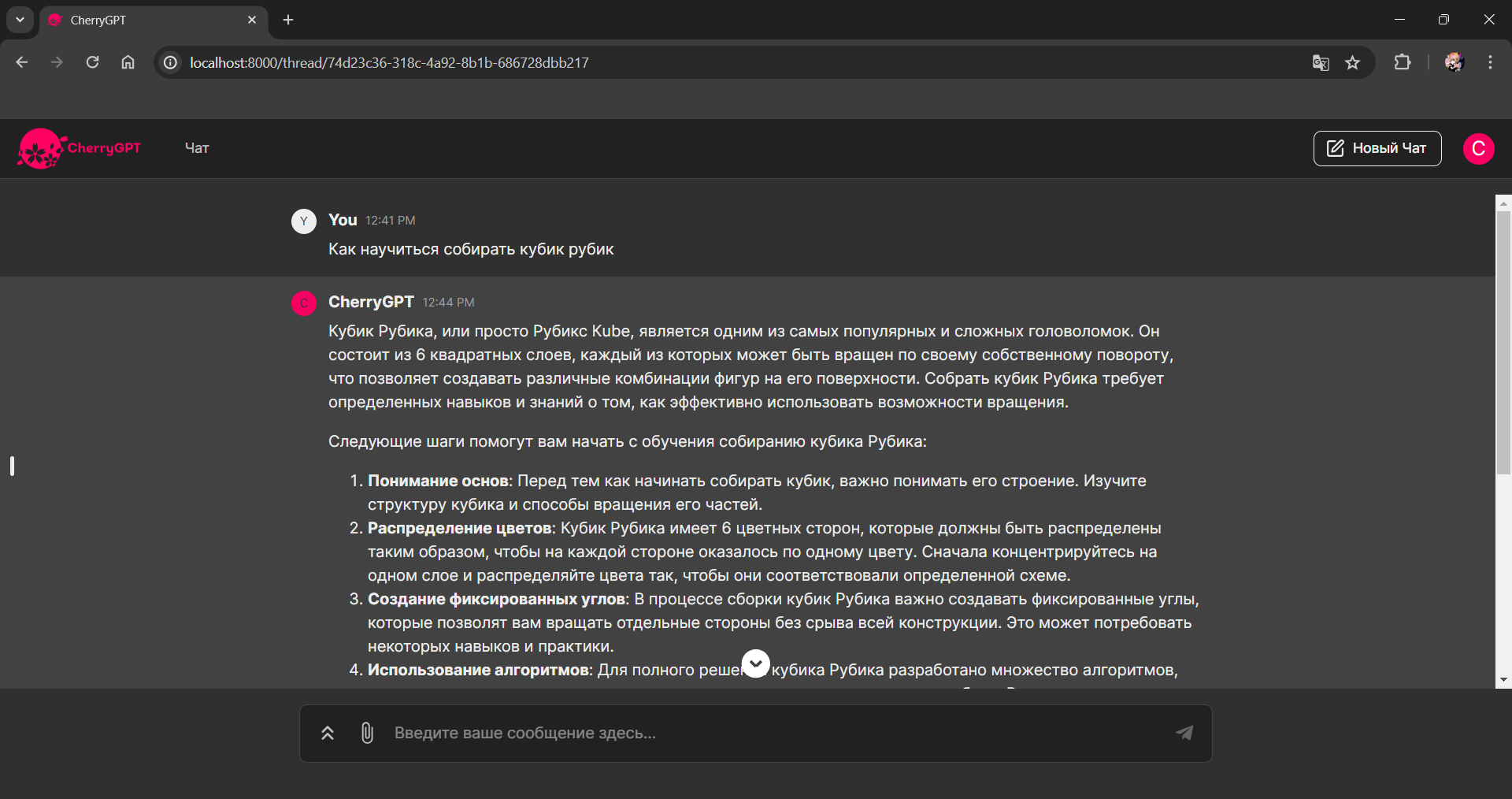
1. Мәтін енгізу аймағы: пайдаланушыларға чатботпен өзара әрекеттесудің ыңғайлы және икемді әдісін ұсынады. Бұл тек мәтіндік хабарламаларды енгізуге ғана емес, сонымен қатар суреттерді, аудио файлдарды және бейне файлдарды жүктеуге, жасанды интеллект жүйесімен байланыс мүмкіндіктерін кеңейтуге мүмкіндік береді. Сонымен қатар, енгізу аймағын енгізуді дұрыстығын тексеру, теру кезінде автоматты түрде кеңес беру және чатботпен өңдеуге жібермес бұрын деректерді тексеру және пішімдеу сияқты әртүрлі қосымша мүмкіндіктер-

ге теңшеуге болады. Бұл жасанды интеллект жүйесімен өзара әрекеттесу тиімділігін арттыра отырып, пайдаланушылар үшін ыңғайлы және интуитивті интерфейс құрады (Сурет 1.8.1).



Сурет 1.8.1 Жобаның веб-интерфейсі

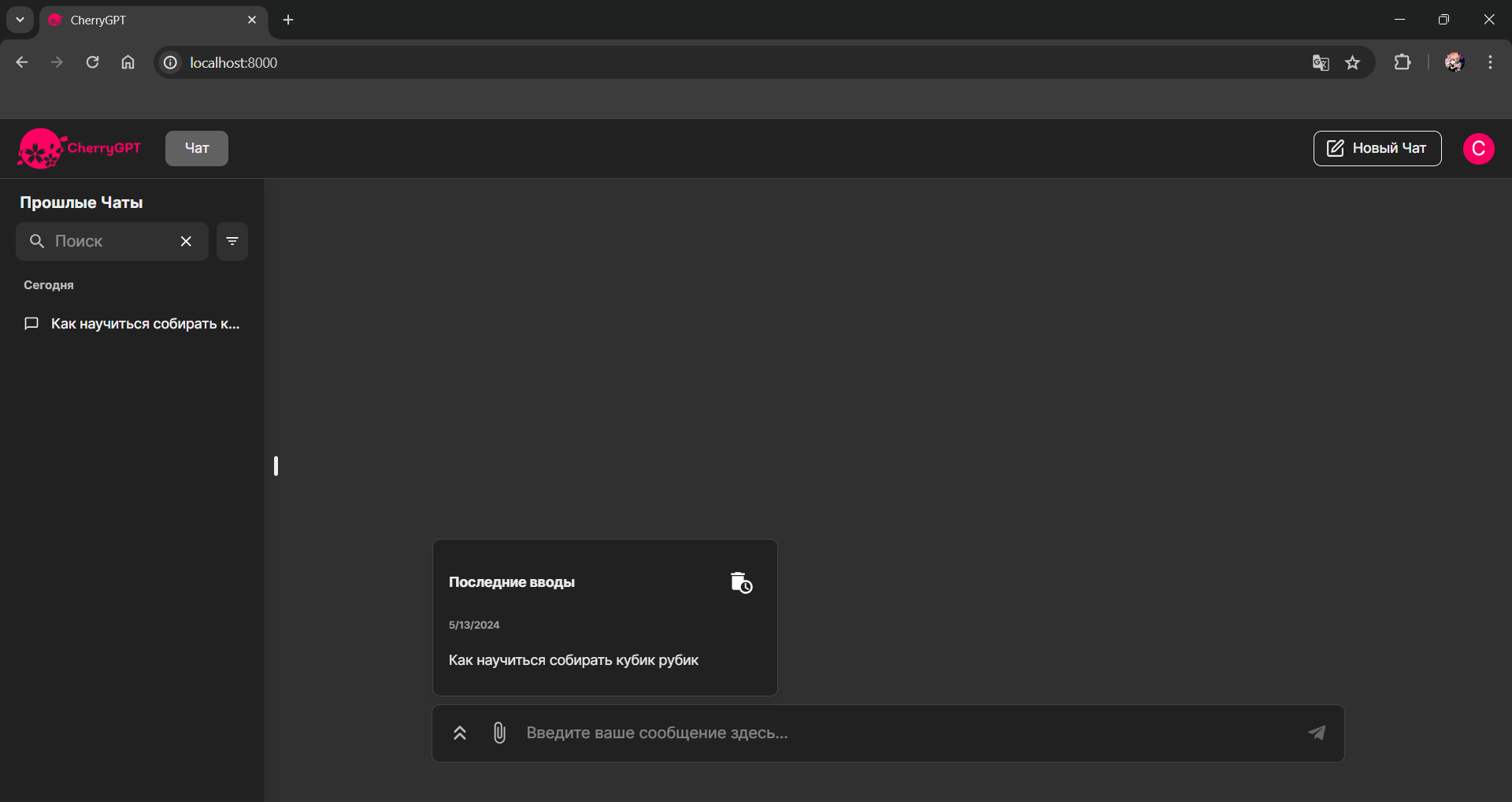
2. Жауап көрсету: пайдаланушының хабарламаларына жасанды интеллект жасаған жауаптарды көрсетуге мүмкіндік береді. Бұл мүмкіндік мәтін, кескіндер, аудио, бейне және интерактивті элементтер сияқты әртүрлі шығыс форматтарын қолдайды, бұл пайдаланушы тәжірибесін байытады және чатбот тәжірибесін жақсартады. Іс жүзінде сынап көрейік, мысалы, кубик рубикті қалай жинауға болатынын сұрайық. Мен кубик рубикті жинай аламын, сондықтан жасанды интеллект ақпаратының пайдалылығын тексере аламын (Сурет 1.8.2).



Сурет 1.8.2 Жасанды интеллекттің жауабы

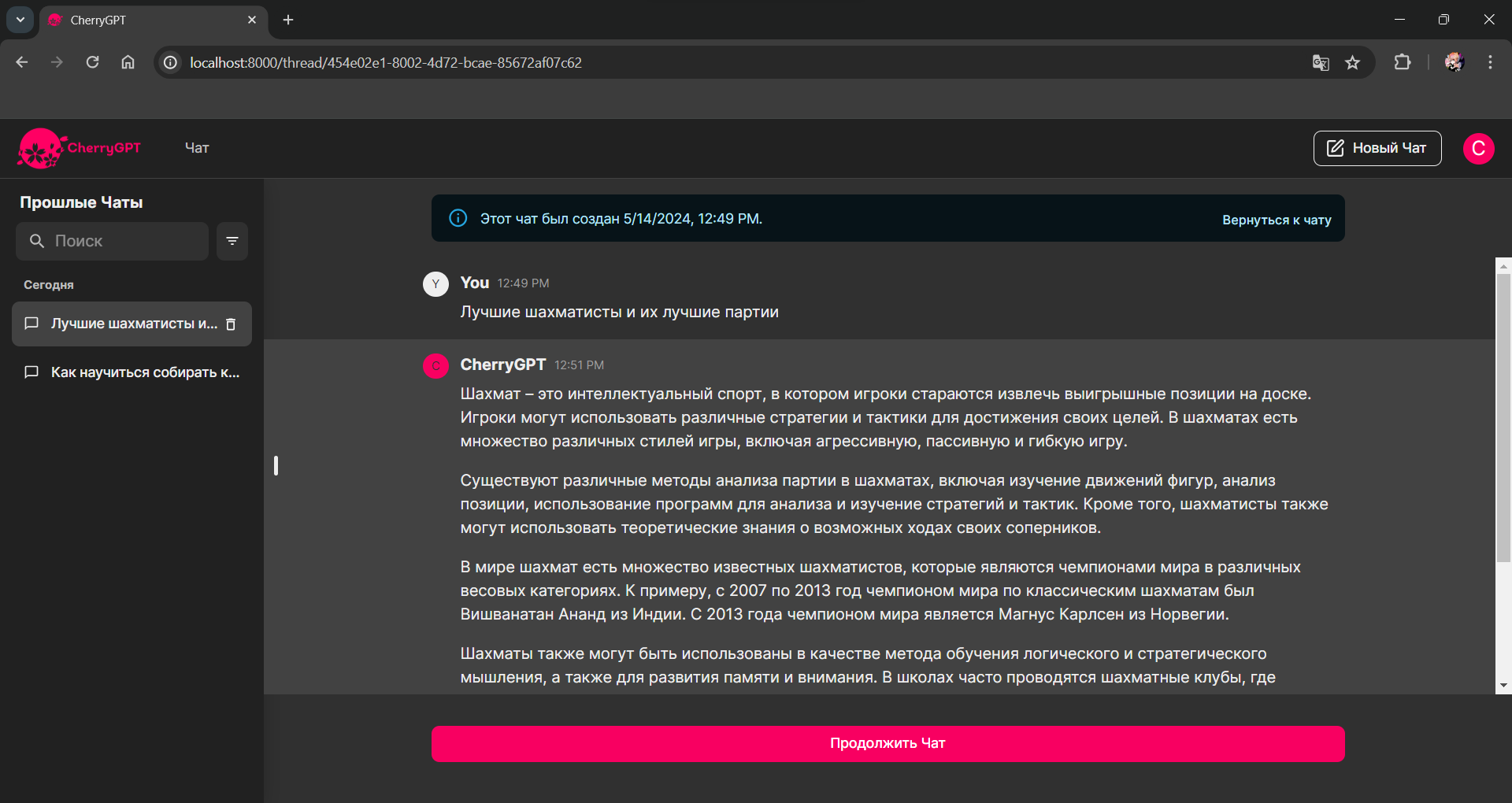
3. Диалог тарихы: пайдаланушының алдыңғы хабарламалары мен жасанды интеллект жүйесінің тарихын сақтайды және көрсетеді. Пайдаланушыларға диалог барысын көруге және жасанды интеллект жауаптарын бақылауға мүмкін-

