Әл-Фараби атындағы Қазақ ұлттық университеті

ӘОЖ 004.4 Қолжазба құқында

ШОРМАКОВА АСЕМ НОЯБРЕВНА

Ағылшын тілінен қазақ тіліне машиналық аударманың пост-редакциялау үлгілерін, әдістерін және программалық құралдарын зерттеу және өңдеу

6D070300 – Ақпараттық жүйелер

Философия докторы (PhD) дәрежесін алу үшін
дайындалған диссертация

Отандық ғылыми жетекші:
т.ғ.д.,профессор Тукеев У.А.
Шетелдік ғылыми жетекші:
Phd доктор, профессор М. Форкада
(Испания, Аликанте университеті)

Қазақстан Республикасы
Алматы, 2022

МАЗМҰНЫ

БЕЛГІЛЕУЛЕР МЕН ҚЫСҚАРТУЛАР

APE– Automatic postediting–автоматты пост-редакциялау
CAT–Computer-assisted translation–компьютерлік қосымша аударма
FMS – Fuzzy match score–дәл емес сәйкестік есебі
MT – Machine Translation–машиналық аударма (МА)
NMT– Neural machine translation–нейрондық машиналық аударма
NNAPE– Neural network automatic post-editing– нейрондық желі автоматты пост-редакциялау
OL– Online learning– онлайн оқыту
OLAPE–Online analytical processing– онлайн аналитикалық өңдеу
PBSMT– Phrase-based statistical machine translation– фразалық статистикалық машиналық аударма
PE-LC – Post Edit - Lexical Choice – пост-редакция-лексикалық таңдау
QE – Quality estimation – сапаны бағалау
RNN–Recurrent neural network–рекурентті нейрондық желі
SBI – Sources of Bilingual information–екі тілді ақпарат көздері
SL – Source Language– бастапқы тіл (БТ)
SMT – Statistical Machine Translation–статистикалық машина аударылымы
SSA– Sub-segment alignment–саб-сегмент туралауы
TL – Target Language– арнаулы тіл (АТ)
TM – Translation Memory–аударма жады
TS – Translation spotting–аударма анықтамалығы
TU – Translation Unit–аударма бірлестіктер
WMT–Workshop on machine translation– машиналық аударма семинары
АА – Автоматтандырылған аударма
АЖ – Ақпараттық жүйе
ЖТ – Жады технологиясы
МА – Машиналық аударма

КІРІСПЕ

Зерттеу тақырыбының өзектілігі ақпараттық жүйелер саласындағы машиналық аударма мен пост-редакциялаудың заманауи дамуымен байланысты. Ақпараттық жүйелер қазіргі қоғам өмірінің барлық дерлік салаларында қолданылады. Сонымен қатар, ақпараттық технологиялар әр салада тиімділік пен өнімділікті арттырады және көптеген артықшылықтарға ие, сондықтан білім беру және басқа салаларда өсу үшін машиналық аударма өзекті болып табылады. Бүгінгі таңда пайдаланушылар үшін атап айтқанда, интерактивті ақпараттық жүйелер саласында машиналық аударма сапасы маңызды рөл атқарады.

Машиналық аударма – ақпараттық жүйелер саласындағы жасанды интеллекттің жетекші бағыттарының бірі. Машиналық аударма дүние жүзіндегі халықтар мен елдер арасындағы байланысты жақсартудың жаһандық мәселесін шешуде маңызды рөл атқарады. Машиналық аударманың сапасы жылдан-жылға артып келеді, бірақ кәсіби аударма сапасына әлі жеткен жоқ.

Машиналық аударманың сапасын жақсарту үшін ең маңызды және практикалық әдістердің бірі – пост-редакциялау үрдісі, яғни машиналық аударма сапасын жақсарту мақсатында машиналық аударманы түзету. Машиналық аударманы пост-редакциялау қолмен де, автоматтандырылған нұсқаларда да жүзеге асыруға болады. Машиналық аударманы қолмен пост-редакциялау – біршама еңбекті қажет ететін үрдіс. Автоматтандырылған пост-редакциялау машиналық аударма бағыты табиғи тілдердің машиналық аудармасының өзекті бағыттарының бірі болып табылады.

Соңғы жылдары машиналық аударма қолданушылар саны қарқынды өскен, әсіресе оқу орындары, жекеменшік кәсіпорындар, аударма орталықтары жиі қолданып жүр. Сонымен қоса шет елдегі компаниялардың басым көпшілігі машиналық аударманың көмегіне жүгініп жүр. Осыған қоса көптеген қолданушылар күнделікті тұрмыс жағдайында да машиналық аударманы қолданады.

Қазақ тілі машиналық аудармасы кәсіби аудармашылардың денгейіне жеткен жоқ әлі де, сол себептен қазіргі уақытта постредакциялау бағытын қолданып қазақ тілі машиналық аудармасының сапасын көтеру өте өзекті мәселе болып тұр.

Бұл жұмыстың ғылыми үлесі – ағылшын – қазақ сөйлемдерді туралау әдісімен талдау негізінде қате аударылған сөзді табу, табылған қате аударылған сөзге мағынасы жақын сөздердің тізімін (каталог) қалыптастыру және олардың ішінен лексикалық таңдау тапсырмасының технологиясын қолдана отырып, ең ықтимал дұрыс сөзді таңдауға негізделген қазақ тіліне арналған автоматты пост-редакциялау технологиясын өңдеу болып табылады.

Диссертациялық жұмыстың мақсаты. Бұл жұмыстың негізгі мақсаты – лексикалық таңдау негізінде машиналық аударманы автоматты түрде пост-

редакциялау арқылы ағылшын тілінен қазақ тіліне машиналық аударманың сапасын арттыру.

Зерттелу есептері: Осы мақсатқа жету үшін 4 тапсырма қарастырылды:

1 – қазақ тіліндегі сөйлемнің қате аударылған сөздерін анықтау;

2 – қате аударылған сөздердің синонимдер каталогын автоматты түрде қалыптастыру;

3 – қате аударылған сөзді мағынасы жағынан жақын синониммен ауыстырып түзетілген сөйлемнің машиналық аудармасын шығару.

4 – жоғарыдағы үш тапсырманы біріктіріп пост-редакциялау технологиясын құрастыру.

Зерттеу әдістері: табиғи тілдерді өңдеу үлгілері мен әдістері.

Зерттеу нысаны: ағылшын тілінен қазақ тіліне машиналық аударманың мәтіндері.

Зерттеу пәні: ағылшын тілінен қазақ тіліне машиналық аударманы автоматты түрде пост-редакциялау.

Жұмыстың ғылыми жаңалығы.

1) Алғаш рет ағылшын – қазақ машиналық аударманың Post Edit – Lexical Choice (PE-LC) пост-редакциялау технологиясы әзірленді.

2) Ағылшын тілінен қазақ тіліне қате аударылған сөздерді табу әдісі кері аудармамен жетілдірілген.

3) Алғаш рет қате аударылған қазақ сөздердің синонимдер каталогын автоматты түрде қалыптастыру әдісі әзірленді.

4) Қате аударылған сөздің ықтималдығы жоғары синоним сөзді таңдау семантикалық текше әдісінің үлгісі мен алгоритмі бейімделді.

Зерттеудің теориялық құндылығы: Зерттеудің теориялық маңыздылығы ағылшын тілінен қазақ тіліне машиналық аударманы пост-редакциялау технологиясына мәтіндерді өңдеудің белгілі әдістерін әзірлеуде және біріктіруде болып табылады.

Зерттеудің практикалық құндылығы: Зерттеудің практикалық маңыздылығы ағылшыннан қазақшаға аударылған мәтінге пост-редакциялау технологиясын құру және бағдарламалық жабдықтау құралдарын өңдеп қолдану болып табылады.

Қорғауға шығарылатын негізгі жағдайлар:

1. Ағылшын–қазақ машиналық аударманың жаңа автоматты пост-редакциялау технологиясы.
2. Ағылшын тілінен қазақ тіліне қате аударылған сөздерді анықтаудың жетілдірілген әдісі.
3. Ағылшын тілінен қате аударылған қазақ сөздер синонимдерінің каталогын автоматты түрде қалыптастыру технологиясы.
4. Семантикалық текше негізінде ықтималдығы жоғары синонимді таңдау бейімделген әдісі.

Сенімділік дәрежесі мен апробациялау нәтижелері. Алынған нәтижелердің сенімділігін пост-редакциялау технологиясының

эксперименттерінің нәтижелері, журналдардағы жарияланымдар мен халықаралық конференциялар материалдарындағы апробациялау нәтижелерінен көруге болады.

Жұмыстың ғылыми нәтижелері келесі халықаралық ғылыми конференциялар мен ғылыми семинарларда ұсынылып, талқыланды:

• ACIIDS 2017 интеллектуалды ақпарат және деректер базасы жүйелері бойынша 9-шы Азия конференциясы;

• Есептеу ұжымдық интеллект бойынша 11-ші халықаралық конференция ICCCI 2019;

• «Фараби әлемі» студенттер мен жас ғалымдардың халықаралық ғылыми конференциясы, Алматы, 2014, 2015, 2017, 2018 ж.

Сондай-ақ бұл тақырып әл-Фараби атындағы Қазақ ұлттық университетінің ақпараттық жүйелер кафедрасында және ақпараттық технологиялар факультетінің ғылыми семинарларында талқыланды.

Диссертациялық тақырыптың ғылыми бағдарламалармен байланысы. Диссертациялық жұмыс PhD докторлық диссертациясының жоспарына сәйкес және «Қазақ тіліндегі ақпараттық-аналитикалық деректерді іздеу жүйесін дамыту» гранттық қаржыландыру жобасының ғылыми-зерттеу жұмыстарының жоспарына сәйкес орындалды. (2018-2020 ж., мемлекеттік тіркеу нөмірі: No AP05132950). Диссертациялық жұмыс бойынша жүргізілген кейбір зерттеу нәтижелері осы жобаның 2018-2020 жылдарға арналған есептеріне енгізілген.

Әрбір басылымды дайындаудағы докторанттың үлесі. Жарияланған мақалалар мен ғылыми еңбектер диссертация тақырыбы бойынша зерттеу нәтижелерін сипаттайды. Ғылыми жұмыс барысында 14 ғылыми жұмыс жазылды, оның ішінде: Scopus индексі бар журналда 1 ғылыми мақала жарыққа шықты:

1. Shormakova A., Zhumanov Z.H., Rakhimova D. "Post-editing of words in Kazakh sentences for information retrieval". *Journal of Theoretical and Applied Information Technology,* 2019, 97(6), p. 1896–1908. (Scopus 2021: Q4, CiteScore-1.3; Percentile- 30%)

Қазақстан Республикасы Білім және ғылым министрлігі Білім және ғылым саласындағы бақылау комитеті ұсынған журналдарда 4 мақала шықты:

1. Абеустанова (Шормакова) А.Н. "Машиналық аударманың нарықтағы және Қазақстандағы күйі". *ҚазҰТУ хабаршысы* № 6(106), 2014. –150-152 б.
2. Абеустанова (Шормакова) А.Н. "Қазақ тіліндегі көпмағыналы сөздердің бірін анықтаудың бір болжамы". *ҚазҰТУ хабаршысы* №4(110) 2015. –625-628 б.
3. Абеустанова (Шормакова) А.Н. "Ағылшын тілінен қазақ тіліне аударылған қазақша қате сөздерді анықтау және баламалар каталогын құру". *ҚазҰТУ хабаршысы* №6 2017. –313-317 б.
4. Шормакова А.Н. "Екі табиғи тілдегі аударылған мәтінді туралау". *ҚазҰТУ хабаршысы*, №4(128), 2018. –344-349 б.

Scopus негізінде индекстелген халықаралық ғылыми-тәжірибелік конференциялар жинақтарында 2 ғылыми мақала жарияланған:

1. Abeustanova (Shormakova) A., Tukeyev U. "Automatic Post-editing of Kazakh Sentences Machine Translated from English". *Studies in Computational Intelligence/Advanced Topics in Intelligent Information and Database Systems*, vol. 710 – Springer International Publishing, 2017. – p. 283-295. (Scopus 2021: Q4, CiteScore-1.8; Percentile- 27%).