дік береді. Сондай-ақ, пайдаланушыға диалог тарихын өшіруге немесе кейінірек пайдалану үшін сақтай алады (Сурет 1.8.3).



Сурет 1.8.3 Диалог тарихы мен енгізу тарихы

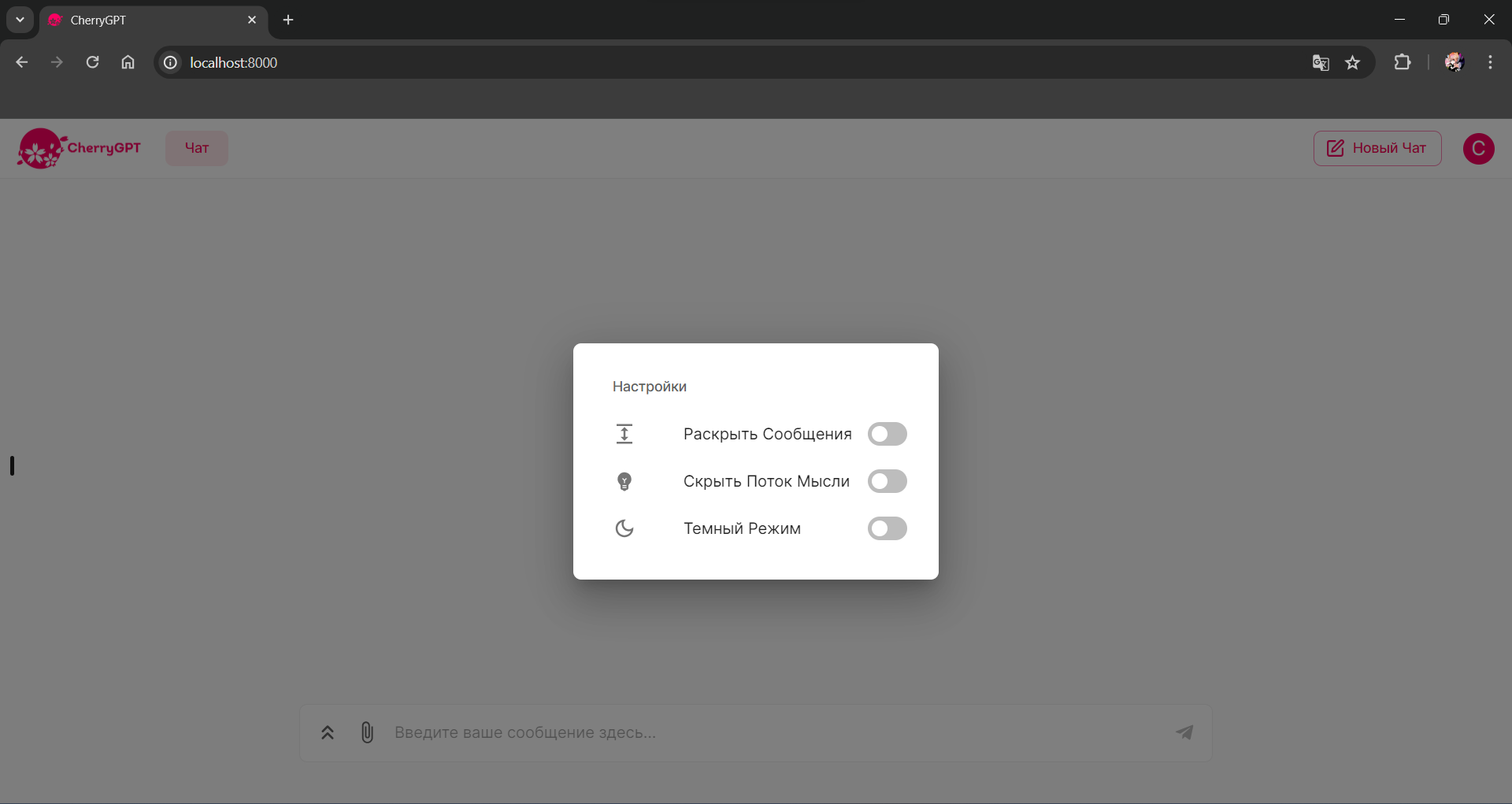
4. Диалогты басқару: диалогтың барысын басқару мүмкіндіктері ұсынады, кідірту, аяқтау және параметрлерді өзгерту. Ол батырмалар, мәзірлер немесе басқа басқару элементтері арқылы жүзеге асырылады. Пайдаланушылар кез келген уақытта диалогтарды тоқтата алады, содан кейін өзара әрекеттесудің үздіксіздігін мен дәйектілігін сақтай отырып, оны тоқтатылған жерден чатты жалғастыра алады (Сурет 1.8.4).



Сурет 1.8.4 Диалогты жалғастыру функциясы

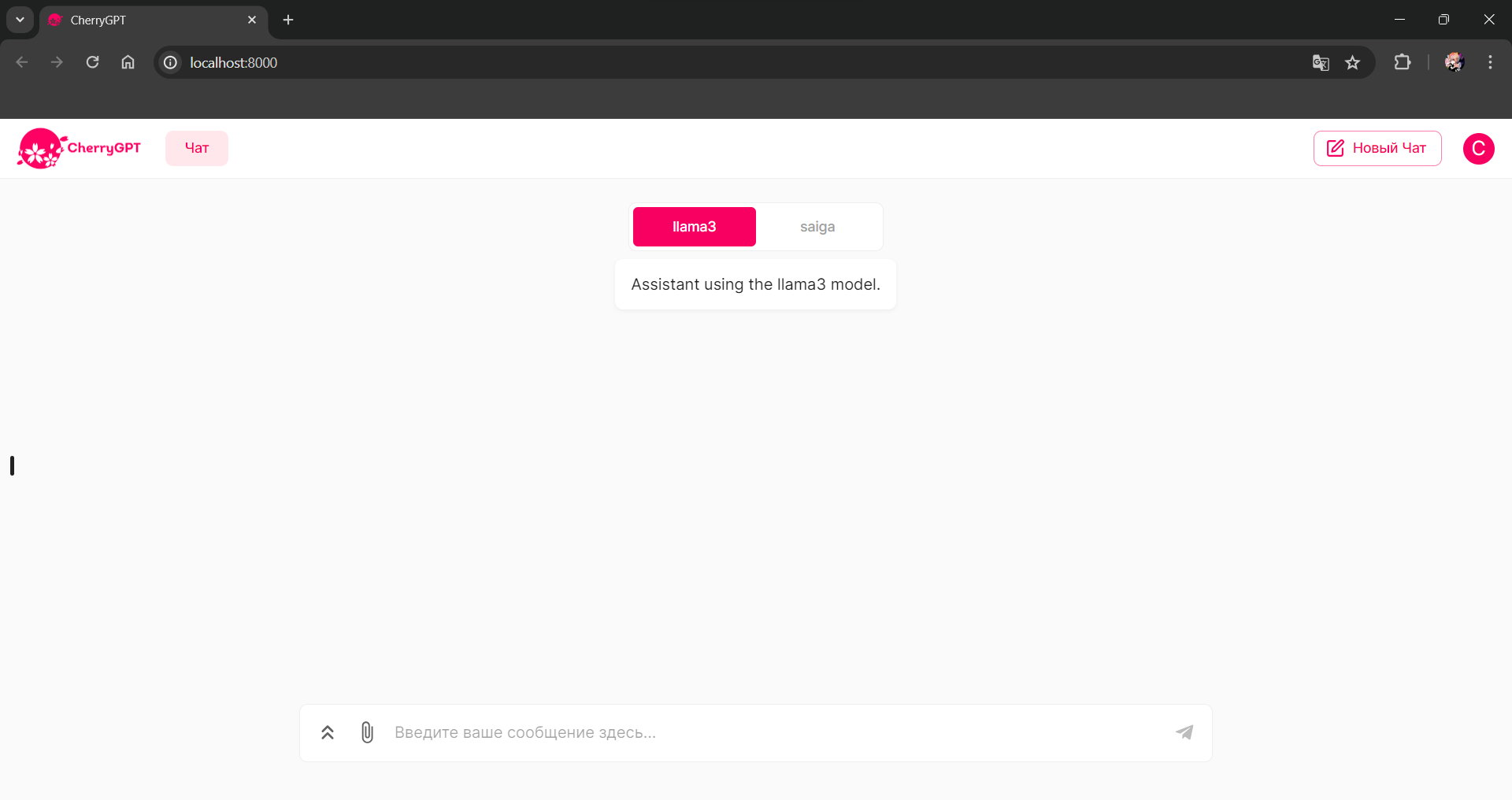
5. Персонализация: пайдаланушыларға түс схемасы, қаріптерді, орналасу-

ды және басқа дизайн элементтерін қоса, интерфейсті өз қалауына қарай реттеуге мүмкіндік береді. Параметрлер мәзірінде қараңғы режимі қосу, хабарламаларды ашу және ой ағынын жасыру мүмкіндіктері бар (Сурет 1.8.5).



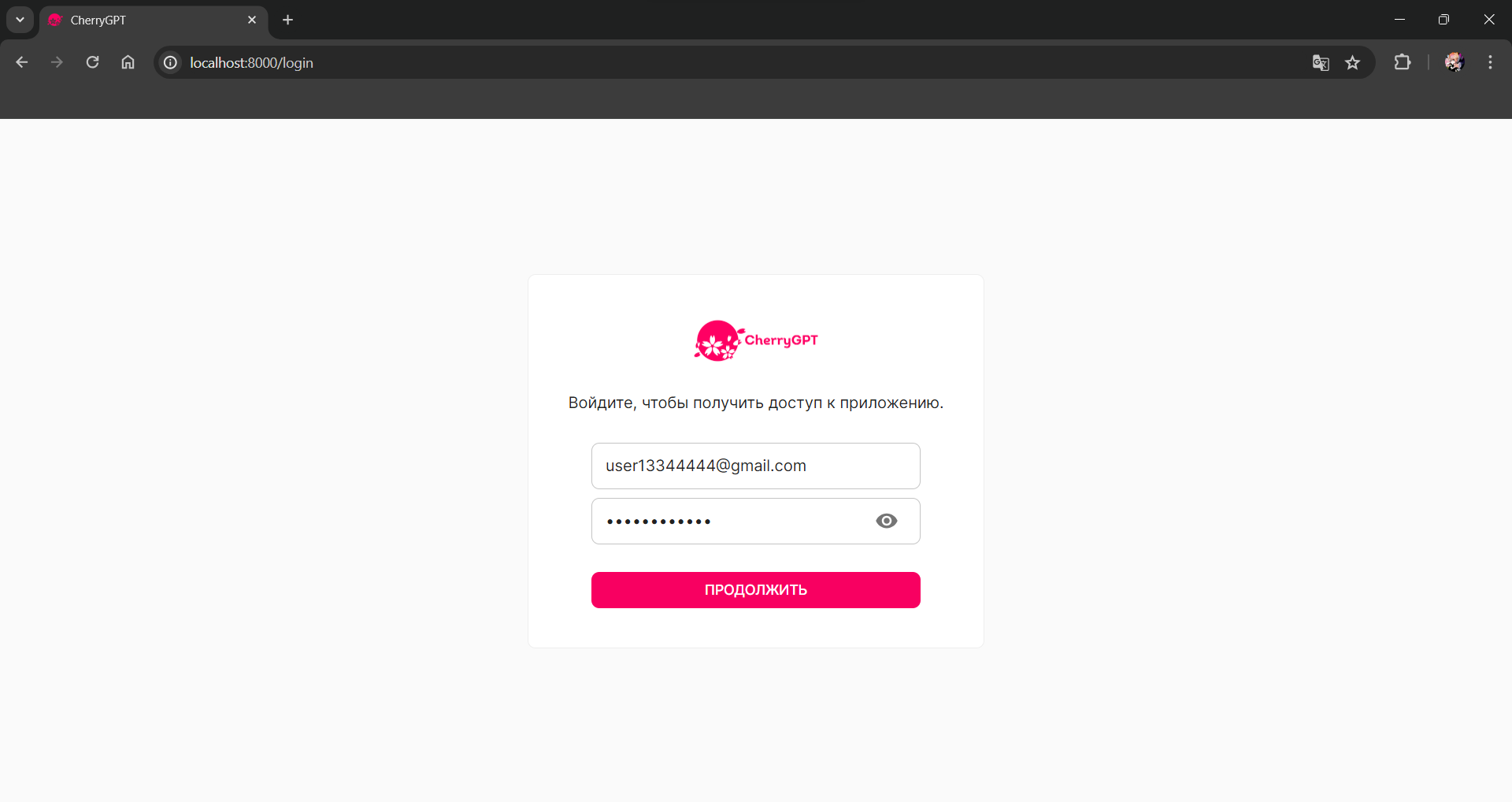
Сурет 1.8.5 Жобаның ақ темасы

6. Жасанды интеллект модельдерімен интеграция: Ollama, Hugging Face және басқа әртүрлі машиналық оқыту платформаларымен интеграция чаты үшін әртүрлі жасанды интеллект модельдерін пайдалануға мүмкіндік береді. Менің чат ботымда екі модель бар. Ollama-ның llama3 моделі мен Hugging Face-тің saiga моделі, олар чат профильдері ретінде жұмыс істейді. llama3 моделі жылдам жұмыс істейді, бірақ кейде ағылшын тілінде жауап береді, ал saiga моделінде ондай проблема жоқ, бірақ ол баяу жұмыс істейді (Сурет 1.8.6).



Сурет 1.8.6 Жасанды интеллект профильдері

7. Қауіпсіздік: пайдаланушы деректерін қорғауды қамтамасыз етеді. Бұған қосымшаға кіруге рұқсат бермес бұрын пайдаланушылардың түпнұсқалығын тексеретін аутентификация механизмдерін енгізу арқылы қол жеткізіледі. Бұл деректерге рұқсатсыз қол жеткізуден қорғауға және веб-интерфейс арқылы берілетін ақпараттың құпиялылығына кепілдік береді (Сурет 1.8.7).



Сурет 1.8.7 Авторизация мәзірі

2 ЭКОНОМИКАЛЫҚ БӨЛІМ

2.1 Жобаның экономикалық зерттеуі

Дипломдық жобаның тақырыбы: «Жасанды интеллект чаты».

Жұмыстың мақсаты: LLM модель негізінде Қазақстан пайдаланушылары үшін тиімді және ыңғайлы чат құру.

Бұл дипломдық жобада мен қазақ және орыс тілдерін табиғи тілде түсінуге қабілетті жасанды интеллект чатын құрдым.

Жобаның экономикалық бөлімі «Есептеу техникасы және бағдарламлық қамтамасыз ету» мамандығы мен «Жасанды интеллект» саласы үшін электронды әдебиет бойынша есептелді. Зерттеу барысында жобаның жұмысшылар үшін, материалдық, мұқтаждық және амортизациялық шығындары есептелді, олардың жалпы сметасы жүргізілді. Одан басқа, жобаның рентабельділігі мен тиімділігі оның табысы негізінде анализ жүргізілді.

Жасанды интеллект – IT саласындағы маңызды қадам. Ол жоғары сапалы қолдауды қамтамасыз ете отырып, адамның араласуын қажет етпейтін нақты уақыттағы жауаптар беру арқылы пайдаланушы тәжірибесін жақсарта алады.

Жасанды интеллект чаттары клиенттерге қолдау көрсету, маркетинг, білім беру, денсаулық сақтау, персоналды басқару, банк ici, электрондық коммерция, туризм және қонақжайлылық сияқты заманауи салаларда қолданылады.

2.2 Жұмысшы шығындары мен жұмысшылардың ЖТС-тары

Жобаны жасанды интеллект чат әзірлеушісі ретінде 1 адам жүзеге асырды. Әзірлеушінің жалақысы 192000 теңгені құрайды. Жұмыс беруші төлейтін шығындарды өлшеу үшін әлеуметтік аударымдар, әлеуметтік салық және жұмыс берушінің міндетті зейнетақы жарналарын есептеу керекпіз.

Әлеуметтік аудармалар формуласы (3.5%):

; (2.2)

Сонда, әлеуметтік аударымдар:

; (2.2.1)

Әлеуметтік салық формуласы (9.5%):

; (2.2.2)

Сонда, әлеуметтік салық:

; (2.2.3)

Жұмыс берушінің міндетті зейнетақы жарналар формуласы (1.5%):

; (2.2.4)

Сонда, ЖМЗЖ:

; (2.2.5)

Жұмысшы үшін барлық шығындарды келесі формуламен есептейміз:

; (2.2.6)

мұндағы: – жұмысшы үшін шығындар;

– жұмыс мерзімі.

Уақыт мерзімі 30 күнге тең, ол шамамен 1 толық айға тең, cонда:

; (2.2.7)

Енді, жұмысшының жеке салық салыну шамалары. Олар міндетті зейнетақы жарналарынан, міндетті әлеуметтік медициналық сақтандырудан және жеке табыс салаларынан тұрады.

Міндетті зейнейтақы жарналары (жалақыдан 10%):

; (2.2.8)

Міндетті әлеуметтік медициналық сақтандыру (жалақыдан 2%):

; (2.2.9)

Жеке табыс салығын есептесек, ол келесі формуламен есептеледі:

; (2.2.10)

Қазіргі АЕК 3692 теңгеге тең. Жеке табыс салығы 14 АЕК-ке, яғни 51688 теңгеге тең шаманы қолданады. Сонда, жеке табыс салығы:

; (2.2.11)

Сонда, таза жалақы:

; (2.2.12)

Жеке жұмысшы 192000 теңге жалақысынан салыққа кететін бөлігі 34767.2 теңгеге тең. Жалпы, жұмысшы үшін шығындар 210931.2 теңгеге тең.

2.3 Материалдар мен жасақтама шығындары

Кесте-2.3 Материалдар және оған кеткен шығындар

|  |  |  |  |
| --- | --- | --- | --- |
| Материал | Саны | 1 дана бағасы, тг | Соммасы, тг. |
| Процессор: AMD Ryzen 5 5600G | 1 | 99990 | 99990 |
| Видеокарта: NVIDIA GeForce RTX 2060 6GB | 1 | 149990 | 149990 |
| Жедел жад: DDR4-3200 16GB (2x8GB) | 2 | 12490 | 24980 |
| SSD диск: Samsung 870 QVO 512GB | 1 | 19990 | 19990 |
| Қуат көзі: Corsair RM 600W | 1 | 24990 | 24990 |
| Жасанды интеллект моделін оқыту | 1 | 9990 | 9990 |
| Барлығы |  |  | 329930 |

Кестедегі шығындар тек тікелей жұмысқа қатысты шығындарды ескереді. Материалдық шығындар 329930 теңгені құрайды.

2.4 Электр энергиясының шығындары

Электр энергиясына арналған шығындар келесі формуламен есептеледі:

; (2.4)

мұндағы: – бір жұмыс күнінің электр шығыны;

– бір сағатта тұтынылатын қуат (кВт);

– жұмыс уақыты;

– бір сағаттағы электр энергиясының тарифы.

Электр энергия тарифы шамамен 26 теңгеге тең, ал жобаға құртылған қуат шамамен 600 ваттқа тең. Күндізгі жұмыс уақыты 8 сағатқа тең. Сонда:

; (2.4.1)

Егер, толық жұмыс мерзімінің электр шығындарын санасақ, бізге бір күндік электр шығынын барлық жұмыс күндеріне көбейту керек:

; (2.4.2)

мұндағы: – толық электр шығыны;

мұндағы: – жұмыс режимінің мерзімі.

Жоба 30 күн бойы жасалған, сонда толық электр шығыны:

; (2.4.3)

Толық электр шығыны 3744 теңгені құрайды.

2.5 Амортизация шығындары

Кесте-2.5 Амортизацияға ұшырайтын шығындар

|  |  |  |  |
| --- | --- | --- | --- |
| Материал | Саны | 1 дана бағасы, тг | Соммасы, тг. |
| Процессор: AMD Ryzen 5 5600G | 1 | 99990 | 99990 |
| Видеокарта: NVIDIA GeForce RTX 2060 6GB | 1 | 149990 | 149990 |
| Барлығы |  |  | 249980 |

Айлық амортизация шығындары келесі формула бойынша есептеледі:

; (2.5)

мұндағы: – амортизацияның бір айлық шығыны;

– амортизацияланатын баға;

– амортизацияның жылдық пайызы.

Компьютерлер үшін жылдық амортизацияның нормасы 30%-ды құрайды. Сонда, амортизацияның бір айлық шығыны:

; (2.5.1)

Жалпы амортизация шығыны – 6249.5 теңге.

2.6 Жасанды интеллект чатына құруға кеткен жалпы шығындар

Бүкіл жобаны құруға кеткен шығындар соммасы:

; (2.6)

Жасанды интеллект чаты жалпы шығындары 550854.7 теңгені құрайды.

2.7 Табыс пен рентабельдікті анықтау

Жасанды интеллект чатының табыс көлемін келесі формула арқылы шығарса болады:

; (2.7)

мұндағы: – табыс көлемі;

– мекеменің рентабельдік нормативі;

Рентабельдік норматив 25%-ға тең. Жоба шығынын қолданып, есептейміз:

; (2.7.1)

Енді жасанды интеллект чатының жобалық бағасын есептейміз, ол үшін келесі формула қолданылады:

; (2.7.2)

Сонда, жоба бағасы:

; (2.7.3)

Жобаның толық бағасы 688568.4 теңгені құрайды.