2. Rakhimova D., Assem S. "Problems of Semantics of Words of the Kazakh Language in the Information Retrieval". *Lecture Notes in Computer Science* (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2019, 11684 LNAI, p. 70–81. (Scopus 2021: Q2, SJR=0.25, CS=2.1, Percentile-50%)

Халықаралық ғылыми конференциялар жинақтарында 6 және ғылыми-техникалық журналда 1 мақала жарияланған:

1. Shormakova A. "Machine translation and post-editing". *Материалы международной конференции студентов и молодых ученых «Мир науки»*, 17-19 апреля 2013г. – Алматы: Қазақ университеті, 2013. – с. 222

2. Шормакова А.Н. "Информатика терминдерінің мемлекеттік тілге аудару ерекшеліктері". *Материалы III международного конгресса студентов и молодых ученых «Мир науки»*, 23-28 апреля, 2009г.-Алматы: Қазақ университеті,- с. 249.

3. Шормакова А.Н., Тукеев У.А. "Технология машинного перевода с обучением английского языка на казахский язык". *Материалы международной конференции студентов и молодых ученых «Мир науки»*, 23-26 апреля 2012г. – Алматы: Қазақ университеті, – с. 154.

4. Sundetova A., Forcada M.L., Shormakova A., Aitkulova A. "Structural transfer rules for English-Kazakh machine translation in the free/open-source platform Apertium", in *Proceedings of the I International Conference on Computer Processing of Turkic Languages (TurkLang-2013)* (Astana, 3-4 oct. 2013) , p. 322-331.

5. Шормакова А.Н., Айткулова А. "Добавление новой англо-казахской языковой пары в платформу машинного перевода Апертиум". *51-я Международная научная студенческая конференция «Студент и научно-технический прогресс»*, Новосибирск, 12-18 апреля 2013, Секция “Информационные технологии”.- с. 241.

6. Тукеев У.А., Абеустанова (Шормакова) А.Н., Сундетова А. "Ағылшын – қазақ тілдік жұбы үшін Apertium платформасындағы сөйлемді синтаксистік құрылымдық түрлендіру ережелері және мәселелері" . *IV международная научно-практическая конференция:* (секция «Искусственный интеллект»). Қоғамды ақпараттандыру IV Халықаралық ғылыми-практикалық конференция еңбектері, Астана 2014 ,127-129 б.

7. Шормакова А.Н. Қазақ тіліндегі автоматтандырылған синонимдер тізімін құру [Мәтін] / А.Н. Шормакова, У.А. Тукеев // Механика және технологиялар /

Ғылыми журнал. – 2022. – №3(77). – Б.44-49. https://doi.org/10.55956/AEQO3045

Диссертация құрылымы мен көлемі. Диссертациялық жұмыс кіріспеден, алты бөлімнен, қорытындыдан, пайдаланған әдебиеттер тізімінен және екі қосымшадан тұрады. 77 беттік машинамен жазылған мәтінді құрайды, оның ішіне 12 кесте, 7 сурет кіреді.

Кіріспеде диссертациялық жұмыстың өзектілігін негіздейді. Жұмыстың мақсаты, зерттеу жұмысының объектісі мен пәні тұжырымдалды. Ғылыми жаңалығы мен практикалық маңыздылығы анықталды. Жүргізілген зерттеу нәтижелері сипатталған. Зерттеу және жариялау нәтижелерін апробациялау туралы ақпарат берілген.

Бірінші бөлімде машиналық аудармаға және автоматты пост-редакциялауға шолу берілген. Диссертациялық жұмысқа қатысты қолданылған терминдер мен ұғымдар келтірілген. Пост-редакциялау бойынша бар, жаңа ғылыми жұмыстар сипатталған. Осы тақырып бойынша ғылыми еңбектерге сілтеме жасалып, шолу жасалды.

Екінші бөлімде PE-LC пост-редакциялау технологиясының құрылымы мен алгоритмі сипатталған. Диссертацияда қойылған төрт тапсырма туралы қысқаша ақпарат берілген. Зерттеу жұмысында ұсынылған PE-LC технологиясының жалпы алгоритмі жазылған.

Үшінші бөлімде Бірінші тапсырманың шешімі жан-жақты қарастырылды: аударылған сөйлемдегі қате аударылған сөзді анықтау. Ағылшын тілінен қазақ тіліне қате аударылған сөздерді анықтаудың жетілдірілген әдісі сипатталған.

Төртінші бөлімде қате аударылған сөздерден жасалған синонимдердің автоматты каталогын (тізімін) жасаудан тұратын екінші тапсырма сипатталды. Каталогты автоматты түрде қалыптастыруға арналған құралдар мен сілтемелер таныстырылды. Каталогты автоматты түрде қалыптастыру тапсырмасының қате аударылған сөздердің синонимдерінің мысалдары келтірілген.

Бесінші бөлімде үшінші тапсырма, яғни қате аударылған сөздің лексикалық таңдау мәселесі сипатталған. Семантикалық текше әдісінің жетілдірілген үлгісі мен алгоритмі негізінде берілген қате аударылған сөзге ең қолайлы синонимді таңдау ұсынылған. Табылған қате аударылған сөздерге семантикалық текшені құру кезінде кестелер мен мысалдар есептеулері келтірілген.

Алтыншы бөлімде ұсынылған PE-LC технологиясы мен оның Google Translate-пен салыстыру эксперименттерінен кейін алынған нәтижелер көрсетілген. Ұсынылған жұмыстағы жақсартуларды анықтау үшін эксперименттік мәліметтердің статистикалық мәнділігі есептеліп көрсетілді. Зерттеу жұмысының нәтижелерін салыстыру үшін бірнеше құралдар мен көрсеткіштер пайдаланылды.

Қорытындыда диссертацияда алынған негізгі нәтижелер тұжырымдалды.

1 МАШИНАЛЫҚ АУДАРМАНЫҢ АВТОМАТТАНДЫРЫЛҒАН ПОСТ-РЕДАКЦИЯЛАУ МАҢЫЗДЫЛЫҒЫ МЕН ЕРЕКШЕЛІКТЕРІ

1.1 Машиналық аударма мен автоматтандырылған пост-редакциялау жүйесі

Машиналық оқыту қазіргі әлемге және жарқын болашаққа деген көзқарасқа айтарлықтай әсер етуде. Өзін-өзі басқаратын көліктер, смартфондардағы ақылды көмекшілер және бейне-аналитика – технологияның қаншалықты алға жылжығанының мысалдарынан байқауға болады [1-3].

Бір табиғи тілден екіншісіне машиналық аудару үрдісі – машиналық оқыту үшін тамаша тапсырма болып табылады, мұнда жоғары сапалы деректер шешуші рөл атқарады [4-6].

Ағылшын тіліндегі терминологияда ағылшынша *machine translation, MT(толық автоматты аударма)* және *machine-aided* немесе *machine-assisted translation (MAT)* (автоматтандырылған) терминдері кездеседі [7-10].

Заманауи аударма құралдары. Бүгінгі таңда Google компаниясы ең танымал машиналық аударма (МА) жүйесі болып табылады. Бұл МА жүйесі аудару жылдамдығы мен аударма сапасымен ерекшеленеді.

МА-ның тағы бір үлкен атауы - Bing компаниясының Microsoft Translator. Енгізілген мәтін тілі автоматты түрде анықталады. Аударылатын мәтінді ауызша жазу мүмкіндігі де бар [11].

Translatedict МА-сында 50-ден астам тілдер қарастырылған. Бұл МА қолданылған кезде қажет тілдің бірі таңдалынып, пайдаланушы пікірінше диалект автоматты түрде анықталады. Тек сөзді, сөз тіркесін немесе көп мөлшердегі мәтінді енгізіп, аударма тілі таңдалынған кезде қажетті *аудару* батырмасы басылады. Бұл МА ерекшелігі: ауызша айтылған мәтін жазба түрге де келтіріледі [12]. Яғни ауызша айтылған мәтін МА-ға жазылып шығарылады.

Microsoft Translator live – бұл аударма мен транскрипцияның ақысыз қызмет атқаратын аударма жүйесі, iOS, Android және тағы сол сияқты бірнеше құрылғылар арқылы мәтіндерді аударуға мүмкіндік береді [13].

Yandex.Translate (бұрын Yandex.Translation деп аталған) – бұл мәтінді немесе веб-парақтарды басқа тілге аударуға арналған Яндекс ұсынған веб-қызмет [14].

Ақысыз DeepL Translator-ді қолданып, DeepL – әлемдегі жетекші нейрондық желісінің технологиясымен қамтамасыз етілген ең жақсы МА-мен де мәтіндер аударылып жүр. Google, Microsoft және Facebook секілді атақты компанияларды DeepL деп аталатын шағын компания едәуір озып, алға шықты [15].

Негізі аударманың сапасы берілген мәтіннің тақырыбы мен стиліне байланысты болады. Көркем шығармалардың мәтіндерінің МА-сы әрқашан қанағаттанбайтын сапада болады. Дегенмен, техникалық құжаттарға мамандандырылған машиналық сөздіктер мен жүйеге аз ғана баптау жасау арқылы, аз көлемдегі редактрлық өзгертулер жасауды қажет ететін қолжетімді

сапада аударма жасауға болады. Берілген құжаттың стилі неғұрлым нысандандырылған болса, соғұрлым аударма сапасын жоғарлатуды күтуге болады. МА-ны қолдана отырып техникалық және ресми іс-қағаздар стилінде жазылған мәтіндерді аудару ең жақсы нәтижеге арқылы жетуге болады [16].

Кез-келген аударушы ұзақ уақытты жоба немесе бұрын аударылған мәтінді қайта қолдануда келісілген терминологиялық глоссарийді қолданғанда қиындыққа тап болып жатады. Табиғатына қарай, мұндай мәселелер оңай нысандандырылады және программаланады, сол себепті жұмыс орынды локализаторды автоматтандырылған құралдармен жабдықтау нормаланған сала болып табылады, ал кейбір осындай құралдар салалық стандартқа сәйкес келеді. Мұндай құралдардың басым көпшілігі аударма жады (translation memory) концепциясына сәйкес құрастырылған – әрбір жазба параллелді мәтін бірлігін құрайтын қарапайым мәліметтер қоры. Мұндай мәліметтер қоры бұрынғы аудармаларды қайта қолдану немесе мәселелерді шешу мақсатында қолдану үшін сақтайды. Аударма жадымен жабдықталғанына қарамастан, программалар автоматтандырылған аударма жүйесі деп аталады. Негізі техникалық құжаттама аудармашысы үшін берілген шарттарды қолдануда жады технологиясы кілт ретінде қолданылады. МА жүйелеріне аз орын бөлінген, өйткені олардың мүмкіндіктері шектелген және мәтінмен кәсіби түрде жұмыс істеуде қолдануға мүмкіндік бермейді. Компьютерлік қосымша аударма (CAT, computer-aided/assisted translation), яғни аударма жады өз алдына ештеңе аудара алмайды, ал МА берілген мәтіннің грамматикалық талдау нәтижесі бойынша аудару генерациясына негізделген деп аталатын программамен шатастыруға болмайды. Ережеге сәйкес, аудару жадының жазбасы екі сегменттен тұрады: бастапқы (source)тіл және соңғы (target) тіл. Егер бірдей (немесе ұқсас) сегмент бастапқы тілдегі мәтінде кездессе, соңғы тілдегі сегмент аудару жадында көрсетіледі және аудармашыға жаңа аударманың негізі ретінде ұсынылады. Егер бастапқы мәтіннің сөйлемі дерекқорда сақталған сөйлемге сәйкес келсе (дәл сәйкестік, ағылшын тіліндегі дәл сәйкестік болса), оны автоматты түрде аударуға болады. Жаңа сөйлем де дерекқордағы сақталған ретінен біршама бөлек ерекшеленуі мүмкін (дұрыс емес сәйкестік, ағылшын тілінде анық емес сәйкестік). Мұндай ұсынысты аударуға да ауыстыруға болады, бірақ аудармашыға қажет өзгерістерді енгізуге тура келеді. Қайталанатын фрагменттерді аудару үдерісін және аударылған мәтіндерге енгізілген өзгертулерді (мысалы, бағдарламалық өнімдердің жаңа нұсқалары немесе заңнамадағы өзгерістер) жеделдетуден басқа, ақпараттық жүйе терминологияны бірдей фрагменттерде біркелкі аударуды қамтамасыз етеді, бұл әсіресе техникалық аудармада маңызды болып саналады. Екінші жағынан, егер аудармашы аударма дерекқорларынан алынған аудармаларды үнемі алмастыра берсе, оларды жаңа контексте пайдалану кезінде аударылған мәтіннің сапасы нашарлауы мүмкін. Автоматты түрде табылған мәтін түзетіледі немесе толықтай қабылданбайды. Программалардың басым көпшілігі тақ сәйкестік алгоритмін қолданады, ол функционалдық мүмкіндіктерін

жақсартып, ізделініп отырған сөз тіркестері(фраза) бар сөйлемдерді табуға көмектеседі. Мұндай бағдарламалық қамтамасыз етуді қолданудың артықшылықтары бастапқыда анық болмауы мүмкін, алайда дерекқор толық болған сайын, аударманың негіздерін ауыстырудың нәтижелері дәлірек және тұрақты болады [17,18].