2.8 Жобаға экономикалық анализ жасау

Жасанды интеллектті дамыту және енгізу экономикаға және әртүрлі салаларға терең әсер ететін қазіргі заманның ең маңызды технологиялық жетістіктерінің бірі болып табылады. Жасанды интеллект чатын енгізудің экономикалық аспектілерін талдау осы технологияларды біріктірудегі ұйымдар-

дың негізгі артықшылықтары мен қиындықтарын анықтады.

Жобаның функционалдығы мен қолданылуы кез-келген пайдаланушының мақсаттары мен қажеттіліктеріне байланысты. Жасанды интеллект чаттарын тұтынушыларға қолдау көрсетуге, жекелендірілген ұсыныстар беруге, бизнес-процестерді автоматтандыру және қызмет көрсету сапасын жақсарту үшін пайдалануға болады.

Жасанды интеллект чаты орыс және қазақ тілдерін қолдайтындықтан, ол көптеген пайдаланушылар үшін қол жетімді және коммерциядан бастап мемлекеттік қызметтерге дейін әртүрлі салаларда пайдалануға болады. Жоба жасанды интеллектке негізделгендіктен, оның дамыту және қолдану перcпекти-

валары тек техникалық талаптармен шектеледі, бұл болашақта оны одан әрі дамытуға және пайдаланушылардың нақты қажеттіліктеріне бейімделуге кең мүмкіндіктер ашады.

# 3 ЕҢБЕКТІ ҚОРҒАУ БӨЛІМІ

## 3.1 Еңбек қауіпсіздігі

Еңбек қауіпсіздігі – бұл жұмысшыларға қауіпті мен зиянды өндірістік фак-торлардың әсерін болдырмайтын және олар үшін компенсация жүргізетін еңбек жағдайларының күйі.

Қазақстан Республикасының Еңбек кодексінің 1-бөлім («Жалпы ере-желер»), 1-тарау («Негізгі ережелер»), 1-бабының («Негізгі ережелер»), 32-пунк-тінде келтірілген анықтама бойынша, еңбекті қорғау – құқықтық, әлеуметтік, экономикалық, техникалық, санитариялық, профилактикалық, оңалту, емдеу, қа-уіпсіздік және өзге де іс-шаралар мен құралдарды қамтитын, еңбек қызметі про-цессінде жұмыскерлердің өмірі, құқығы және денсаулығының қауіпсіздігін қам-тамасыз ету жүйесі.

Еңбекті қорғау өз мәні бойынша медициналық тексерулер жүргізу, қызмет-керлерді белгіленген нормалар бойынша арнайы киіммен қамтамасыз ету, зиян-ды еңбек жағдайлары үшін жеңілдіктер мен өтемақылар беру, белгілі уақыт ара-лығында оларға демалыс беру сияқты жұмыс берушілер орындауға міндетті іс-шаралар жиынтығын білдіреді.

Халықаралық еңбек ұйымының мәліметі бойынша, жыл сайын өндірістегі жазатайым оқиғалар мен кәсіптік аурулардың салдарынан 2.3 миллион адам қаза табады, оның 317 мыңға жуығы өліммен аяқталған жазатайым оқиғалар. Қазақ-стан Республикадағы еңбекті қорғаудың мәселелеріне кәсіптік жарақаттану-лардың қолайсыз жоғары деңгейде тұруы жатады. Еңбекке қабілетті жастағы халықтың өлім-жітім деңгейі ЕО-ға қарағанда 4.5 есе және дамушы елдерге қара-ғанда 1.5 есе жоғарырақ. Халықаралық еңбек ұйымының мәліметтеріне сүйен-сек, Қазақстан Республикасында жыл сайын еңбек жағдайларынан 190 мың адам қайтыс болады.

Дипломдық жоба еңбек қорғау келесі сұрақтарды қарастырады:

1. Еңбек қауіпсіздігі.

2. Еңбек жағдайлары, яғни өндіріс санитария талаптары.

3. Техникалық қауіпсіздіктің талаптары.

4. Өрт қауіпсіздігінің талаптары.

### 3.1.1 Еңбек режимі

Дербес компьютермен жұмыс істеу кезінде дұрыс жұмыс пен демалыс ре-жимін сақтау өте маңызды рөл атқарады. Әйтпесе, қызметкерлерде жұмысқа қа-нағаттанбау, бас ауруы, ашуланшақтық, ұйқының бұзылуы, шаршау және көзде, төменгі арқада, мойын мен қолда ауырсыну шағымдары пайда болатын көру ап-параттарының айтарлықтай кернеуі байқалады.

Белгіленетін демалыс уақыттары жұмыс категориясы, жұмыс мөлшері жә-не жалпы жұмыс уақыттарына тәуелді.

Кесте-3.1.1 ДЭЕМ-мен жұмыс кезіндегі демалыс уақыттары

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Жұмыс категориясы | ВДТ-ны қолданғандағы жұмыс мөлшері | | | Толық белгіленетін жұмыс уақыттары, мин. | |
|  | А тобы, таңба саны | Б тобы, таңба саны | В тобы, сағат саны | 8-сағат режимде | 12-сағат режимде |
| I | 0-20000 | 0-15 000 | 0-2 | 30 | 70 |
| II | 20000-40000 | 15000-30000 | 2-4 | 50 | 90 |
| III | 40000-60000 | 30000-40000 | 4-6 | 70 | 120 |

Компьютерді пайдалануға байланысты барлық жұмыс үш топқа бөлінеді:

А тобы: экраннан ақпаратты оқу бойынша жұмыс;

Б тобы: ақпаратты енгізу бойынша жұмыс;

В тобы: компьютермен диалог режиміндегі жұмыс.

Үзілістердің тиімділігі өндірістік гимнастикамен немесе ыңғайлы жұмсақ жиһазбен, аквариуммен, жасыл аймақпен және т.б. персоналдың демалуына ар-налған арнайы бөлмені ұйымдастырумен үйлескенде артады.

## 3.2 Өндіріс санитария және еңбек гигиенасы

Өндірістік санитария – бұл жұмысшыларға зиянды өндірістік фактор-лардың әсерін болдырмайтын немесе азайтатын ұйымдастырушылық шаралар мен техникалық құралдар жүйесі. Адам денсаулығына көптеген түрлі факторлар әсер ете алады.

Негізгі қауіпті және зиянды өндірістік факторлар:

- жұмыс аймағының ауасының шаңдануы, газдануының және ластануының жоғарылауы;

- жұмыс аймағының ауа температурасының жоғарылауы немесе төмендеуі;

- жұмыс аймағындағы ауаның ылғалдылығы;

- шу деңгейінің жоғарылауы;

- діріл деңгейінің жоғарылауы;

- жұмыс аймағының жеткіліксіз жарықтандырылуы.

Еңбек гигиенасы, кәсіби гигиена – еңбектің және айналадағы өндірістік ортаның адам организміне тигізетін әсерін зерттейтін, еңбек етуге қолайлы жағ-дай туғызу, кәсіби ауруларға жол бермеу шараларын қарастыратын гигиена саласы.

Таза жұмыс кеңістігі қызметкерлердің денсаулығы мен әл-ауқатын жақ-сартуға ықпал етеді. Шаң, кір және қоқыстар аллергия мен тыныс алу қиын-дығын күшейтіп, өнімділіктің төмендеуіне және ауру күндерінің көбеюіне әкелуі мүмкін.

Жұмыс орындарын ретке келтіру және тәртіпсіздіктерден тазарту қау-іпсіздік тәуекелдерін азайтуға көмектеседі. Таза ұйымдастырылған ортада рұқ-сатсыз кіруді немесе араласуды анықтауға оңайырақ болады.

IT мамандар үшін жабдыққа техникалық қызмет көрсету де маңызды. Тұрақты тазалау компьютерлер, пернетақталар және мониторлар сияқты жаб-дықтарының қызмет ету мерзімін ұзартуға көмектеседі. Шаң жүйелі түрде таза-ланбаса, олар ақауларға әкелуі мүмкін.

Желілермен жұмыс істеу кезінде кабельді дұрыс басқару маңызды, олар шатасу және сүріну қаупін болдыру мүмкін, оны болдырмау үшін кабельдерді реттелген түрде ұстау керек. Бұған кабельдік байланыстар, қыстырғыштар және кабельдік науалар сияқты кабельді басқару және жүйелеу құралдары көмектесе алады.

Тазалық деректердің қауіпсіздігі үшін де маңызды. Ретсіз жұмыс орындары құпия ақпараттың орынсыз орналасу немесе рұқсат етілмеген адам-дардың оған қол жеткізу қаупін арттырады. Үстелдерді құпия құжаттардан таза ұстау және қараусыз қалған кезде экрандарды құлыптау деректердің құпия-лылығын сақтауға көмектеседі.

3.2.1 Микроклимат

Микроклимат – бұл бөлмедегі метеорологиялық жағдайлардың кешені: температура, салыстырмалы ылғалдылық, аэроиондар саны, ауа алмасу, ауа қоз-ғалысының жылдамдығы, ауадағы бөлшектердің (шаңның) мөлшері, денсаулық-қа жағымды иістердің болуы (ароматерапия) және т.б.

SN-245-71 санитарлық стандарттары қолайлы жағдайлар жасайтын микро-климат параметрлерінің мәндерін орнатады. Бұл нормалар мезгіл уақытына және өндірістік объектінің сипатына байланысты белгіленеді.

Кесте-3.2 Компьютерлі бөлмелер үшін микроклимат параметрлері

|  |  |  |
| --- | --- | --- |
| Микроклимат параметрі | Жыл мезгілі | |
|  | Жылы | Суық |
| Бөлмедегі ауа температурысы | 22-25 °C | 20-22 °С |
| Салыстырмалы ылғалдылық | 30-60% | 30-45% |
| Ауа қозғалысының жылдамдығы | 0.25 м/с-дан кем | 0.15 м/с-дан кем |

Бөлмеде қолайлы микроклимат құру үшін арнайы жүйелер қолданылады: жылы мезгілде – желдету және кондиционерлеу жүйелері.

Суық мезгілде – жылыту жүйелері: газ, ағаш немесе көмір пештері, орта-лықтандырылған сумен жылыту жүйесі мен электр жылытқыштар.

Температураны басқару кезінде қолданылатын өлшем бірліктерді мұқият бақылау керек, себебі әртүрлі термометрлер сәйкесінше Цельсий (°C), Кельвин (°K) немесе Фаренгейт (°F) шамаларын қолданады.

СИ (ағыл. System International – «Халықаралық жүйе») жүйесіндегі Цельсий суға қатысты температураны, ал Кельвин атомдарға қатысты температураны өлшесе, империялық жүйедегі Фаренгейт адамға қатысты температурына өлшейді, сондықтан өндіріс кезінде не ылғалдылықты түсінуге көмектесетін Цельсий шамасы, не жұмысшы сезімін сипаттайтын Фаренгейт шамасы қолданылады.

Есептеу орталықтарының қызметкерлері орналастырылатын үй-жайлар-дың көлемі бір мезгілде ауысымда жұмыс істейтіндердің ең көп санын ескере отырып, 19,5м3/адамнан кем болмауы керек.

Кесте-3.2.1 Компьютерлерлік бөлмелерге таза ауаны беру нормалары

|  |  |
| --- | --- |
| Бөлменің сипаттама | Бір адам үшін таза ауаның көлемдік шығыны, м3/сағатына |
| Бір адамға 20 м3-қа дейін | 30-дан кем емес |
| Бір адамға 20-40 м3 | 20-дан кем емес |
| Бір адамға 40 м3-тан астам | Табиғи желдету |
| Терезелері мен жарық шамдары жоқ үй-жай | 60-тан кем емес |

### 3.2.2 Жарықтандыру

Өндірістік жарықтандыру – бұл өндірістің әртүрлі міндеттерін орындау үшін жарықтандырудың оңтайлы жағдайларын қамтамасыз ету мақсатында өндірістік және өндірістік үй-жайларда қолданылатын арнайы жобаланған жарықтандыру жүйесі.