Автоматтандырылған аударма жүйесі компьютерлік қосымша аудармадан (CAT) ерекшеленеді, себебі аударма жадында бүкіл аудару үрдісін адам жүзеге асырады. Ал автоматтандырылған аударма жүйесі компьютер қолданушысына дайын мәтінді аз уақытта немесе сапалы түрде шығаруға көмектеседі. Көптеген автоматтандырылған аударма жүйесі кем дегенде сөздіктерді құруды және пайдалануды, параллель мәтін негізіне сүйене отырып (ағылшын тілін туралау) жаңа дерекқорларды құруды, сондай-ақ терминологияны жартылай автоматты түрде шығаруды жүзеге асыруға көмегін тигізеді [19,20]. Сондықтан да автоматтандырылған жүйені қолдана отырып, МА сапасын жақсарту мақсатында пост-редакциялаудың рөлі аз емес.

1.2 Қазіргі кездегі автоматты пост-редакциялау жүйе ерекшеліктері және қойылатын тапсырмалар

Пост-редакциялау үрдісінен кейінгі МА-ның басты артықшылығы – өнімділіктің жоғарылауы. Бұл қоданушыларға көптеген тілде мазмұнды тезірек және тиімді бағамен шығаруға көмектесетін шешім. Негізі кәсіби лингвист редакциялайтын МА құралы үлкен көлемді жобалар кезінде өте пайдалы. Алайда, МА адамның аудармасына әзірше тең келмейтінін ұмытпаған жөн [21,22].

МА-ны пост-редакциялау – бір тілден екінші тілге аударылған мәтінді аударма сапасы жағынан жақсарту үрдісі. Алайда аударманың сапасы жылдамдық пен уақыт сияқты көптеген параметрлерге тәуелді болғандықтан бірден кәсіби аудармаға қол жеткізу қиынға соғады. Жұмысты жеңілдету мақсатында аудармашыларға (қолданушыларға) сапалы материалдардың жылдамдығын қамтамасыз етуге көмектесетін компьютерлік қосымша аударма (CAT) құралдарын қолдану ұсынылады [23].

МА-ның ең жаңа буыны – бұл нейрондық машиналық аударма (NMT). Көптеген ғылыми жұмыстарда NMT аударма нәтижелерінің сапасы мен дәлдік деңгейі ескеріліп, кәсіби аудармаларды жақсы шығара бастады. Сонымен қатар, көлемді мәтіндерді тезірек аудару үшін нейрондық МА жақсы құрал болып табылады. Бірақ нейрондық МА нәтижелерінде кездесетін қателерді түзету үшін кәсіби аудармашының да көмегі керек. Бұл пост-редакцияланған МА (MTPE- Machine Translation of Post-editing) үрдісі деп аталады. Бұл қызмет түрін әдетте қажет тілді жақсы меңгерген лингвисттер, редакторлар мен корректорлар жасайды. Нейрондық МА-ға негізделген пост-редакциялау тәсілдерін қолданған көптеген жұмыстар бар. Бірақ тестілеу жүргізілген кезде мәліметтер қорында түрлі терминалогиялар жалқы есімдер (тілдегі ерекшелік)

кездессе қиындықтарға тап болуы мүмкін. Нейрондық МА-ны қолданылмас бұрын әртүрлі аспектілер ескеріледі: аударылатын мәтіннің анықтығы; белгілі бір нақты секторлармен (заңды, медициналық және т.б.) жұмыс істеу. Сонымен қатар, бұл тәсіл өте үлкен корпустарды қажет етеді [24].

Жыл сайын МА WMT конференциясында автоматты пост-редакциялау тапсырмасы қарастырылады [25]. Джейкоб Мундт (Jacob Mundt et al. 2012) [26] «Learning to Automatically Post-Edit Dropped Words in MT» мақаласында автоматты пост-редакциялау (APE) МА нәтижелерінің дұрыстығын қайта енгізу үрдісі арқылы жақсарта алды. Бірақ сөздер енгізу орынын ескеру өте маңызды. Бұл жұмыста белгілі бір тілдер мен МА жүйелері үшін қайта енгізу ережелерін үйренудің ықтималдық тәсілі ұсынылды. Сондай-ақ аудармалардан дайын деректерді синтездеу әдісі сипатталды. Қытай тілінен ағылшын тіліне және араб тілінен ағылшын тіліне арналған МА жүйелері үшін енгізу логикасы тексерілген. Нәтижесінде нейрондық МА тәсілімен бейімделген APE жүйесіндегі үш сөз арасындағы жеткіліктілік көрсеткіші араб-ағылшынша МА мәтінінде 73%, ал қытай-ағылшынша МА мәтінінде 67% жетті. Ережеге негізделген енгізу тәсілімен салыстырыла отырып автоматтандырылған жеткіліктілік көрсеткіштері бойынша жақсартылған өнім ұсынылды. NMT проблемасының нақты аспектілері мен оны машиналық оқыту шешімдеріне қолдану тиімділігі қарастырылған.

Интерактивті МА-мен тәжірибе жасалған алғашқы жүйелер статистикалық МА технологияларымен байланысты болды (Simard and Foster 2013) [27]. Олардың кейбіреулері онлайн оқытуды пост-редакциялау үрдісінде бейімделген сөйлемдермен біріктірілді (Ortiz-Martínez and Casacuberta 2014; Lagarda et al. 2015) [28,29]. Чатержи (Chatterjee et al. 2017) [30] әртүрлі домендердегі деректерден ең жақсы түзетулерді таңдай алатын онлайн аналитикалық өңдеу жүйесін (OLAPE) қолданды. Дәлірек айтқанда, OLAPE жүйесінің екі түрі қолданылды. Сондай-ақ өңдеу тапсырмасын жеңілдететін CAT жүйелері де қарастырылды, бірақ оларды қолдану үшін үлкен аударма жадыны және көп еңбекті қажет етеді.

Сондай-ақ, Сантану Пал және т.б. (Santanu Pal et al. 2016) [31] МА-ның нәтижесін жақсарту үшін нейрондық желіге негізделген автоматты пост-редакциялау (APE) жүйесін ұсынған. Олардың APE (NNAPE) нейрондық үлгісі екі бағытты қайталанатын нейрондық желі (RNN) үлгісіне негізделген және МА нәтижесін бекітілген ұзындық векторына кодтайтын кодтауыштан тұрды. Оның дешифраторы пост-редакцияланған толық сөйлем аудармасына қолданылды.

«Log-linear Combinations of Monolingual and Bilingual Neural Machine Translation Models for Automatic Post-Editing» мақаласында Марсин Юнчис-Даумунт (Marcin Junczys-Dowmunt et al. 2016) [32] Адам Мицкевич атындағы университеттің ғалымдар тобы автоматты түрде пост-редакциялау жұмысы туралы сипаттады. Олар нейрондық аударма үлгілерін қолдануды зерттеді. APE проблемасында жақсы нәтижелерге қол жеткізу мақсатында әртүрлі үлгілерді компоненттер ретінде қарастырып, бірнеше кіріс сөйлемдерге мүмкіндік

беретін логикалық-сызықтық үлгі декодталған. Ішіне біріктірілген, қарапайым жолға сәйкес келетін сызықтық үлгі басқару үшін қолданылды. Зерттеу нәтижелерінде бағалау метрикалары 3,2% TER және + 5,5% BLEU бойынша кез-келген басқа жүйеден жақсырақ көрсеткіш көрсетіп, бастапқы деңгейге қарағанда жоғары нәтижелерге қол жеткізгендері туралы айтылған.

МА мен АЖ-ні біріктірумен тығыз байланысты тағы бір тұжырымдама – интерактивті МА жүйелерін дамыту болып табылады. Торрегроса және т.б. (Torregrosa et al. 2017) [33] интерактивті МА-ға көзқарасы екітілді ақпараттың жаңа көздерін үздіксіз дерлік қосуға мүмкіндік береді деп санады. Олар ағылшын –испан, араб –ағылшын және ағылшын-қытай сияқты туыс емес тілдер арасындағы аударма тапсырмаларын автоматты түрде бағалау арқылы шыны жәшік (glass-box) пен қара жәшік (black-box) тәсілдерін алғаш рет салыстырды.

Крис Хокамп және т.б.(Chris Hokamp et al. 2017) [34] “Ensembling Factored Neural Machine Translation Models for Automatic Post-Editing and Quality Estimation” деген еңбегінде мамандандырылған нейрондық МА-ның құралдарын пайдаланып, автоматты пост-редакциялауға (APE) және сөз сапасын бағалауға (QE) жаңа тәсіл ұсынған. Сапаны бағалау (QE) үшін тиімді болатын сөз деңгейіндегі функциялар кіріс факторлары ретінде қолданылған. Жұмыстың негізгі идеясы – автоматтандырылған өңделген гипотезаны құру үшін пайдаланылатын МА гипотезасын қолдану. Ол үшін әртүрлі мәліметтер қорын қолданатын NMT нейрондық МА үлгілерінің жиынтығы дайындалды. Барлық үлгілер біріктіріліп, APE тапсырмасы үшін бағалау сапасы QE есептелінді. Осылайша олар ең озық тәсілдерді APE және бағалау сапасын бірыңғай құрылымда байланыстыруға тырысты.

Негри және т.б. (Negri et al. 2018) [35] NMT нәтижесін шығару үшін OL [онлайн оқыту] үрдісін APE жүйелеріне қолданып жүйелерді үздіксіз біріктірді. Нейрондық МА үшін МА жүйелерін нөлден бастап дәл баптау немесе қайта оқыту қажеттілігін жою үшін ұсынылған жүйелердің пайдалылығын атап өткен.

2018 жылы APE тапсырмасы екі автоматты пост-редакциялау қосалқы тапсырмадан тұрды, бірі фразаға негізделген статистикалық МА-ға (PBSMT) және екіншісі ағылшын – неміс тілдеріне және басқа тілдерге арналған нейрондық машина аудармасына (NMT) арналған. Чаттержи және т.б. (Chatterjee et al. 2018) [36] ғылыми топ осы МА сынағын жүргізіп, кез келген APE жүйесі үшін маңызды екі аспектіге назар аударды: жылдам үйрену және бастапқы мәтінді қарастыру. APE жүйесі өнімділігін үнемі жақсарту үшін пост-редакциялау кезінде деректердің үлкен көлемінен жылдам үйренуге қабілетті болуы керектігі ескерілді. Бұрынғы APE жүйелері бір тілдік тәсілді ұстанғанымен бәрібір соңында МА нәтижесіне назар аударылды. Дегенмен де бастапқы мәтінді де қарастыру арқылы жақсы нәтижелерге қол жеткізілді. Шынында да, МА нәтижесіндегі(output) кейбір қателерді бастапқы мәтіндерде(source text) қарамай анықтау мүмкін емес. Яғни, сөйлемдердегі

қателерді анықтау үшін алтын стандарт қажет, оны салыстыру үшін де қолдануға болады.