Дұрыс жобаланған және орындалған өндірістік жарықтандыру визуалды еңбек жағдайларын жақсартады, шаршауды азайтады, еңбек өнімділігін арт-тырады, жұмыс ортасына пайдалы әсер етеді, жұмысшыға жағымды пси-хологиялық әсер етеді, қауіпсіздігін арттырады және қатер-жарақаттарды азай-тады.

Жеткіліксіз жарық көздің шаршауына әкеледі, зейінді әлсіретеді және ерте шаршаудың пайда болуына әкеледі. Тым жарқын жарық көздің соқырлығын, ті-тіркенуін және ауырсынуын тудырады. Жұмыс орнындағы жарықтың дұрыс емес бағыты жұмысшының қатал көлеңкелерін, жарқылын және бағытының бұзылуын тудыруы мүмкін. Осы себептердің барлығы жазатайым оқиғаларға не-месе кәсіптік ауруларға әкелуі мүмкін, сондықтан жарықтандыруды дұрыс есеп-теу өте маңызды.

Жарықтандырудың үш түрі бар – табиғи, жасанды және аралас.

Табиғи жарықтандыру – үй-жайларды сыртқы қоршау құрылымдарындағы жарық саңылаулары арқылы еніп, күндізгі жарықпен жарықтандыру. Табиғи жарықтандыру күннің мезгіліне, жыл мезгіліне, аймақтың табиғатына және бас-қа да бірқатар факторларға байланысты өзгереді.

Жасанды жарықтандыру – түнде және күндізгі уақытта табиғи жарық коэффициентінің нормаланған мәндерін қамтамасыз ету мүмкін болмаған кезде (мысалы, бұлтты ауа-райы кезінде) қолданылатын жарықтандыру түрі. Жасанды жарық жұмыстық, апаттық жағдайлық, эвакуациялық және қауіпсіздік болып бө-лінеді.

Аралас жарықтандыру – табиғи жарық жеткіліксіз болып, жасанды жарық-пен толықтырылатын жарықтандыру.

SNiP II-4-79 сәйкес есептеу орталықтарының үй-жайларында аралас жа-рықтандыру жүйесін қолдану қажет.

Көру дәлдігі жоғары санаттағы жұмыстарды орындау кезінде (кемсіту объектісінің ең кіші өлшемі 0.3...0.5 мм) табиғи жарықтандыру коэффициентінің мәні (ТЖК) кемінде 1,5%, ал визуалды жұмыс үшін орташа дәлдік (кемсітушілік объектісінің ең кіші мөлшері 0.5...1.0 мм) ТЖК кем дегенде 1.0% болуы керек. Жасанды жарықтандыру көздері ретінде әдетте LB немесе DRL типті люминес-центті лампалар қолданылады, олар жұмыс беттерінің үстінде біркелкі орнала-суы керек шамдарға біріктіріледі.

Компьютерлер орнатылған бөлмелердегі жарықтандыруға қойылатын та-лаптар:

- дәлдігі жоғары визуалды жұмысты орындау кезінде жалпы жарықтанды-ру 300 лк, ал аралас жарықтандыру 750 лк болуы керек;

- орташа дәлдіктегі жұмыстарды орындау кезіндегі ұқсас талаптар – сәйке-сінше 200 және 300 лк.

ЭЕМ қолданушының көз алдындағы жарықтылық біркелкі емес таралу-ының шектеу қажет, бұл кезде жұмыс беттері жарықтылықтың арасындағы қатынас (үстел:экран) 3:1-5:1, ал жұмыс беттері мен қабырға беттері және қон-дырғылар арасындағы 10:1-ден артық болмауы шарт.

Әкімшілік-қоғамдық жэне өндірістік орындарда шағылысатын жарықтан-дыратын құралдар ретінде күші 250 Вт дейін металл-галогендік лампаларды қол-дануға рұқсат етіледі.

ЭЕМ орналасқан орындарды жарықтандыру үшін пульсация коэффици-ентін төмендетуге арналған жоғары жиілікті реттегіш аппараттармен жаб-дықталған айналы торлы ЛП036 серияның жылы түсті люменисцентті шамдарын қолданған дұрыс.

Жұмыс орнына қажетті жарықтандыруды қамтамасыз ету үшін бізге шам-дардың саны мен түрлеріне негізделген есептеу жүргізу керек.

Біріншіден, бөлменің жарық ағымын келесі формуламен анықтаймыз:

; (3.2)

мұндағы: – жарық ағыны;

– нормаланған минималды жарықтандыру;

– жарық ағынының төмендеуін ескеретін қауіпсіздік коэффиценті;

– бөлменің ауданы;

– орташа жарықтандырудың минимумға қатынасы;

– пайдалану коэффиценті.

ДЭЕМ-мен жұмыс үшін минималды жарықтандыру шамамен 300 люкске тең. Қауіпсіздік коэффиценті, біздің жағдайда, шамамен 1.5-ке тең. Бөлменің ау-даны 4x3 метрге, яғни 12 м2-қа тең. Орташа жарықтандырудың минимумға қаты-насы 1.1-ге тең шамамен қабылданды. Ал пайдалану коэффицентін өлшеу үшін бізге бөлменің коэффицентін табу керек, ол келесі формуламен есептеледі:

; (3.2.1)

мұндағы: – бөлме коэффиценті;

– бөлме биіктігі;

– бөлме ұзындығы;

– бөлме ені.

Бөлменің биіктігі стандартты 3 метрге тең, ал ұзындық пен ені жоғарыда жазылғандай сәйкесінше 4 пен 3 метрлерге тең. Сонда, бөлме коэффиценті:

; (3.2.2)

Енді, төбенің, қабырғалардың және еденннің шағылу коэффиценттерін са-най отырып (50%, 30%, 10%), пайдалану коэффиценті тұрақты 0.33-ге тең екенін анықтаймыз. Бұл шамаларды ескере отырып, енді жарық ағының табамыз:

; (3.2.3)

Енді, керек шамдар санын табу үшін, жарық ағымын шамдар жарығына бөлеміз. Жарықтандыруға 3000 Люменға тең жылы шамдар қолданылды. Сонда:

; (3.2.4)

Сонда, комфорты жарықтандыру үшін бізге 6 шам қажет.

3.3 Қауіпсіздік техникасы

Кез-келген компьютермен жұмыс істеген кезде міндетті түрде техника қауіпсіздігін сақтауы тиіс.

Жұмысқа дейінгі талаптар:

- аппаратураны жұмысқа дейін алдын-ала тексеру керек, егер артық дыбыстар немесе перифириялық құрылғылар өзімен-өзі өшіп қалу жағдайлары болса, ол туралы хабарландыру қажет;

- көздерді қорғайтын көзілдіріктерді (көру қабілеті нашар болмаса да) қолданған жөн;

- аппаратураны қажет рұқсатынсыз мүлдем қосуға болмайды;

- жұмыс орнының тазалығын қамтамасыз ету керек;

- монитордың көру бұрышын, қашықтығын және жалпы көру ыңғайлы-лығын қадағалау;

- тінтуірдің (компьютерлік тышқанның) қолдану бұрышын ыңғайлық бой-ынша түзеу.

Жұмыс кезіндегі компьютерде отыру талаптары:

- қолданушы мен компьютер арасында 60-70 см қашықтық болу керек;

- омыртқа тік ұсталу керек, қолданушы бүкшемей және еңкеймей отыру керек;

- егер қолданушы жоғары шарттармен экранды нашар көрсе, арнайы көзіл-дірік немесе линза қолдану керек;

- әрбір 20 минут жұмыс үшін 5 минут демалған жөн;

- аппаратураны абайлап қолдану керек, мысалы, пернетақта мен тінтуірді ақырын басу, монитор мен жүйелік блокты ұрмау;

- жеке гигиенаның бүкіл жұмыс кезеңдері үшін толық сақталуын қадаға-лау;

- байланыстағы сым жүйелерінің айырғыштарына тиіспеу;

- бөгде заттарды үстелдің үстіне қоймау;

- су қолмен және дымқыл киіммен жұмыс істемеу;

- аппаратураның жұмыс кезінде шыгып жатқан ақауларын өздігінен дұрыс-тамау, керек маманды шақыру;

- егер пернелердің кемінде біреуісі жұмыс істемесе, пернетақтаны ауыс-тыру шараларын жүргізу.

Жұмыстан кейінгі талаптар:

- аппаратураның жұмыстан кейінгі күйін тексеру керек, келесі күнге дей-інгі жұмысқа қабілетін қамтамасыз ету қажет;

- компьютердің өшуін қадағалау қажет;

- түнге қалтырылған техниканың өрт пен электр қауіпсіздігін қадағалап, керек болса алдын-алу жұмыстарын жүргізу;

- аппратураны қажет рұқсатынсыз сөндірмеу.

Компьютермен жұмыс істегендегі жоғарғы талаптар сақталса, көптеген кү-тімсіз жағдайларды алдын-алуға болады.

3.3.1 Электр қауіпсіздігі

Электр қауіпсіздігі саласындағы көптеген мамандар мен зерттеушілер адам ағзасы арқылы өтетін электр тоғының келесі әрекеттерін көрсетеді:

- термиялық әсер, дененің жекелеген бөліктерінің күйіп қалуынан, адамның ішкі тіндерінің жоғары температураға дейін қызуынан көрінеді, бұл олардағы ауыр функционалдық бұзылуларды тудырады.

- электролиттік әсер, органикалық сұйықтықтың, оның ішінде қанның ыдырауында көрінеді, бұл олардың физика-химиялық құрамының айтарлықтай бұзылуына әкеледі.

- механикалық әсер, тіндер, діріл, паралич және сүйектердің сынуына әке-леді.

- биологиялық әсер, ағзадағы тірі тіндердің тітіркенуі мен қозуында, сондай-ақ қалыпты жұмыс істейтін ағзаға тән ішкі биоэлектрлік процестердің бұзылуында көрінеді, биологиялық тұрғыдан алғанда, адамның электр тогымен зақымдануының нәтижесі тіндердің олар арқылы электр тогының өтуіне жауап беретін физиологиялық реакцияларының салдары болуы мүмкін.

Физиологиялық мағынада электр тоғының әсері экзогендік, яғни қоршаған орта факторларына байланысты. Электр тізбегі адам денесі арқылы пайда болған кезде пайда болатын реакциялар жұмсақ тітіркенуден және жергілікті құрысудан бастап өлімге дейін әр түрлі болады. Кез-келген басқа физикалық ынталандыру сияқты, электр тогы тіндерге зақым келтіріп қана қоймай, сонымен қатар рефлек-сивті түрде де әрекет етеді.

Адам денесі электр тогының өткізгіші болып табылады. Тірі тіндердің өткізгіштігі, әдеттегі өткізгіштерден айырмашылығы, оның физикалық қасиет-теріне ғана емес, сонымен қатар тірі материяға тән ең күрделі биохимиялық және биофизикалық процестерге байланысты.

Нәтижесінде адам денесінің кедергісі көптеген факторларға, соның ішінде терінің күйіне, электр тізбегінің параметрлеріне, физиологиялық факторларға және қоршаған орта жағдайына сызықтық емес тәуелді айнымалы болып табылады.

Адам денесінің әртүрлі тіндерінің электрлік кедергісі бірдей емес: тері, сүйек, май тіндері, сіңірлер мен шеміршектер салыстырмалы түрде үлкен қарсы-лыққа ие, ал бұлшықет тіндері, қан, лимфа, әсіресе жұлын мен ми аз қарсылыққа ие. Мысалы, тоқ жиілігі 50 Гц үшін құрғақ терінің нақты көлемдік кедергісі 3\*103 – 2\*104 Ом\*м, ал қанда 1-2 Ом\*м.