Сондай-ақ, Негри тобының жұмысы (Negri et al. 2018) [37] пайдаланушыны қадамдық оқыту арқылы үздіксіз дамыту үшін желіде жұмыс істейтін нейрондық жүйелерді дамытуға бағытталған. Бұл жұмыстың мақсаты – APE нейрондық үлгісінің параметрлерін жылдам жаңарту үшін қайта оқуды тоқтатпай қолданушы өңдеу жұмыстарын пайдалану болды. Бұл нейрондық желіге негізделген пост-редакциялау тәсілдері туралы көптеген басқа жұмыстардың біршама бөлігін қамтиды.

Компьютерлік қосымша аударма (CAT) құралдары пост-редакциялау жұмысын жеңілдетеді. МА-ның сапасын жақсарту мен аударма жадысын (ТМ) біріктіру мақсатында қол жеткізуге болады, бірақ Ортега және т.б. (Ortega et al. 2019) [38] жұмыстарында көрсетілгендей тек кәсіби аударма ғана емес бағалау сапасы да ескеріледі.

Антонио жұмысында (Antonio Gois т.б. 2020) [39] қолданушылардың сұранысы бойынша аудармаларды монотонды емес автоматты түрде пост-редакциялау тәсілі арқылы зерттеді. Олардың көзқарасы бойынша алынған жүйе солдан бастап оңға және пост-редакциялау кезінде кездейсоқ ретпен оқытылатын жүйемен салыстырылды. Ұсынылған тәсілге қосымша аудармаларды автоматты түрде пост-редакциялау әдісін өңдеуді үйренетін BERT құралы көмегімен алдын ала дайындалған Transformer негізіндегі үлгіні қосымша ұсынды. Олар бұл үлгіні үш түрлі жолмен зерттеді: адам көмегімен, солдан оңға қарай реттілік немесе кездейсоқ кездесу реттілігін қолданды. Барлық үш өлшемді қолдану нәтижесінде үлгі түзетілмеген МА-ның базалық деңгейінен жоғары көрсеткіш көрсетті. (Berard et al., 2017) [40]. Үлгіде қарастырылған үш жағдай ұсынылған параметрлермен оқытылды.

Феликс тобы бастаған тәсіл (Félix do Carmo et al. 2020) [41] төрт өңдеу жұмысына негізделген пост-редакциялауды автоматты аяқталған функциясы бар аудармамен салыстырды. Өңдеу кезінде негізгі төрт әрекет таңдалады: сөздерді жою, кірістіру, жылжыту және ауыстыру. Ағылшын тілінен еуропалық португал тіліне автоматты түрде аударылған төрт мәтін төрт түрлі сессияда өңделді, онда әрбір аудармашы мәтіндері мен жұмыс режимі ауыстырылып отырды. Жұмыс режимдерінің бірі әдеттегі автотолтыру мүмкіндігін қамтыса, екіншісі төрт әрекетке негізделген. Қатысушылар семинарға дейін, семинар барысында және одан кейін сауалнамаларға жауап берді. Тәжірибелер кезінде жазылған сауалнама жауаптары мен журналдарға сипаттамалық талдау жүргізілді. Төрт әрекетті өңдеу режимі анағұрлым нәтижелі болып көрінеді, жоспарланған шешімдерді қабылдауға мүмкіндік береді. Бұл режим көп уақытты алса да, аудармашылар аз түзетулер жасайды. Өңдеу мәселесін зерттеу үшін бұл тәсілдің пайдалылығы көрсетілген. Бұл әдісті қолдану өңдеу процесінде ыңғайлы, бірақ автоматтандырылған пост-редакциялаудағы төрт қадамды ескеру тереңірек талдауды қажет етеді, әсіресе бұл тәсілдің авторлары көп уақытты қажет ететіндігін айтты.

МА жинағында APE-ге қатысты көптеген халықаралық тапсырмалардың нәтижелері көрсетілген. Мысалы, Чаттержи және т.б. (Chatterjee et al. 2020) [42] WMT APE конкурсының алтыншы раундындағы нәтижелерді ұсынды. Бұнда қатысушылар әртүрлі сөйлемдердің ішінде қолданушы түзетулерін үйрену арқылы «қара жәшік» МА жүйесінің нәтижесін түзету үшін APE-ні қолдануға тырысты. Басқа жағдайда, Ян және т.б. (Yang et al. 2020) [43] дайын NMT үлгілерін аз мөлшердегі APE корпусымен баптау кезінде МА-ның өнімділігін арттыра отырып әдісті жақсартуға болатынын көрсетуге тырысты.

Шарма және т.б. (Sharma et al. 2021) [44] автоматты пост-редакциялау үшін нейрондық машина аудармасын бейімдеген. Автоматты пост-редакциялау үлгілері қолданушының пост-редакциялау үлгілерін үйрену арқылы МА жүйесінің шығысын(output) түзету үшін пайдаланылды. Олар WikiMatrix құралын (Schwenk және т.б., 2021) [45] пайдаланып, МА үлгісін доменге бұрынғы жалпы тапсырма конференцияларындағы (WMT-16, 17, 18) қосымша APE мысалдары арқылы бейімдеді және үлгілерді біріктірді. Бұл тапсырма үшін олар заманауи МА жүйесін пайдаланды. Әрі қарай жақсарту үшін олар WikiMatrix көмегімен МА үлгісін доменге бейімдеді. Содан кейін жалпы конференциядағы алдыңғы шығарылымдарынан (WMT-16,17,18) қосымша APE үлгілерімен дәл баптады және үлгілерді біріктірді. Олар 2021 жылы конференция жинағына ағылшын – неміс WMT бірлескен автоматты пост-редакциялау жүйесін ұсынды.

Көптеген мақалалар ағылшын – қазақ тіліне емес, ағылшын – неміс тіліне және басқа тіл жұптарының жұмыстарын сипаттайды. 2021 жылға дейін ағылшын – қазақ жұбына арналған пост-редакциялау жұмыстары қарастырылмады. Тек соңғы жылы Рахимова Д.Р. бастаған ғылыми топ «The Development of the Light Post-editing Module for English-Kazakh Translation» мақаласында машиналық оқытуды қарастыра бастады (Rakhimova D etc. 2021) [46].

Жоғарыда келтірілген салыстыруларды ескере отырып, бұл жұмыста ұсынылған пост-редакциялау технологиясының (PE-LC) ерекшелігі нейрондық желілерге негізделген әдістегідей өте үлкен деректерді қажет етпейді. CAT(аударма жады) жүйесі үшін де үлкен көлемді мәліметтер қоры қажет. Қолданушы үшін бұл жақсы құрал, бірақ аударма жады ресурсын қажет етеді, сондықтан да қазақ тіліндегідей ресурсы төмен тілдер үшін қолдану оңай емес.

Жоғарыдағы айтылған салыстыруларды ескере отырып, бұл зерттеу жұмысы үшін нейрондық желілер негізіндегі әдіс сияқты өте үлкен деректер қажет етпейді. CAT жүйесі жақсы құрал болғанымен, ол көбірек аударма жадысын қажет етеді. Ал терең талдау үшін, үлкен көлемді деректер қажет және негізінен бұл әдіс нейрондық желі әдістеріне қолданылады.

Қазігі уақытқа дейін қазақ тіліне арналған толық APE қарастырылмаған.

МА-ның сапасы, тіпті нейромашиналық аударманың әсерлі нәтижелерімен де, кәсіби аударманың сапасына әлі жеткен жоқ. Дәлірек айтсақ, Баоның (Bao 2015) [47] еңбегінде лексикалық қателердің қанша түрі

анықталғанын көруге болады. Корпустан іздеу кезінде 633 лексикалық қате, оның ішінде сөздердің орын тәртібінің 17 қатесі, 116 сөз табының қатесі, ауыстырудың 209 қатесі, 96 эллипс қатесі, 100 артық қате, 3 қайталанатын қате, 92 семантикалық көпмағыналық қателер анықталды. Ол жиілігі жоғары (ең көп кездесетін) лексикалық қателердің 4 түрін ғана зерттеген. Бұл көлемді және көп уақытты қажет ететін тапсырма. Сондықтан мәселенің бір бөлігін шешу үшін бұл ұсынылып отырған жұмыста пост-редакциялау кезінде тек лексикалық таңдауды қажет ететін қателер, яғни ауыстыру қателері (substitution) қарастырылды.

Бұл ғылыми жұмыста МА-ның автоматты пост-редакциялау (APE) әдісі қарастырылған. Дәлірек айтқанда, лексикалық таңдау әдісін пост-редакциялауға үрдісіне қолданылды, яғни ағылшын тілінен қазақ тіліне аударылған мәтіннің дұрыс емес болу мүмкіндігіне назар аударылды.

Пост-редакциялау үрдісінің бұл жұмыста маңызды бөлігінің бірі *лексикалық таңдау* әдісі қате сөздерді түзету кезінде сөз мағынасын ескереді. Аударма нәтижесінде алынған қазақ тіліндегі сөйлемде қате сөздер кездескен кезде лексикалық таңдау әдісі сөз мағынасын ескеріп сөйлемге қатысты дұрыс сөз таңдауға көмегін тигізеді. Яғни, бұл ғылыми зерттеу жұмыс ағылшын – қазақ тіліндегі МА-мен аударылған мәтіндегі лексикалық таңдау қателерін автоматты түрде пост-редакциялау әдісін зерттеуге бағытталған.

Сонымен, лексикалық таңдау әдісіне негізделіп келесі тапсырмалар қойылады:

- мақсатты (қазақ тілінде) сөйлемдердегі қате аударылған сөздерді анықтау;
- қазақ тіліндегі қате аударылған сөздердің синонимдер каталогын автоматты түрде қалыптастыру;
- қате аударылған сөздің ықтималдығы жоғары синоним сөзді таңдап семантикалық текше әдісінің үлгісі мен алгоритмін құру;
- ағылшын тілінен қазақ тіліне аударылған МА мәтінін PE-LC пост-редакциялау (Post Edit - Lexical Choice) технологиясын құру.

Бірінші тараудың қорытындысы

МА мен оның түрлері жайлы жазылды. Қазіргі кезде жиі қолданылып жүрген Google, Translatedict, Bing компаниясы ұсынған Microsoft Translator, Microsoft Translator live , Yandex Translate, Translator DeepL заманауи аударма құралдарына шолу жасалды.

Жыл сайын өтетін WMT конференциясы APE тапсырмасын қамтитын соңғы, жаңа еңбектеріне шолу жасалды.

Машиналық пост-редакциялау туралы және оның маңыздылығы жайлы айтылып, қазіргі кездегі автоматты пост-редакциялау жұмыстары мен ерекшеліктері туралы айтылды.

Шолу, талдау және лексикалық таңдау есебі негізінде зерттеу тапсырмалары қойылды.

2 ЛЕКСИКАЛЫҚ ТАҢДАУ НЕГІЗІНДЕГІ (PE-LC) ПОСТ-РЕДАКЦИЯЛАУ ТЕХНОЛОГИЯСЫ

2.1 PE-LC пост-редакциялау технологиясының құрылымы мен алгоритмі

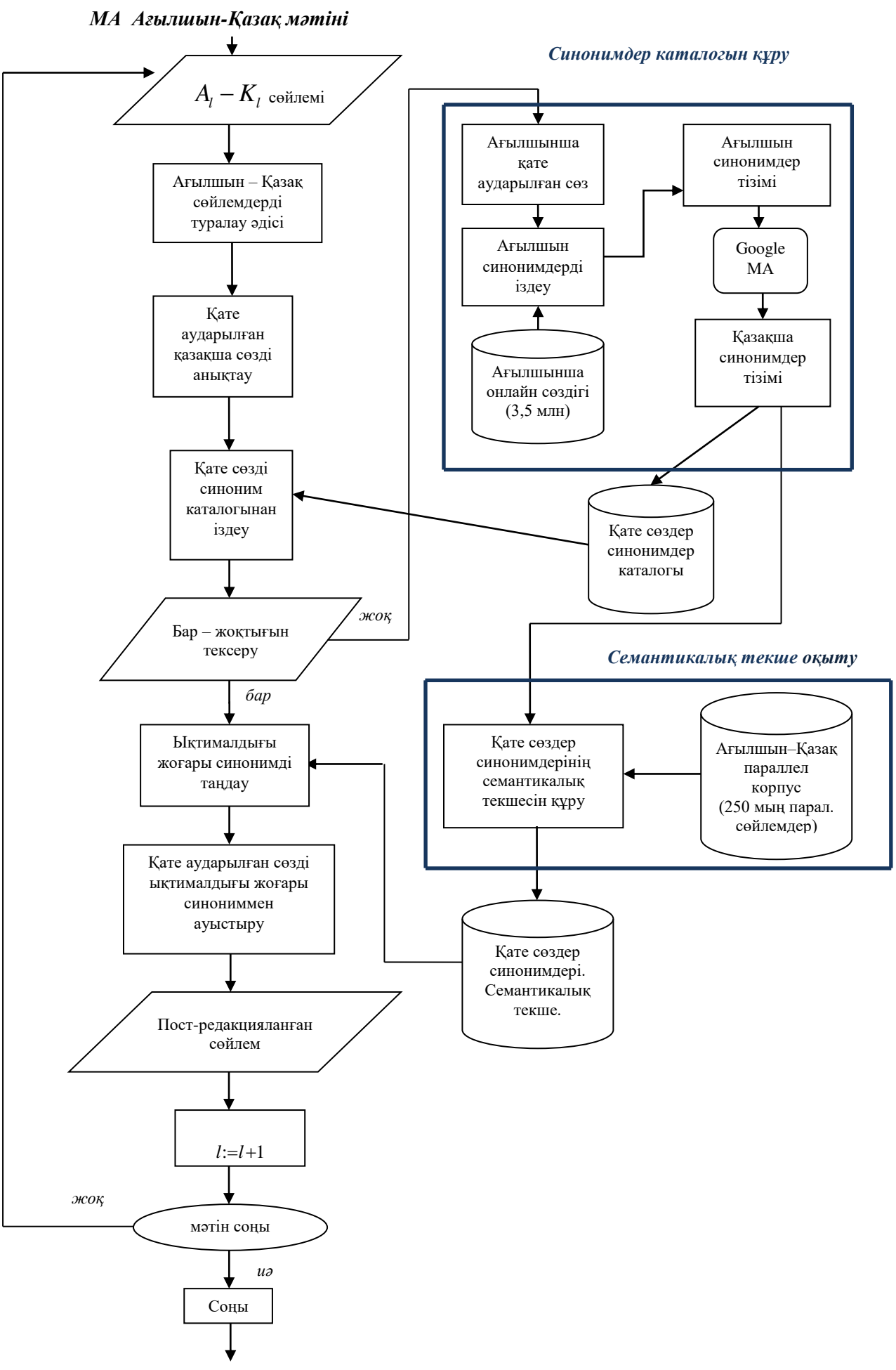
Ұсынылған APE технологиясы үш тапсырманы қамтиды: (1) қазақ тіліндегі сөйлемнің қате аударылған сөздерін анықтау; (2) қате аударылған сөздердің синонимдер каталогын автоматты түрде қалыптастыру және (3) қате аударылған сөзді мағынасы жағынан жақын синониммен ауыстыру; (4) жоғарыдағы үш тапсырманы біріктіріп пост-редакциялау технологиясын құрастыру.

Бірінші тапсырмада МА-дағы қазақша сөйлемдегі қате аударылған сөздерді анықтау үшін Esplà-Gomis және т.б. (2012) [48], Esplà-Gomis және т.б. (2015) [49] еңбектері қолданылып, аударма жадының орнына кері аудару әдісін қолдану арқылы оны жақсарту ұсынылды. Толық ақпарат 3.2 бөлімінде сипатталған.

Екінші тапсырмада лексикалық таңдау негізінде синонимдердің каталогын қалыптастыру автоматты түрде жасалды. Мақсатты сөйлемнен қате аударылған сөздер анықталғаннан кейін бұл сөздер каталогтан іздестіріледі. Егер олар каталогта болмаса, каталогқа қосуға болатын жаңа тізім жасалады. Бұл синонимдердің тізімін біртіндеп қалыптастыру болып табылады. Ағылшын тіліндегі қате аударылған сөзді алып, *thesaurus.com* онлайн сайтынан ағылшын тіліндегі синонимдерді тауып, Google Translate арқылы олардың қазақ тіліне аудармасын құрастыру арқылы тізім жасалды. Осылайша, аударма синонимдерінің тізімі автоматты түрде қалыптастырылды. Бұл үрдістің толығырақ мәліметтері 4.2 бөлімінде берілген.

Үшінші тапсырма үшін Төкеев және т.б. ұсынған (2017) [50] семантикалық текше үлгісі деп аталатын әдіс қолданылды. Бұл үлгі Тайерс және т.б. (2015) [51] енгізген максималды-энтропия әдісіне негізделген. Төкеев және т.б. (2017) кейінірек зерттеді. Тайерс және т.б. (2015) ағымдағы сөздің сол жағындағы екі сөздің және оң жағындағы екі сөздің бірізді тіркесін қолданады, ал семантикалық текшеде ағымдағы сөздің контексті ретінде бүкіл сөйлем үшін ретімен қолданылады. Толық ақпарат 5.1 бөлімде қарастырылған. Төртінші тапсырмада аталған үш тапсырма біріктіріліп пост-редакциялау технологиясы құрастырылды.

Аталған төрт тапсырма ғылыми жұмыстың ұсынылып отырған PE-LC пост-редакциялау технологиясының сұлбасында сипатталған (2.1-сурет).


МА Ағылшын-Қазақ мәтіні
$A_l - K_l$ сөйлемі
Ағылшын – Қазақ сөйлемдерді туралау әдісі
Қате аударылған қазақша сөзді анықтау
Қате сөзді синоним каталогынан іздеу
Бар – жоқтығын тексеру
жоқ
бар
Ықтималдығы жоғары синонимді таңдау
Қате аударылған сөзді ықтималдығы жоғары синониммен ауыстыру
Пост-редакцияланған сөйлем
l:=l+1
жоқ
мәтін соңы
иә
Соңы
Түзетілген МА қазақ мәтіні
Синонимдер каталогын құру
Ағылшынша қате аударылған сөз
Ағылшын синонимдер тізімі
Ағылшын синонимдерді іздеу
Google MA
Ағылшынша онлайн сөздігі (3,5 млн)
Қазақша синонимдер тізімі
Қате сөздер синонимдер каталогы
Семантикалық текше оқыту
Қате сөздер синонимдерінің семантикалық текшесін құру
Ағылшын–Қазақ параллел корпус (250 мың парал. сөйлемдер)
Қате сөздер синонимдері. Семантикалық текше.

Сурет 2.1 – PE-LC пост-редакциялау технологиясының сұлбасы

Жоғарыда аталған үш тапсырманың PE-LC технологиясының алгоритмдері:

Алгоритм 1. Негізгі алгоритм.

1.Кіріс деректер: $A_l \leftrightarrow K_l$ (ағылшын – қазақ жұп мәтіні, $l=1$) N – параллель жұп сойлемдер саны; A_l – ағылшынша l-сөйлемі, K_l – ағылшыннан аударылған қазақша l-сөйлемі;

2. Ағылшын мен қазақ тіліндегі сөйлемдердің сөздер мен сөз-тіркестерін (3-грамм) туралау мен G туралау коэффициентін есептеу;

3. Егер $G > 0.5$, онда 5-қадамға өту, басқа жағдайда 10-қадам орындалады.

4. Қате аударылған сөздер синонимін табу (каталогтан іздеу), синоним бар болса ықтималдығы жоғары синонимді таңдау үшін 5-қадамға өту керек, жоқ болса ***Алгоритм 2*** мен ***Алгоритм 3*** орындалады.

5. $w_j^{каз}$ қате аударылған қазақ сөздің ықтималдығы ең жоғары (*max*) синонимін таңдау .

6. K_l сөйлеміндегі қате аударылған қазақ сөзінің ықтималдығы max синоним мәнімен ауыстыру.

7. Пост-редакцияланған сөйлем шығару

8. $l := l + 1$;

9. егер $l > N$, онда 10-қадамға өту, басқа жағдайда келесі ағылшын – қазақ жұп мәтінін енгізу үшін 1-қадамға өту.

10.Соңы.

Алгоритм 2. Қате аударылған сөздің синонимдерін табу алгоритмі.

2.1 Кіріс деректер: $w_j^{каз}$ - қазақ қате аударылған сөздер, $w_i^{анг}$ - қазақ тіліне сәйкес ағылшын сөзі.

2.2 Синоним каталогында $w_j^{каз}$ іздеу.

2.3 Егер $w_j^{каз}$ каталогта болса, $w_j^{каз}$ синонимдер тізімін шығару үшін 5-қадамға өту, кері жағдайда келесі 2.4-қадам орындалады.

2.4 Ағылшын синоним сөзінен $w_i^{анг}$ іздеу. (3,5 млн. сөз) [*https://www.thesaurus.com/*]

2.5 $w_i^{анг}$ синонимдерін Google Traslate-пен қазақ тіліне аудару;

2.6 Жаңа жолда $w_j^{каз}$ - қазақ синонимдерімен каталогты толықтыру (кеңейтілуі). Синонимдер тізімін шығару. Алгоритм 3-ке өту.

Алгоритм 3. Қате аударылған қазақ сөзі үшін семантикалық текше құру алгоритмі.

3.1 Алгоритм 2 - ден алынған кіріс дерегі: қате аударылған қазақ сөзінің синонимдер тізімі. Жалпы қолданатын дерек: ағылшын – қазақ жұп корпусы (250 мың сөйлем).

3.2 Қате аударылған қазақ сөзіне семантикалық текшеде бөлек контекст кестесін синонимдер тізімімен құру.

3.3 Ағылшын – қазақ жұп корпусын қолданып максималды энтропия оқыту әдісі арқылы қате аударылған сөздің контекст кестесін алдымен контекст жиіліктермен толтыру.

3.4 Қате аударылған сөздің жиіліктермен толтырылған контекст кестесін максималды энтропия формуласы арқылы ықтималдықтарға түрлендіру.

Екінші тараудың қорытындысы

Бұл екінші тарауда қарастырылып отырған ғылыми жұмыстың PE-LC пост-редакциялау технологиясының құрылымы мен алгоритмі сипатталды. Дәлірек айтсақ:

- ғылыми жұмыстың негізгі үш тапсырмасына сипаттама жасалды.
- ғылыми жұмыста ұсынылған PE-LC технологиясының сұлбасы келтірілді.
- PE-LC технологиясының алгоритмдері сипатталды.

3 ҚАЗАҚ ТІЛІНДЕГІ СӨЙЛЕМНІҢ ҚАТЕ АУДАРЫЛҒАН СӨЗДЕРІН АНЫҚТАУ

3.1 Қазақ тіліндегі сөйлемнің қате аударылған сөздерін жетілдірілген әдіс арқылы анықтау

PE-LC технологиясының бірінші тапсырмасы ағылшын тілінен қазақ тіліне аударылған мәтіндегі қате аударылған сөзді автоматты түрде іздеу болып табылады. Бұл сөздерді табу үшін біріктірілген тәсіл қолданылды. Бұл біріктірілген әдіс туралау тұрақтылығына, сондай-ақ кері аудармаға негізделген қате сөзді автоматты түрде табу әдісі арқылы жүзеге асырылады. Ол үшін Esplà-Gomis(2012) және т.б. сипаттаған қате сөзді автоматты түрде табу әдісі қолданылды. Бұл үрдісте бастапқы *S* сөйлемнің бір, екі және т.б., сөздерінен тұратын сөз тіркестерін аударып және оларды аударылған мақсатты сөйлемнен табу үшін МА пайдаланылды. Қарастырылып отырған әдіс *S* сөйлем ішіндегі барлық *i* позициялары мен *T* сөйлем ішіндегі *j* позициялардың барлығы арасындағы *туралау тұрақтылығын* (*alignment strength*) анықтайды. Сонда бұл МА жүйесінде ағылшын – қазақ жұбының көптеген шағын сегменттері аударылады. Бұл жұмыста сөз тіркестерін аударған кезде униграммалар, биграммалар, триграммалар қарастырылды. Сонда n=1 болған кезде униграмма қарастырылады. Яғни ағылшын тіліндегі сөз тіркестерінің ұзындығы 1-ге тең. Ал n = 2 болғанда сөз тіркесінің ұзындығы 2-ге, n = 3 болғанда сөз тіркестерінің ұзындығы 3-ке тең. Бұл ғылыми жұмыста сөз тіркестерінің ұзындығы үшке тең болған жағдайды қарастырады. Негізі n мәніне шектеу жоқ, тіл ерекшелігін ескеріліп қалаған сөз тіркестер ұзындықтары қарастырыла береді.