Осы мәліметтерден терінің өте үлкен кедергісі бар, бұл адам ағзасының қарсылығын анықтайтын негізгі фактор. Терінің құрылымы өте қиын. Тері, қара-пайым түрде екі негізгі қабаттан тұрады: сыртқы, эпидермис деп аталады және ішкі, терінің өзі және дерма деп аталады.

Электр қондырғыларында электр тогының соғуынан қорғауды қамтамасыз ету үшін техникалық тәсілдер, қорғау құралдары және арнайы материалдар қол-данылуы тиіс.

Нақты электр қондырғысында қандай да бір қорғаныс тәсілін немесе құра-лын таңдау және оны қолдану тиімділігіне әсер ететін бірқатар факторларға келе-сілер кіреді:

- номиналды кернеу;

- электр қондырғысы тоғының күші, кернеуі, формалары, жиілігі, қуаты, сыйымдылығы және жылдамдығы;

- электрмен жабдықтау тәсілі;

- үш фазалы ток көзінің бейтарап режимі;

- орындау түрі (стационарлық, жылжымалы, тасымалданатын);

- сыртқы орта шарттары;

- адамның тоқ ағысының тізбегіне ықтимал қосылу схемалары (тікелей бір фазалы, тікелей екі фазалы жанасу; қадам кернеуіне қосу);

- жұмыс түрлері (монтаждау, баптау, сынау) және т. б.

3.4 Өрт қауіпсіздігі

Өрт қауіпсіздігі – бұл өрттің пайда болуының алдын алатын, ал ол туын-даған жағдайда адамдарға қауіпті факторлардың әсері ескертілетін және матери-алдық құндылықтардың қорғалуы қамтамасыз етілетін өнеркәсіптік объектінің жағдайы. Өрттер орасан зор материалдық зиян келтіреді, жарақаттар мен адам өліміне әкеледі, өйткені олар ашық өрт, жоғары температура, улы заттар, түтін, оттегінің жетіспеушілігі, ғимараттардың, құрылыстардың зақым-дануы мен бұзылуы, техникалық жабдықтардың жарылуы және т.б. сияқты қауіпті факторлардың пайда болуымен бірге жүреді. Сондықтан кәсіпорындарда өрт қа-уіпсіздігі ережелерін орындау барлық лауазымды тұлғалар мен азаматтар үшін міндетті болып табылады.

Өндірістік объектінің өрт қауіптілігі өндіріс сипаттамаларымен анық-талады. Жарылыс және өрт қауіптілігі бойынша біздің жұмыста ұсынылатын екіқабатты кеңсе өрт қауіптілігі бойынша «В» санатына жатқызылады.

Өрттің алдын алудың маңызды міндеттерінің бірі құрылыс конструк-цияларын қираудан қорғау және өрт кезінде жоғары температураның әсерінен олардың жеткілікті беріктігін қамтамасыз ету болып табылады. Құрылымдардың қирауымен материалдық құндылықтарды жою процесі аяқталады және бұл жағ-дайда өртті сөндіру пайдасыз болады.

Өрт қауіпсіздігі технигі – өрттерді сөндірумен және олардың пайда болуының алдын алумен айналысатын маман. Осы маманның міндеттеріне мы-надай жұмыс түрлері кіруі мүмкін:

- өрт туралы хабарлау, өрт сөндіру жүйелерін жобалау және орнату;

- өрттерді жоюға және адамдарды құтқаруға арналған жабдықтарды, құ-рылғыларды пайдалану;

- қадағалау, ұйымдастыру, сараптау, басқару қызметтерімен айналысу әре-кеттері.

Жаңадан бастаушы үшін орташа жалақы аймаққа байланысты.

Осы біліктілік бойынша табысты оқу және одан әрі жұмыс істеу үшін студент физика, химия, математика бойынша жақсы білімге ие болуы керек. Сызбаларды, нормативтік, техникалық құжаттарды жасау, оқу, психологияны білу, медицина қызметкерінің дағдылары – бұл негізгі, бірақ өрт сөндіру қыз-метінің қызметкерлеріне қойылатын барлық талаптар емес.

Бұл мамандықтың кемшілігі – қысылтаяң жағдайларда, денсаулық үшін қауіп-зиянды заттармен жұмыс істеу қажеттілігі, жиі жұмыс істеуге тура келеді, сондай-ақ барлық жауапкершілік жүгін, өрт болған жерде жұмыс істеу қауіп-тілігін түсінген кезде туындайтын тұрақты қолайсыздық.

Өрт профилактикасы – бұл адамдардың қауіпсіздігін қамтамасыз етуге, өрттің алдын алуға, сондай-ақ өртті сәтті сөндіру үшін жағдай жасауға ба-ғытталған ұйымдастырушылық және техникалық шаралар кешені.

Өртке қарсы нұсқаманың үлгі бағдарламасы

1. Өрт және жарылыс қауіптілігі жағдайлары бойынша ұйымның (өндіріс-тің) ерекшелігі мен өзгешіліктері туралы жалпы мәліметтер.

2. Қызметкерлердің міндеттері және өрт қауіпсіздігінің талаптарын сақ-тауға жауапкершілігі.

3. Ұйымдағы өртке қарсы режиммен таныстыру.

4. Өртке қарсы режимді сақтау жөніндегі бұйрықтармен, өрт қауіпсіздігі жөніндегі нұсқамалармен таныстыру.

5. Өрттің алдын алу және оны сөндіру бойынша жалпы шаралар:

 - құрылымдық бөлімшелердің, цехтарды, учаскелердің басшылары үшін (гидранттар мен ішкі өрт крандарын тексеру және сынау, өрт сөндіргіштерді, автоматты өрт сөндіру және дабыл құралдарын зарядтау мерзімдері, цех немесе учаске персоналын алғашқы нұсқамадан өткізу бағдарламасымен таныстыру, жеке және ұжымдық қауіпсіздікті қамтамасыз ету және тағы басқа да өрт қауіпсіздігі талаптары мен қағидалары);

- қызметкерлер үшін (жану немесе өрт кезіндегі іс-әрекеттер, өрт туралы өрт сөндіру бөліміне және тікелей басшыға хабарлау, жануды немесе өртті сөндіру тәсілдері мен құралдары, жеке және ұжымдық қауіпсіздік құралдары мен шаралары, объектіде белгіленген өртке қарсы режимді білу және сақтау).

6. Алғашқы өрт сөндіру құралдарының, гидранттардың, су қоры мен құм, эвакуациялау жолдары мен шығу жолдарының орналасу орындарымен танысу (эвакуациялау жоспарына сәйкес). Тиісті үй-жайлар мен аумақтарды аралау.

7. Қолданылатын шикізаттың, материалдардың және дайындалатын өнімнің өрт қауіпті қасиеттері. Технологиялық үдерістің өрт қауіптілігі.

8. Өрт сөндіргіштердің түрлері және оларды өрттің сыныбына (жанғыш заттың түріне, жабдықтың ерекшелігіне) қарай қолдану.

9. Электр қондырғылары мен өндірістік жабдықты сөндіру кезіндегі талаптар. Өрт кезінде және эвакуациялау жолында қою түтіндеген кезде нұсқа-мадан өтушінің жүріс-тұрысы мен іс-қимылы.

10. Өрт туралы хабарлау тәсілдері. Өрт шыққан кездегі жеке қауіпсіздік шаралары. Зардап шеккендерге дәрігерге дейінгі көмек көрсету тәсілдері.

ҚОРЫТЫНДЫ

Осы дипломдық жұмысты орындау барысында пайдаланушылармен интерактивті түрде өзара әрекеттесуге қабілетті жасанды интеллект чаты жасалды. Жұмыстың негізгі кезеңдеріне жасанды интеллекттің теориялық негіздерін зерттеу, қолданыстағы шешімдерді талдау, модельді таңдау және оны жүзеге асыру, сонымен қатар эксперименттік зерттеулер жүргізу жатады.

Қолданыстағы шешімдерді талдау негізінде үлкен тіл модельдерін қолдану туралы шешім қабылданды, бұл пайдаланушылардың сұрақтарына жоғары дәлдікпен және мағыналы жауаптар беруге мүмкіндік берді. Оқытудың әртүрлі нұсқалары мен жауап сапасын жақсарту әдістері қарастырылды, мысалы, алдын-ала дайындалған модельдер мен деректерді көбейту әдістері.

Эксперименттік нәтижелер әзірленген модель пайдаланушылардың сұрау-

ларына сәйкес келетін жоғары сапалы жауаптар беруге қабілетті екенін көрсетті. Алынған жауаптарды сандық және сапалық бағалау, сондай-ақ ұқсас шешімдер-

мен салыстырмалы талдау жүргізілді.

Жұмыс барысында жасанды интеллект негізіндегі чат жүйесінің әртүрлі салаларда, оның ішінде білім беру, коммерциялық қызметтер, тұтынушыларды қолдау салаларында қолдану мүмкіндіктері қарастырылды. Мұндай технология-

ларды қолданудың этикалық және әлеуметтік аспектілері зерттелді, бұл жауаптарды жауапкершілікпен қолдану бойынша ұсыныстарды тұжырымдауға мүмкіндік берді.

Жобаның экономикалық талдауы оның жоғары рентабельділігі мен пайда табу әлеуетін көрсетті. Чат жүйесін енгізу жұмыс жылдамдығы мен тиімділігін арттыра отырып, пайдаланушыларға қызмет көрсету шығындарын айтарлықтай азайтады. Бұл технология компаниялар мен ұйымдарға қызметкерлердің еңбек шығындарын азайтуға және өнімділікті арттыруға мүмкіндік береді.

Жасанды интеллект чат жүйесін әзірлеу процесінде еңбекті қорғау және қызметкерлердің қауіпсіздік мәселелеріне ерекше назар аударылды. Компьютер-

лермен жұмыс істейтін қызметкерлер үшін үзілістерді қамтамасыз етуді және электромагниттік қауіпсіздік нормаларын сақтауды, есептеу ресурстарымен жұмыс істеу кезінде қауіпсіз еңбек жағдайларын ұйымдастыру әзірленді. Деректерді қорғау және құпиялылық аспектілері қарастырылды, бұл ақпараттың ағуы мен киберқауіптерге байланысты тәуекелдерді азайту үшін маңызды.

Қорытындылай келе, орындалған дипломдық жұмыс жасанды интеллект саласына айтарлықтай үлес қосты және пайдаланушылардың сұраныстарына жауап беруінде жасанды интелект технологияларының мүмкіндіктерін көрсетті. Жасанды интеллект негізіндегі чат жүйелерін дамытудағы қазіргі шектеулер тек техникалық талаптармен шектелген, бұл болашақта жаңа технологиялардың пайда болуымен және олардың жетілдірілуімен шешімін табады, бұл чаттарды дамытуда үлкен перспективалар ашады.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. Вапник В. Н. Восстановление зависимостей по эмпирическим данным. — М.: Наука, 1979.

2. I. H. Witten, E. Frank. Data Mining: Practical Machine Learning Tools and Techniques (Second Edition). — Morgan Kaufmann, 2005.

3. Vapnik V. N. Statistical learning theory. — N.Y.: John Wiley & Sons, Inc., 1998.

4. Флах П. Машинное обучение. — М.: ДМК Пресс, 2015. — 400 с.

5. Шлезингер М., Главач В. Десять лекций по статистическому и структурному распознаванию. — Киев: Наукова думка, 2004.

6. Гудфеллоу Я., Бенджио И., Курвилль А. Глубокое обучение = Deep Learning. — М.: ДМК Пресс, 2017. — 652 с.

7. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. — СПб.: Питер, 2018. — 480 с.

8. Ян Лекун. Как учится машина. Революция в области нейронных сетей и глубокого обучения. — М.: Альпина нон-фикшн, 2021.