Эспла-Гомис және т.б. (2012) қате аударылған сөздерді анықтау үшін туралау әдісін қолданды. Эспла-Гомис (және т.б. 2012) еңбегінде көрсетілгендей, *туралау салмағы* (*weight alignment*) келесі формула бойынша есептелді:

$$A_{jk}(S,T,M) = \sum_{(\sigma,\tau)} \frac{\mathrm{cov}\,er(j,k,\sigma,\tau)}{|\sigma| * |\tau|} \qquad (3.1)$$

мұндағы A_{jk} – *S* сөйлем саб-сегментіндегі (бастапқы сегмент) *j*-ші сөз бен *T* сөйлем саб-сегметіндегі (мақсатты сегмент) *k*-ші сөз арасындағы туралау салмағы ; *M* – *S* және *T* параллель саб-сегменттерінің жұбы үшін анықталған SSA саб-сегментті туралау (sub-segment alignments) жиынтығы, | σ | – *S* сөйлемінен қарастырылған саб-сегменттің ұзындығы. | τ | — *T* сөйлеміндегі қарастырылатын саб-сегменттің ұзындығы; (cover(*j, k, σ, τ*)) – σ-ның *S* тіліндегі *j*-ші сөз бен τ-дың *T* тіліндегі *k*-ші сөз кездесуі (қамтылуы): егер кездессе 1-ге тең, ал басқаша жағдайда 0-ге тең болады. Бұл формула $A_{1,1}$ және $A_{1,2}$ және т.б. ішкі саб-сегменттердің әрбір жұбы үшін туралау салмақтарын есептейді.

Есептеліп шығарылған мысалдар төменде толығырақ көрсетілген. Сөйлемдерді аудару үшін Google Translator[1] пайдаланылды.

S және *T* сөйлемдерін туралау үшін келесі ереже қолданылды: егер $A_{jk} > 0 \wedge A_{jk} \geq A_{jl}, \forall l \in [1, |T|]$ шарты орындалса, *S* сөйлеміндегі *j*-шы сөз бен *T* сөйлеміндегі *k*-шы сөзді тураланады, ал кері жағдайда орындалмайды.

[3.1] формуласында қолданылатын **туралау салмақтарының қосындысын** табу үшін *T* сөйлеміндегі *j*-ші сөз бен *S* сөйлемдегі *k*-ші сөз арасындағы салмақтарының қосындысы қолданылады және келесі (3.2)-формула ретінде анықталады:

$$A(j,k,S,T) = \sum_{m=1}^{L^S} \sum_{n=1}^{L^T} \frac{st\,\mathrm{cov}\,er(j,k,seg_m(S),seg_n(T),M)}{m*n} \qquad (3.2)$$

Мұнда L^S, L^T – *S* және *T* сөйлемінің ұзындығы; *M* – *S* және *T* сөйлемінің SSA жиынтығы; $seg_m(S)$ – *S* сөйлемінің сегменттер саны; $seg_n(T)$ – *T* сөйлемдегі сегменттер саны болып табылады. Толығырақ мағлұматтарды 3.1 кестедегі барлық есептеу нәтижелерінен көруге болады.

Алайда, Эспла-Гомис және т.б. (2012) МА-мен саб-сегменттерді туралау кезінде берілген ағылшын тіліндегі мәтіннің түпнұсқасы қолданылды. Ұсынылған жұмыстың ерекшелігі берілген сөйлемдердегі сөздердің тек түбірлері қолданылды. Эксперименттердегі мәтіндердің леммалары (түбірлері) Apertium платформасы [52-55] арқылы табылды. Платформа қазақ тіл мен ағылшын тіліне де қолданылды. Сөздердің леммаларын қолданудың себебі басқа сөздермен салыстырғанда кездесетін тіпті бір таңбаның сәйкес келмеуіне жол бермеу болды, өйткені қазақ тілінің жалғаулары сөздің мағынасын айтарлықтай өзгертетін тіл. Әсіресе, қазақ тілі – агглютинативті тіл, бір символдың өзі мәселені қиындатуы мүмкін. Қолданылған екі атаудың (сөздер және леммалар) ішінен лемманың зерттеу нәтижесіндегі көрсеткіштері жақсырақ болды.

Әрі қарай, туралау тұрақтылығын Эспла-Гомис және т.б. (2015) мақсатты сөйлемдегі *j* позициясының *S* және *S′* (*G* функциясы) арасындағы сәйкес келмейтін сөздің *i* позициясына сәйкес келетінін анықтау үшін пайдаланды. Сәйкес келмейтін *i* позициясын содан кейін Эспла-Гомис (2012) және т.б. әзірлеген әдіске сәйкес алдыңғы қадамда көрсетілген туралау тұрақтылығын пайдаланып *T* мақсатты сөйлемге қолданды.

Қате сөзді автоматты түрде табу әдісін қолдану үшін сөздерді туралау салмақтарының қосындысы *A(j,k,S,T)* үлесін есептейтін *G (k, S′, S, T)* функциясы пайдаланылды. Мұнда t_k және s_j *S* сөйлеміндегі барлық сөздер үшін **туралау тұрақтылығының қосындысы** (*G* функциясы) бойынша *S′* және *S* сөйлемдер арасындағы сәйкестіктің бөлігі болып табылады:

[1] https://translate.google.kz/?hl=ru&sl=en&tl=kk&text=I%20have%20a%20nice%20baby%0A&op=translate

$$G(k,S',S,T)=\frac{\sum_{k=1}^{|S|}A(j,k,S,T)*match(j,S',S)}{\sum_{k=1}^{|S|}A(j,k,S,T)} \quad (3.3)$$

Келесі қадам *S* және *S'* сөйлем арасындағы сәйкес келмейтін сөз(дер)ге тура сәйкес келетін *T* сөзді табу болды. Содан кейін қате сөзді автоматты түрде табу әдісі келесідей шарттардан тұрды. Егер *G (k, S', S, T)* ≤1/2 болса, онда бұл сөзді өзгерту қажет екені көрсетілген. Кері жағайда сақтау керек деп белгіленді.

Қате сөздерді анықтау үшін ұсынылған әдісте – қате сөздерді автоматты түрде табу әдісі кері аударма әдісімен біріктірілген. Кері аударма *G (k, S', S, T)* функциясы үшін (3.3) формуласында қолданылады. *A(j,k,S,T)* функциясы салмақтарының қосындысын есептегеннен кейін *S* және *S'* сөйлеміндегі сөздердің сәйкестігі іздестіріледі. Сәйкестік болса мән 1-ге, кері жағдайда 0-ге тең.

Жұмыстың мұндағы жаңа ғылыми үлесі (3.3) формуласын есептеу үшін қате сөзді автоматты түрде табу әдісі үшін кері аударманы қолдану болып табылады.

3.2. Қате аударылған қазақ тілдегі сөзді табу алгоритмі мен мысалы

Қазақ тіліндегі сөйлемнің қате аударылған сөздерін анықтау үшін келесі алгоритм 1 жүзеге асырады.

Қате аударылған қазақ тілдегі сөзді табу алгоритмі.

1. Кіріс деректер: $A_l \leftrightarrow K_l$, $l=1$, (ағылшын – қазақ жұп мәтін корпусы, N – паралель корпустағы жұп сойлемдер саны); A_l – ағылшынша l-сөйлемі, K_l – ағылшыннан аударылған қазақша l-сөйлемі;

2. Ағылшын мен қазақ тіліндегі сөйлемдердің сөздер мен сөз-тіркестерін (3-грамм) туралау мен *G* туралау коэффициентін есептеу;

3. Егер *G* > 0.5, онда 5-қадамға өту, басқа жағдайда келесі қадам орындалады.

4. Қате аударылған сөздер синонимін табу және ықтималдығы жоғары синонимді таңдау.

5.Соңы.

Жоғарыда сипатталған алгоритм негізінде қате аударылған қазақ тілдегі сөзді табу мысалы қарастырылды:

1. Ағылшын тіліндегі *S* сөйлемі Google Translator арқылы қазақ тіліне аударылды: *S= I have a nice baby*.

2. Аударма нәтижесінде *T* деп аталатын *S* сөйлемнің қазақша аудармасын аламыз. *T= Менің сүйкімді балам бар* сөйлемі содан кейін ағылшын тіліне қайта аударылды, ол *S'* деп аталды.

3. *T* қазақ тіліндегі сөйлем ағылшын тіліне аударғандағы сөйлем аудармасы *S'= I have a cute baby* болды. *S'* сосын (3.3) формуласында қолданылады.

Сонда *S* (σ) ағылшын сөйлемі *I have a nice baby* қазақшаға *Менің сүйкімді балам бар* деп аударылды. Туралау тұрақтылығын анықтау үшін аударылған сөйлемдегі саб-сегменттер екі түрлі бағытта қарастырылады (тура және кері): ағылшыннан қазақшаға (және керісінше). Бұл (3.1) формуладағы саб-сегменттерден тұратын *M* жиыны болып есептелінеді:

I ⟷ *мен*
have ⟶ *бар*
a ⟶ *а*
nice ⟶ *жақсы*
baby ⟶ *сәби*
I have ⟷ *мен(де) бар*
have a ⟶ *бар*
a nice ⟶ *жақсы*
nice baby ⟶ *жақсы бала*
I have a ⟶ *мен(де) бар*
have a nice ⟶ *жақсы өт(іңіз)*
a nice baby ⟶ *сүйкімді сәби .*

Сонда *M* жиының n=3 жағдайын қарастырылды, яғни сөйлемдегі алынатын сөз тіркестерінің ұзындығының саны 3-ке тең.

Бір бағытта және екі бағытта жебелер бар екенін байқауға болады. Олардың айырмашылығы – бір бағыттағы жебелер тек бір бағытта, ал екі бағыттағы жебелер тікелей және кері аудармада да бірдей аударылып, нәтижесінде түпнұсқа сөз сияқты аудармада аударылғаны көрсетілген.

Енді алынған саб-сегменттерді (3.1) формуласына қойылып есептелінді. Түсініктірек болу үшін қолданатын екі (*S,T*) сөйлемнің сөздері нөмірленіп, сөйлемдегі сөздердің леммалары Апертиум платформасы арқылы тауып алынды:

$S= I_1\ have_2\ a_3\ nice_4\ baby_5$
$T= Мен_1\ сүйкімді_2\ бала_3\ бар_4$

Келесі ағылшын сөйлеміндегі бірінші сөз *I* мен қазақ тіліндегі сөйлемнің бірінші сөзі *мен* үшін $A_{1,1}$ туралау салмақ мәні табылды. Ол үшін *I* мен *мен* сөзінің *M* жиынында неше рет кездескені есептелінді. Сонда байқалғандай *M* жиынында *мен* үш рет кездеседі:

I ⟷ *мен*
I have ⟷ *мен(де) бар*
I have a ⟶ *мен(де) бар*

Сосын (3.1) формулада көрсетілгендей әр саб-сегмент үшін екі бағыттағы сөздердің ұзындық санының көбейтіндісін 1-ге бөліп есептелінді:

$A_{1,1}$=1/(1×1)+1/(2×2)+1/(3×2)=1+0.25+0.16=1.41

Тағы сол сияқты қалған саб-сегменттер үшін есептеулер жүргізілді:

$A_{1,4}$=1/(2×2)+1/(3×2)=0.25+0.16=0.41

$A_{2,1}$=1/(2×2)+1/(3×2)=0.25+0.16=0.41

$A_{2,4}$=1/(1×1)+1/(2×1)+1/(2×2)+1/(3×2)=1+0.5+0.25+0.16=1.91

$A_{3,1}$=1/(3×2)=0.16

$A_{3,2}$=1/(3×2)=0.16

$A_{3,4}$=1/(2×1)+1/(3×2)=0.5+0.16=0.66

$A_{4,2}$=1/(3×2)=0.16

$A_{4,3}$=1/(2×2)=0.25

$A_{5,1}$=1/(2×2)+1/(2×2)=0.5

$A_{5,2}$=1/3×2=0.16

$A_{5,3}$=1/(2×2)=0.25

Мәні жазылмаған $A_{1,2}$, $A_{1,3}$ сияқты тағы басқа саб-сегменттер мәні 0-ге тең, өйткені ағылшын тілінде кездескен сөз қазақ тілде немесе керісінше бағытта кездеспейді. Алынған нәтижелерді келесі 3.1- кесте түрінде көруге болады.