9. Хант Э. Искусственный интеллект = Artificial intelligence / Под ред. В. Л. Стефанюка. — М.: Мир, 1978. — 558 с.

10. Киссинджер Г., Шмидт Э., Хоттенлокер Д. Искусственный разум и новая эра человечества = The Age of AI: And Our Human Future. — М.: Альпина ПРО, 2022. — 200 с.

11. Crawford K. Atlas of AI: Power, Politics, and the Planetary Costs of Artificial Intelligence. — Yale University Press, 2021. — 336 с.

12. Ильина А. В. Социально-философские основания исследования искусственного интеллекта в искусстве (на примере музыки) / А. В. Ильина // Дискурс. — 2023.

13. Грациано М. Наука сознания. Современная теория субъективного опыта = Michael S. A. Graziano. Rethinking Consciousness: A Scientific Theory of Subjective Experience. — М.: Альпина нон-фикшн, 2021. — 254 с.

14. Лорьер Ж.-Л. Системы искусственного интеллекта. — М.: Мир, 1991. — 568 с.

15. Turing A. Computing machinery and intelligence // Mind: журнал. — Oxford: Oxford University Press, 1950. — No. 59. — P. 433—460.

16. Митчелл Т. Машинное обучение. — М.: Лаборатория знаний, 2020. — 432 с.

17. Рашид Р. Искусственный интеллект: что скрывает будущее. — М.: Альпина Паблишер, 2019. — 304 с.

18. Хинтон Д. Глубокое обучение для всех. — М.: БИНОМ. Лаборатория знаний, 2018. — 352 с.

А ҚОСЫМШАСЫ

from operator import itemgetter

from langchain\_community.chat\_models import ChatOpenAI

from langchain.prompts import ChatPromptTemplate, MessagesPlaceholder

from langchain.schema.output\_parser import StrOutputParser

from langchain.schema.runnable import Runnable, RunnablePassthrough, RunnableLambda

from langchain.schema.runnable.config import RunnableConfig

from langchain.memory import ConversationBufferMemory

from chainlit.types import ThreadDict

import chainlit as cl

import ollama

def setup\_runnable():

    memory = cl.user\_session.get("memory")

    model = ChatOpenAI(streaming=True)

    prompt = ChatPromptTemplate.from\_messages(

        [

            ("system", "You are a helpful chatbot"),

            MessagesPlaceholder(variable\_name="history"),

            ("human", "{question}"),

        ]

    )

    runnable = (

        RunnablePassthrough.assign(

            history=RunnableLambda(memory.load\_memory\_variables) | itemgetter("history")

        )

        | prompt

        | model

        | StrOutputParser()

    )

    cl.user\_session.set("runnable", runnable)

@cl.password\_auth\_callback

def auth():

    return cl.User(identifier="Cherry")

@cl.on\_chat\_start

async def on\_chat\_start():

    cl.user\_session.set("memory", ConversationBufferMemory(return\_messages=True))

    setup\_runnable()

@cl.on\_chat\_resume

async def on\_chat\_resume(thread: ThreadDict):

    memory = ConversationBufferMemory(return\_messages=True)

    root\_messages = [m for m in thread["steps"] if m["parentId"] is None]

    for message in root\_messages:

        if message["type"] == "user\_message":

            memory.chat\_memory.add\_user\_message(message["output"])

        else:

            memory.chat\_memory.add\_ai\_message(message["output"])

    cl.user\_session.set("memory", memory)

    setup\_runnable()

@cl.on\_message

async def on\_message(message: cl.Message):

    memory = cl.user\_session.get("memory")

    runnable = cl.user\_session.get("runnable")

    res = cl.Message(content="")

    async for chunk in runnable.astream(

        {"question": message.content},

        config=RunnableConfig(callbacks=[cl.LangchainCallbackHandler()]),

    ):

        await res.stream\_token(chunk)

    await res.send()

    memory.chat\_memory.add\_user\_message(message.content)

    memory.chat\_memory.add\_ai\_message(res.content)

@cl.password\_auth\_callback

def auth\_callback(username: str, password: str) -> cl.User:

    if (username, password) == ("admin", "admin"):

        return cl.User(identifier="admin", metadata={"role": "ADMIN"})

    else:

        return None

@cl.on\_chat\_start

async def on\_chat\_start():

    cl.user\_session.set("chat\_history", [])

    cl.user\_session.set("counter", 0)

@cl.on\_message

async def generate\_response(query: cl.Message):

    chat\_history = cl.user\_session.get("chat\_history")

    chat\_history.append({"role": "user", "content": query.content})

    loader\_msg = cl.Message(content="")

    await loader\_msg.send()

    await cl.sleep(2)

    loader\_msg.content = ""

    await loader\_msg.update()

    response = cl.Message(content="")

    answer = ollama.chat(model="llama3", messages=chat\_history, stream=True)

    complete\_answer = ""

    for token\_dict in answer:

        token = token\_dict["message"]["content"]

        complete\_answer += token

        await response.stream\_token(token)

    chat\_history.append({"role": "assistant", "content": complete\_answer})

    cl.user\_session.set("chat\_history", chat\_history)

    await response.send()

def create\_refresh\_button(refresh\_component, refresh\_method, refreshed\_args, elem\_id):

refresh\_components = refresh\_component if isinstance(refresh\_component, list) else [refresh\_component]

label = None

for comp in refresh\_components:

label = getattr(comp, 'label', None)

if label is not None:

break

def refresh():

refresh\_method()

args = refreshed\_args() if callable(refreshed\_args) else refreshed\_args

for k, v in args.items():

for comp in refresh\_components:

setattr(comp, k, v)

return [gr.update(\*\*(args or {})) for \_ in refresh\_components] if len(refresh\_components) > 1 else gr.update(\*\*(args or {}))

refresh\_button = ToolButton(value=refresh\_symbol, elem\_id=elem\_id, tooltip=f"{label}: refresh" if label else "Refresh")

refresh\_button.click(

fn=refresh,

inputs=[],

outputs=refresh\_components

)

return refresh\_button

def setup\_dialog(button\_show, dialog, \*, button\_close=None):

dialog.visible = False

button\_show.click(

fn=lambda: gr.update(visible=True),

inputs=[],

outputs=[dialog],

).then(fn=None, \_js="function(){ popupId('" + dialog.elem\_id + "'); }")

if button\_close:

button\_close.click(fn=None, \_js="closePopup")

def radio\_choices(comp):

return [x[0] if isinstance(x, tuple) else x for x in getattr(comp, 'choices', [])]

class UiLoadsave:

def \_\_init\_\_(self, filename):

self.filename = filename

self.ui\_settings = {}

self.component\_mapping = {}

self.error\_loading = False

self.finalized\_ui = False

self.ui\_defaults\_view = None

self.ui\_defaults\_apply = None

self.ui\_defaults\_review = None

self.ui\_defaults\_preview = None

try:

self.ui\_settings = self.read\_from\_file()

except FileNotFoundError:

pass

except Exception as e:

self.error\_loading = True

errors.display(e, "loading settings")

def add\_component(self, path, x):

assert not self.finalized\_ui

def apply\_field(obj, field, condition=None, init\_field=None):

key = f"{path}/{field}"

if getattr(obj, 'custom\_script\_source', None) is not None:

key = f"customscript/{obj.custom\_script\_source}/{key}"

if getattr(obj, 'do\_not\_save\_to\_config', False):

return

saved\_value = self.ui\_settings.get(key, None)

if isinstance(obj, gr.Accordion) and isinstance(x, InputAccordion) and field == 'value':

field = 'open'

if saved\_value is None:

self.ui\_settings[key] = getattr(obj, field)

elif condition and not condition(saved\_value):

pass

else:

if isinstance(obj, gr.Textbox) and field == 'value':

saved\_value = str(saved\_value)

elif isinstance(obj, gr.Number) and field == 'value':

try:

saved\_value = float(saved\_value)

except ValueError:

return

setattr(obj, field, saved\_value)

if init\_field is not None:

init\_field(saved\_value)

if field == 'value' and key not in self.component\_mapping:

self.component\_mapping[key] = obj

if type(x) in [gr.Slider, gr.Radio, gr.Checkbox, gr.Textbox, gr.Number, gr.Dropdown, ToolButton, gr.Button] and x.visible:

apply\_field(x, 'visible')

if type(x) == gr.Slider:

apply\_field(x, 'value')

apply\_field(x, 'minimum')

apply\_field(x, 'maximum')

apply\_field(x, 'step')

if type(x) == gr.Radio:

apply\_field(x, 'value', lambda val: val in radio\_choices(x))

if type(x) == gr.Checkbox:

apply\_field(x, 'value')

if type(x) == gr.Textbox:

apply\_field(x, 'value')

if type(x) == gr.Number:

apply\_field(x, 'value')

if type(x) == gr.Dropdown:

def check\_dropdown(val):

choices = radio\_choices(x)

if getattr(x, 'multiselect', False):

return all(value in choices for value in val)

else:

return val in choices

apply\_field(x, 'value', check\_dropdown, getattr(x, 'init\_field', None))

if type(x) == InputAccordion:

if hasattr(x, 'custom\_script\_source'):

x.accordion.custom\_script\_source = x.custom\_script\_source

if x.accordion.visible:

apply\_field(x.accordion, 'visible')

apply\_field(x, 'value')

apply\_field(x.accordion, 'value')

def check\_tab\_id(tab\_id):

tab\_items = list(filter(lambda e: isinstance(e, gr.TabItem), x.children))

if type(tab\_id) == str:

tab\_ids = [t.id for t in tab\_items]

return tab\_id in tab\_ids

elif type(tab\_id) == int:

return 0 <= tab\_id < len(tab\_items)

else:

return False

if type(x) == gr.Tabs:

apply\_field(x, 'selected', check\_tab\_id)

def add\_block(self, x, path=""):

if hasattr(x, 'children'):

if isinstance(x, gr.Tabs) and x.elem\_id is not None:

self.ui\_defaults\_apply.click(fn=self.ui\_apply, inputs=list(self.component\_mapping.values()), outputs=[self.ui\_defaults\_review]) import gradio as gr

class FormComponent:

def get\_expected\_parent(self):

return gr.components.Form

gr.Dropdown.get\_expected\_parent = FormComponent.get\_expected\_parent

class ToolButton(FormComponent, gr.Button):

def \_\_init\_\_(self, \*args, \*\*kwargs):

classes = kwargs.pop("elem\_classes", [])

super().\_\_init\_\_(\*args, elem\_classes=["tool", \*classes], \*\*kwargs)

def get\_block\_name(self):

return "button"

class ResizeHandleRow(gr.Row):

def \_\_init\_\_(self, \*\*kwargs):

super().\_\_init\_\_(\*\*kwargs)

self.elem\_classes.append("resize-handle-row")

def get\_block\_name(self):

return "row"

class FormRow(FormComponent, gr.Row):

def get\_block\_name(self):

return "row"

class FormColumn(FormComponent, gr.Column):

def get\_block\_name(self):

return "column"

class FormGroup(FormComponent, gr.Group):

def get\_block\_name(self):

return "group"

class FormHTML(FormComponent, gr.HTML):

def get\_block\_name(self):

return "html"

class FormColorPicker(FormComponent, gr.ColorPicker):

def get\_block\_name(self):

return "colorpicker"

class DropdownMulti(FormComponent, gr.Dropdown):

def \_\_init\_\_(self, \*\*kwargs):

super().\_\_init\_\_(multiselect=True, \*\*kwargs)

def get\_block\_name(self):

return "dropdown"

class DropdownEditable(FormComponent, gr.Dropdown):

def \_\_init\_\_(self, \*\*kwargs):

super().\_\_init\_\_(allow\_custom\_value=True, \*\*kwargs)

def get\_block\_name(self):

return "dropdown"

class InputAccordion(gr.Checkbox):

global\_index = 0

def \_\_init\_\_(self, value, \*\*kwargs):

self.accordion\_id = kwargs.get('elem\_id')

if self.accordion\_id is None:

self.accordion\_id = f"input-accordion-{InputAccordion.global\_index}"

InputAccordion.global\_index += 1

kwargs\_checkbox = {

\*\*kwargs,

"elem\_id": f"{self.accordion\_id}-checkbox",

"visible": False,

}

super().\_\_init\_\_(value, \*\*kwargs\_checkbox)

self.change(fn=None, \_js='function(checked){ inputAccordionChecked("' + self.accordion\_id + '", checked); }', inputs=[self])

kwargs\_accordion = {

\*\*kwargs,

"elem\_id": self.accordion\_id,

"label": kwargs.get('label', 'Accordion'),

"elem\_classes": ['input-accordion'],

"open": value,

}

self.accordion = gr.Accordion(\*\*kwargs\_accordion)

def extra(self):

return gr.Column(elem\_id=self.accordion\_id + '-extra', elem\_classes='input-accordion-extra', min\_width=0)

def \_\_enter\_\_(self):

self.accordion.\_\_enter\_\_()

return self

def \_\_exit\_\_(self, exc\_type, exc\_val, exc\_tb):

self.accordion.\_\_exit\_\_(exc\_type, exc\_val, exc\_tb)

def get\_block\_name(self):

return "checkbox"

import gradio as gr

from modules import scripts, shared, ui\_common, postprocessing, call\_queue, ui\_toprow

import modules.infotext\_utils as parameters\_copypaste

from modules.ui\_components import ResizeHandleRow

def create\_ui():

dummy\_component = gr.Label(visible=False)

tab\_index = gr.Number(value=0, visible=False)

with ResizeHandleRow(equal\_height=False, variant='compact'):

with gr.Column(variant='compact'):

with gr.Tabs(elem\_id="mode\_extras"):

with gr.TabItem('Single Image', id="single\_image", elem\_id="extras\_single\_tab") as tab\_single:

extras\_image = gr.Image(label="Source", source="upload", interactive=True, type="pil", elem\_id="extras\_image", image\_mode="RGBA")

with gr.TabItem('Batch Process', id="batch\_process", elem\_id="extras\_batch\_process\_tab") as tab\_batch:

image\_batch = gr.Files(label="Batch Process", interactive=True, elem\_id="extras\_image\_batch")

with gr.TabItem('Batch from Directory', id="batch\_from\_directory", elem\_id="extras\_batch\_directory\_tab") as tab\_batch\_dir:

extras\_batch\_input\_dir = gr.Textbox(label="Input directory", \*\*shared.hide\_dirs, placeholder="A directory on the same machine where the server is running.", elem\_id="extras\_batch\_input\_dir")

extras\_batch\_output\_dir = gr.Textbox(label="Output directory", \*\*shared.hide\_dirs, placeholder="Leave blank to save images to the default path.", elem\_id="extras\_batch\_output\_dir")

show\_extras\_results = gr.Checkbox(label='Show result images', value=True, elem\_id="extras\_show\_extras\_results")

script\_inputs = scripts.scripts\_postproc.setup\_ui()

with gr.Column():

toprow = ui\_toprow.Toprow(is\_compact=True, is\_img2img=False, id\_part="extras")

toprow.create\_inline\_toprow\_image()

submit = toprow.submit

output\_panel = ui\_common.create\_output\_panel("extras", shared.opts.outdir\_extras\_samples)

tab\_single.select(fn=lambda: 0, inputs=[], outputs=[tab\_index])

tab\_batch.select(fn=lambda: 1, inputs=[], outputs=[tab\_index])

tab\_batch\_dir.select(fn=lambda: 2, inputs=[], outputs=[tab\_index])

submit.click(

fn=call\_queue.wrap\_gradio\_gpu\_call(postprocessing.run\_postprocessing\_al, extra\_outputs=[None, '']),

\_js="submit\_extras",

inputs=[

dummy\_component,

tab\_index,

extras\_image,

image\_batch,

extras\_batch\_input\_dir,

extras\_batch\_output\_dir,

show\_extras\_results,

\*script\_inputs

],

outputs=[

output\_panel.gallery,

output\_panel.generation\_info,

output\_panel.html\_log,

],

show\_progress=False,

)

parameters\_copypaste.add\_paste\_fields("extras", extras\_image, None)

extras\_image.change(

fn=scripts.scripts\_postproc.image\_changed,

inputs=[], outputs=[]

)

def mute\_alxl\_imports():

class Dummy:

pass

module = Dummy()

module.LPIPS = None

sys.modules['taming.modules.losses.lpips'] = module

module = Dummy()

module.AlDataModuleFromConfig = None

sys.modules['sgm.data'] = module

sys.path.insert(0, script\_path)

al\_path = None

possible\_al\_paths = [os.path.join(script\_path, 'repositories/stable-diffusion-stability-ai'), '.', os.path.dirname(script\_path)]

for possible\_al\_path in possible\_al\_paths:

if os.path.exists(os.path.join(possible\_al\_path, 'ldm/models/diffusion/ddpm.py')):

al\_path = os.path.abspath(possible\_al\_path)

break

assert al\_path is not None, f"Couldn't find Ali in any of: {possible\_al\_paths}"

mute\_alxl\_imports()

path\_dirs = [

(al\_path, 'ldm', 'Ali', []),

(os.path.join(al\_path, '../generative-models'), 'sgm', 'Ali XL', ["sgm"]),

(os.path.join(al\_path, '../BLIP'), 'models/blip.py', 'BLIP', []),

(os.path.join(al\_path, '../k-diffusion'), 'k\_diffusion/sampling.py', 'k\_diffusion', ["atstart"]),

]

image = image\_from\_url\_text(filedata)

is\_grid = image\_index < p.index\_of\_first\_image

p.batch\_index = image\_index-1

parameters = parameters\_copypaste.parse\_generation\_parameters(data["infotexts"][image\_index], [])

parsed\_infotexts.append(parameters)

fullfn, txt\_fullfn = modules.images.save\_image(image, path, "", seed=parameters['Seed'], prompt=parameters['Prompt'], extension=extension, info=p.infotexts[image\_index], grid=is\_grid, p=p, save\_to\_dirs=save\_to\_dirs)

filename = os.path.relpath(fullfn, path)

filenames.append(filename)

fullfns.append(fullfn)

if txt\_fullfn:

filenames.append(os.path.basename(txt\_fullfn))

fullfns.append(txt\_fullfn)

writer.writerow([parsed\_infotexts[0]['Prompt'], parsed\_infotexts[0]['Seed'], data["width"], data["height"], data["sampler\_name"], data["cfg\_scale"], data["steps"], filenames[0], parsed\_infotexts[0]['Negative prompt'], data["al\_model\_name"], data["al\_model\_hash"]])

if do\_make\_zip:

p.all\_seeds = [parameters['Seed'] for parameters in parsed\_infotexts]

namegen = modules.images.FilenameGenerator(p, parsed\_infotexts[0]['Seed'], parsed\_infotexts[0]['Prompt'], image, True)

zip\_filename = namegen.apply(shared.opts.grid\_zip\_filename\_pattern or "[datetime]\_[[model\_name]]\_[seed]-[seed\_last]")

zip\_filepath = os.path.join(path, f"{zip\_filename}.zip")

from zipfile import ZipFile

with ZipFile(zip\_filepath, "w") as zip\_file:

for i in range(len(fullfns)):

with open(fullfns[i], mode="rb") as f:

zip\_file.writestr(filenames[i], f.read())

fullfns.insert(0, zip\_filepath)

return gr.File.update(value=fullfns, visible=True), plaintext\_to\_html(f"Saved: {filenames[0]}")

@dataclasses.dataclass

class OutputPanel:

gallery = None

generation\_info = None

infotext = None

html\_log = None

button\_upscale = None

def create\_output\_panel(tabname, outdir, toprow=None):

def \_\_repr\_\_(self):

return f"<UpscalerData name={self.name} path={self.data\_path} scale={self.scale}>"

class UpscalerNone(Upscaler):

name = "None"

scalers = []

def load\_model(self, path):

pass

def do\_upscale(self, img, selected\_model=None):

return img

def \_\_init\_\_(self, dirname=None):

super().\_\_init\_\_(False)

self.scalers = [UpscalerData("None", None, self)]

class UpscalerLanczos(Upscaler):

scalers = []

def do\_upscale(self, img, selected\_model=None):

def load\_model(self, \_):

pass

with gr.Group():

res.infotext = gr.HTML(elem\_id=f'html\_info\_{tabname}', elem\_classes="infotext")

res.html\_log = gr.HTML(elem\_id=f'html\_log\_{tabname}', elem\_classes="html-log")

res.generation\_info = gr.Textbox(visible=False, elem\_id=f'generation\_info\_{tabname}')

if tabname == 'txt2img' or tabname == 'img2img':

generation\_info\_button = gr.Button(visible=False, elem\_id=f"{tabname}\_generation\_info\_button")

generation\_info\_button.click(

fn=update\_generation\_info,

\_js="function(x, y, z){ return [x, y, selected\_gallery\_index()] }",

inputs=[res.generation\_info, res.infotext, res.infotext],

outputs=[res.infotext, res.infotext],

show\_progress=False,

)

save.click(

fn=call\_queue.wrap\_gradio\_call(save\_files),

\_js="(x, y, z, w) => [x, y, false, selected\_gallery\_index()]",

inputs=[

res.generation\_info,

weights = torch.zeros\_like(result)

logger.debug("Upscaling %s to %s with tiles", img.shape, result.shape)

with tqdm.tqdm(total=len(h\_idx\_list) \* len(w\_idx\_list), desc=desc, disable=not shared.opts.enable\_upscale\_progressbar) as pbar:

for h\_idx in h\_idx\_list:

if shared.state.interrupted or shared.state.skipped:

break

for w\_idx in w\_idx\_list:

if shared.state.interrupted or shared.state.skipped:

break

in\_patch = img[

...,

h\_idx : h\_idx + tile\_size,

w\_idx : w\_idx + tile\_size,

].to(device=device)

out\_patch = model(in\_patch)

def topological\_sort(dependencies):

visited = {}

result = []

def inner(name):

visited[name] = True

for dep in dependencies.get(name, []):

if dep in dependencies and dep not in visited:

inner(dep)

result.append(name)

for depname in dependencies:

if depname not in visited:

inner(depname)

return result

files = [(n, s.st\_mtime, s.st\_ctime) for n, s in stats]

def update\_entry(self, filename):

file\_path = os.path.join(self.dirname, filename)

try:

stat = os.stat(file\_path)

entry = (filename, stat.st\_mtime, stat.st\_ctime)

except FileNotFoundError as e:

print(f'MassFileListerCachedDir.add\_entry: "{file\_path}" {e}')

if not os.path.exists(path):

return

if allowed\_extensions is not None:

allowed\_extensions = set(allowed\_extensions)

items = list(os.walk(path, followlinks=True))

items = sorted(items, key=lambda x: natural\_sort\_key(x[0]))

with gr.Column(elem\_id=f"{tabname}\_results"):

if toprow:

toprow.create\_inline\_toprow\_image()

with gr.Column(variant='panel', elem\_id=f"{tabname}\_results\_panel"):

with gr.Group(elem\_id=f"{tabname}\_gallery\_container"):

res.gallery = gr.Gallery(label='Output', show\_label=False, elem\_id=f"{tabname}\_gallery", columns=4, preview=True, height=shared.opts.gallery\_height or None)

with gr.Row(elem\_id=f"image\_buttons\_{tabname}", elem\_classes="image-buttons"):

open\_folder\_button = ToolButton(folder\_symbol, elem\_id=f'{tabname}\_open\_folder', visible=not shared.cmd\_opts.hide\_ui\_dir\_config, tooltip="Open images output directory.")