Кесте 3.1 – *"I have a nice baby"* ағылшын сөйлемінің *"Менің сүйкімді балам бар"* деген қазақша аудармасының туралау салмағының нәтижесі

I	1.41			0,41
have	0.41			1.91
a	0.16	0.16		0,66
nice		0.16	0,25	
baby		0.16	0.25	
	менің	*сүйкімді*	*балам*	*бар*

Осы сөйлемдегі қате аударылған сөзді немесе сөздерді анықтау үшін туралау салмақтары есептеліп, Эспла-Гомис және т.б. (2015) қате сөзді автоматты түрде табу әдісінде қолданды. Толығырақ ақпарат [3.1-3.3] формуласы мен [56] еңбегінде айтылған.

Қате сөзді автоматты түрде табу әдісінің нұсқауы келесідей: егер *G (k, S', S, T)* ≤1/2 болса, онда сөзді өзгерту керек деп белгіленеді. Кері жағдайда сақтау керек деп, яғни өзгеріссіз қалсын деп белгіленеді. Бөлім басында айтылғандай *S'* сөйлемі кері аударма арқылы алынды. *G* функциясын табу үшін кері аударма қолданылады. Алдында көрсетілгендей *S'=Менің сүйкімді балам бар* деп аударылған. Яғни егер *G(k,S',S,T)*≥1/2 болса онда ол дұрыс, қате емес сөз дегенді білдіреді.

Енді (3.3) формуланы қолданып қазақ тілдегі сөздер үшін *G* функциясының мәндері есептелінді.

“Менің” деген сөздің *G (1, S’, S, T)* функциясының мәні =(1,41×1+0,41×1+0,16×1)/(1,41+0,41+0,16)=1,98/1,98=1 тең. Алынған мән 1≥1/2 яғни, «*менің*» деген сөз дұрыс.

“Сүйкімді” деген сөздің *G (2, S’, S, T)* функциясының мәні =(0,16×1+0,16×0+0,16×1)/(0,16+0,16+0,16)= 0,32/0,48=0,6 тең. Алынған мән 0,6≥1/2 яғни*“сүйкімді”* деген сөз дұрыс.

“Балам” деген сөздің *G (3, S’, S, T)* функциясының мәні = (0,25×0+0,25×1)/(0,25+0,25)=0,25/0,5=0,5 тең. Алынған мән 0,5 ≥1/2 яғни *“балам”* деген қате сөз.

“Бар” деген сөздің *G (4, S’, S, T)* = функциясының мәні (0,41×1+1,91×1+0,66×1)/ (0,41+1,91+0,66)= 2,98/2,98=1 тең. Алынған мән 1≥1/2 яғни *“бар”* деген сөз дұрыс.

Осы нәтижелерге сүйене отырып, *“балам”*сөзі қате аударылған сөз болып шыққанын байқалды. Енді осы қате аударылған сөзден автоматты түрде каталог қалыптастыру үшін екінші тапсырма келесі тарауда қарастырылған.

Үшінші тараудың қорытындысы

Бұл үшінші тарауда қарастырылып отырған ғылыми жұмыстың бірінші тапсырмасы – қазақ тіліндегі сөйлемнің қате аударылған сөздерін анықтау қарастырылды. Қорыта келгенде:

- ағылшыннан аударылған қазақ тіліндегі сөйлемнің қате аударылған сөздерін анықтау үшін біріктірілген әдіс үлгісі мен алгоритімі қолданылып, мысалдар келтірілді.
- қате аударылған сөзді автоматты түрде табу әдісіне Апертиум платформасы көмегімен сөздер леммасы қолданылып, әдіс жақсартылды.
- қате сөзді автоматты түрде табу әдісі үшін кері аударманы қолдану жұмыстың ғылыми жаңалығын арттырды.

4 ҚАТЕ АУДАРЫЛҒАН ҚАЗАҚ СӨЗДЕРДІҢ СИНОНИМДЕР КАТАЛОГЫН АВТОМАТТЫ ТҮРДЕ ҚАЛЫПТАСТЫРУ

4.1 Қате аударылған қазақ сөздердің синонимдер каталогын автоматты түрде қалыптастыру құралдары

Анықтаған қате сөздер каталогын қалыптастыру мақсатында PyDictionary кітапханасы қолданылды. PyDictionary – мағыналар, аудармалар, сөздердің синонимдерін және антонимдерін алуға арналған Python 2/3 сөздік модулі[2]. Ол *synonym.com*[3] сайтынан синонимдер және антонимдер алу үшін, Google-ді аудармалар үшін және сөз мағыналарын алу үшін WordNet пайдаланылады. WordNet®[4]- ағылшын тіліндегі үлкен лексикалық дерекқор [57].

Сонымен қатар, каталог құру кезінде онлайн сөздік қолданылды. Ол *thesaurus.com* сайтынан алынған *dictionary.com*[5] ұсынған әлемдегі ең үлкен және ең сенімді тегін онлайн тезаурус [58].

Dictionary.com анықтамалар, сөздердің шығу тегі және тағы басқа құралдар үшін әлемдегі жетекші онлайн көзі болып табылады. *Dictionary.com* миллиондаған адамдар үшін ағылшын тілінің құпияларын ашады. *Dictionary.com* бұл сөздермен жұмыс істейтін әлемде байланыстарды, қарым-қатынасты, оқуды, шығармашылықты және көптеген бағыттағы нәрселерді шабыттандыруға тырысатын құрал. *Dictionary.com* — әлемдегі жетекші цифрлық сөздік. Бұл сөздік миллиондаған ағылшынша анықтамаларды, емлелерді, дыбыстық айтылымдарды, үлгі сөйлемдерді және сөздің шығу тегін ұсынады. Бұл сайттың негізгі меншікті көзі - Random House Unabridged Dictionary, оны тәжірибелі лексикографтар тобы үнемі жаңартып отырады және әртүрлі тіл қажеттіліктерін қолдау үшін сенімді, танылған дереккөздермен, соның ішінде American Heritage және Harper Collins арқылы толықтырылады. Сонымен қатар аударма қызметін, кроссворд шешімін және озық сөз әуесқойлары мен ағылшын тілін үйренушілерге пайдалы болатын көптеген редакциялық мазмұнды ұсынады.

20 жылдан астам уақыт бойы *thesaurus.com* миллиондаған адамдарға ағылшын тілін меңгеруді жақсартуға және 3 миллионнан астам синонимдер мен антонимдері бар нақты сөзді табуға көмектесіп келеді.

Синоним сөздерді *thesaurus.com* сайтынан қолдану үшін Beautiful Soup кітапханасы қолданылды. Beautiful_Soup — бұл HTML мен XML файлдарынан мәліметтерді алу үшін қолданылатын Python кітапханасы немесе Python бағдарламалау тілінде жазылған HTML/XML файлдарын синтаксистік талдауға арналған талдаушы(парсер) [59].

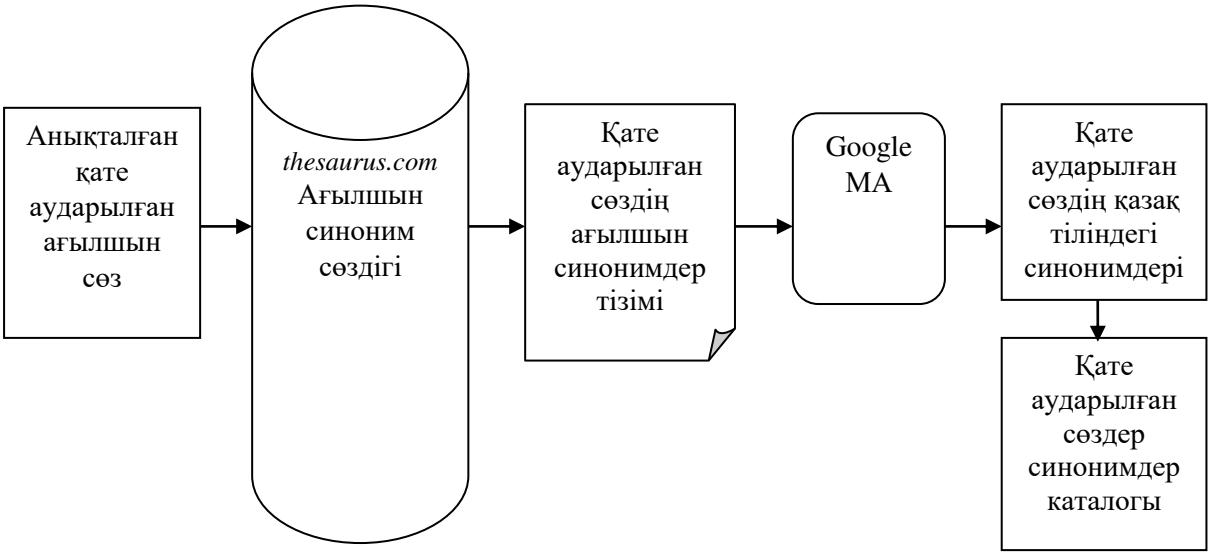
Зерттеу жұмысында бастапқы бірінші тапсырмадағы қате сөздер табылған соң, сол анықталған қате сөздердің ағылшын тіліндегі нұсқалары *thesaurus.com* ресурсынан синонимдерін алынады. Табылған барлық ағылшын

[2] https://pypi.org/project/PyDictionary/
[3] https://www.synonym.com/
[4] https://wordnet.princeton.edu/
[5] https://www.dictionary.com/

синоним сөздерді Google аудармашысы арқылы қазақшаға аударылып файлға сақталады. Сонда аударылған қазақша синоним сөздер каталогы құрылады. Каталогты автоматты түрде құру келесі бөлімде сипатталған.

4.2 Қате аударылған қазақ сөздердің синонимдер каталогын автоматты түрде қалыптастыру сұлбасы мен алгоритмі

PE-LC алгоритмінің екінші тапсырмасы қате аударылған сөздердің синонимдерінің каталогын автоматты түрде құруға бағытталған [60]. Егер кіріс (input) *S* сөйлемде қате сөздер табылса, ол сөздер каталогтан ізделінеді. Егер қате сөз табылса, үрдіс келесі кезеңге өтеді. Егер жоқ болса, каталогқа қате аударылған сөздердің синонимдер тізімін құратын алгоритімі іске асырылып қосылады. Мұндай тізімді құру үшін *thesaurus.com* арқылы ағылшын тіліндегі қате аударылған сөздің синонимдері қолданылады. Бұл процесс толығымен автоматты түрде жүзеге асырылады. Табылған синонимдер тізімделіп, артынан қазақ тіліне МА арқылы аударылады. Осылайша, қазақ тіліндегі каталог автоматты түрде қалыптастырылады. Зерттеудің басында қате аударылған сөздерді табуға болатын кіріспе сөйлем енгізіледі. Бұл үрдіс 4.1- суретте көрсетілген.



Сурет 4.1 – Қате аударылған сөздер синонимдерінің каталогын автоматты түрде қалыптастыру жалпы сұлбасы

Қате аударылған сөздің синонимдерін табу алгоритмі келесі қадамдардан тұрады:

1. Кіріс деректер: $w_j^{каз}$ - қазақ қате аударылған сөздер, $w_i^{анг}$ - қазақ тіліне сәйкес ағылшын сөзі.

2. Синоним каталогында $w_j^{каз}$ іздеу.

3. Егер $w_j^{каз}$ каталогта болса, $w_j^{каз}$ синонимдер тізімін шығару, 8-қадамға өту, кері жағдайда келесі 4-қадам орындалады.

4. $w_j^{каз}$ - ауыстыру үшін ықтималдығы жоғары синонимді таңдау (сол модульге өту).

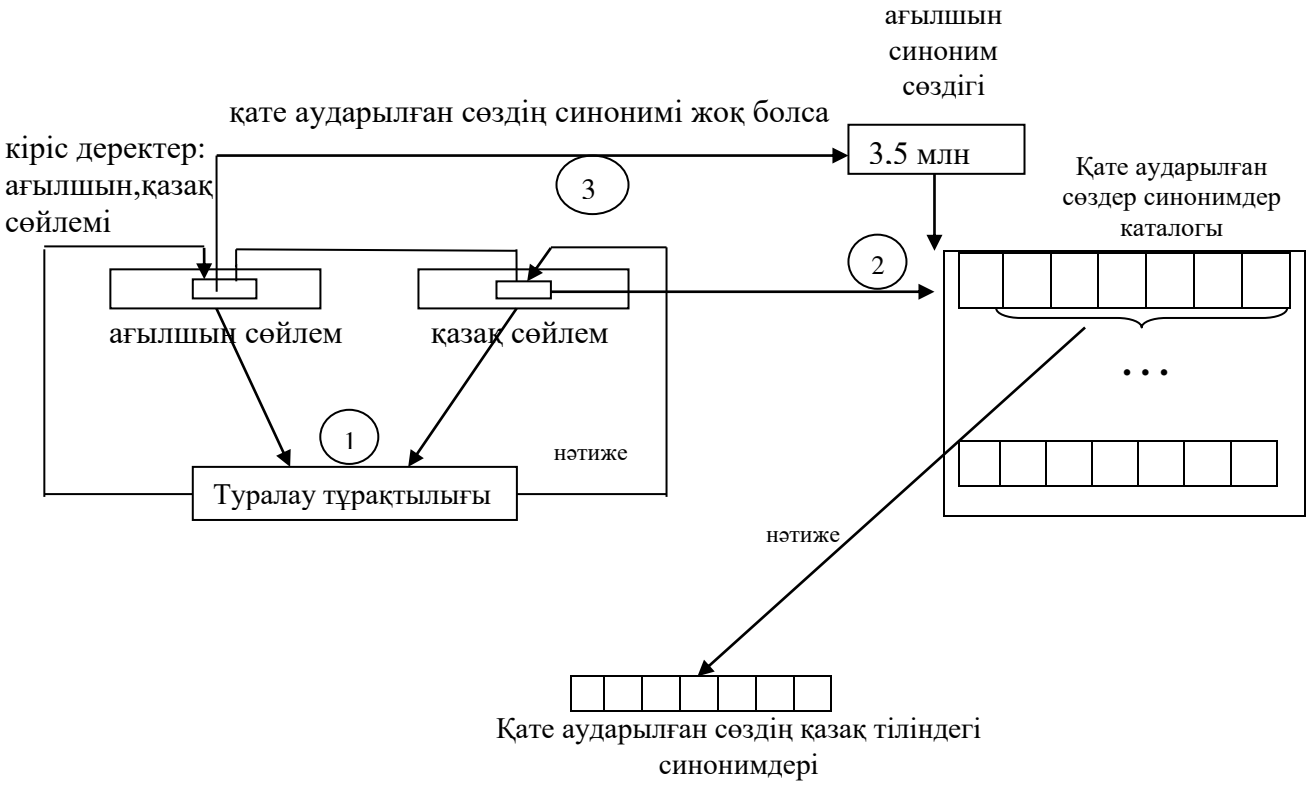
5. Ағылшын синоним сөзінен $w_i^{анг}$ іздеу. (3,5 млн сөз) [*https://www.thesaurus.com/*]

6. $w_i^{анг}$ синонимдерін *Google Traslate*-пен қазақ тіліне аудару;

7. Жаңа жолда $w_j^{каз}$ - қазақ синонимдерімен каталогты толықтыру (кеңейтілуі). Синонимдер тізімін шығару. 8-қадамға өту.

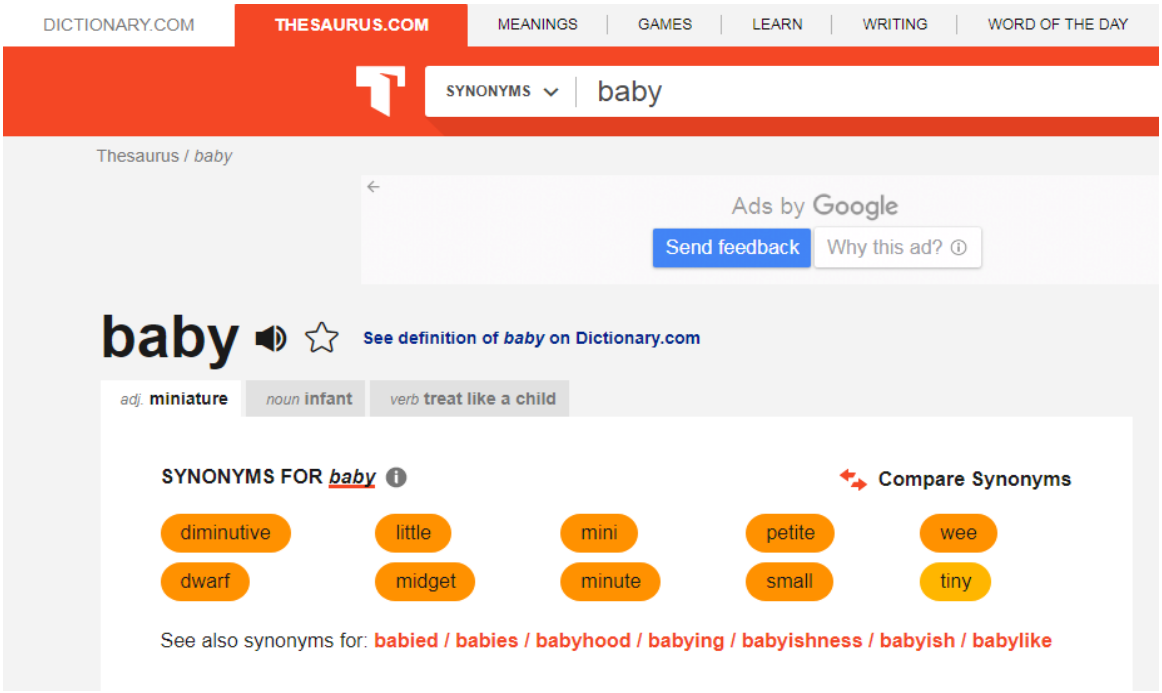
8. Соңы.

Осы қадамдар реті орындалып қате аударылған синонимдер тізімі құрылады. Содан соң автоматты түрде каталог құрылады. Қате аударылған синонимдер каталогын құру алгоритмі келесі 4.2- суретте көрсетілген.



Сурет 4.2 – Каталогтағы қате аударылған қазақша сөздердің синонимдерін табу мен каталогта болмаған жағдайдағы кеңейтілу сұлбасы.

Қолданылатын *thesaurus.com* сайтындағы онлайн сөздікті келесі 4.3-суретте көруге болады.



Сурет 4.3 – *thesaurus.com* сайтындағы *"baby"*қате сөздің синонимдері

4.3-суретте көрсетілген сөздер қате сөздердің барлық ықтимал баламалы сөздері болып табылады, мысалы бұл каталогта қате сөздердің ағылшын тіліндегі нұсқалары көрсетілген. *thesaurus.com* сайтын Beautiful Soup кітапханасын қолдану арқылы бағдарламалық қамтамасыздандыруда қолданылды. Мысалы қате аударылған *baby* сөзінің келесідей синонимдері бар:

Baby: *diminutive, dwarf, little, midget, mini, minute, petite, small, wee, tiny* және т.б.

Табылған ағылшын тіліндегі қате аударылған синонимдер сөздер Google translator көмегімен қазақ тіліне аударылып, каталогқа жазылады. Қазақ тіліндегі әрбір жаңа қате сөзі және оның баламалары каталогтың жаңа жолына жазылып сақталады. Каталог жазбасының қысқа үзінді түрі келесідей көрінеді:

1. *Аз, жеткіліксіз, шамалы, шектеулі*
2. *...*

40. Сыпайы, жақсы, әділет, мейірімді, сүйкімді, қолайлы, тамаша, ерекше

41. Сәби, ергежейлі, кішкентай, кіші, шағын ... т.б.

Каталогта қолданылған сөздік қорын келесі 4.1- кестеден көруге болады.

Кесте 4.1 – Каталог туралы ақпарат

Ағылшын синоним сөздік қоры	Қазақ тіліндегі барлық синонимдер саны
3,5 млн.астам	7000 астам

Негізі бұл көлемі 1000 жолды құрайтын қазақ тілдегі синонимдердің каталогы. Каталог бірінші сөз арқылы индекстеледі. Каталогтың әр жолында қате сөздердің синонимдері көрсетілген, яғни 1000 жолды қате аударылған сөздері бар каталогтың әр жолында он бес-он жеті синонимдерге дейін кездеседі. Әрбір табылған жаңа қате сөз бен синонимдері каталогқа жаңа жолға сақталып жазылады. Егер қажетті қате сөз каталогтан іздегенде табылса, онда оған қатысты кестелерді қолдану үшін керек қарастырылып отырған қате сөздің жол нөмірін ескеріп сол нөмірдегі кесте ашылады. Мысалы жоғарыда көрсетілген каталог тізіміндегі *сыпайы, жақсы, әділет, мейірімді, сүйкімді, қолайлы, тамаша, ерекше* деген сөздер қырықыншы жолда орналасқан яғни осы сөздермен байланысты қажетті кесте нөмірі қырықыншы болып белгіленеді.

Бұл кестелерді құру үлгісі мен алгоритімі келесі 4-ші тарауда үшінші тапсырмада толық сипатталған.

Төртінші тараудың қорытындысы

Бұл төртінші тарауда үшінші тарауда анықталан қате сөздердің синонимдер каталогын қалыптастыру технологиясы сипатталған. Қорыта келгенде бұл тарауда:

- қате аударылған қазақ синонимдер каталогын автоматты түрде қалыптасыру кезінде қолданылған құралдар жайлы жазылды. Python тілінде жазылып қолданылған PyDictionary, Beautiful_Soup кітапханалары туралы сипатталды.
- *thesaurus.com* онлайн синонимдер сөздігінен қажетті сөздер автоматты түрде алынып, Google translate-пен қазақшаға аударылып каталогқа жаңа жолмен сақталды.
- қате аударылған сөздерден алғаш рет синонимдер каталогын автоматты түрде қалыптастыру үлгісі мен алгоритмі сипатталып, мысалдармен түсіндірілді.

5 ҚАТЕ АУДАРЫЛҒАН СӨЗДІ МАҒЫНАСЫ ЖАҒЫНАН ЖАҚЫН СИНОНИММЕН АУЫСТЫРУ

5.1 Қате сөздерді түзету мақсатында қолданылатын семантикалық текше үлгісі мен алгоритмі

Бұл үшінші тапсырмасын шешу үшін Төкеев және т.б. (2017) бастаған ғылыми топ жұмысының максималды энтропияға негізделген «семантикалық текше» әдісі қолданылды. Бұл әдістің ерекшелігі Тайерс т.б. (2015) ағымдағы сөздің контекстін анықтау үшін ағымдағы сөздің сол жағындағы екі сөздің және оң жағындағы екі сөздің дәйекті тіркесімін пайдаланса, Төкеев және т.б. (2017) қарастырылатын сөйлемнің барлық сөз тіркестерін, тікелей байланыс емес тіркестерін (non-consecutive collocation) қарастырды.

Негізі максималды энтропияда классификация формуласы қолданылады:

$$\hat{c} = \arg\max_{c \in C} P(c \mid x) \tag{5.1}$$

Максималды энтропия үлгісінде [61,62], белгілі бір *c* класының ықтималдығы бағаланады (Jurafsky, D. and Martin, J.H. (2007)) [63]:

$$P(c \mid x) = \frac{1}{Z} \exp \sum_j \lambda_j f_j \tag{5.2}$$

Мұнда Z – нормалаушы фактор.

Максималды энтропияда берілген *x*-тің *c* класындағы ықтималдығын есептейтін теңдеу(5.3-теңдеу):

$$P_i^e(c \mid x) = \frac{\exp(\sum_{j=1}^{N^e} \lambda_{ij}^e f_{ij}^e(c, x))}{\sum_{c' \in S^e} \exp(\sum_{j=1}^{N} \lambda_{ij}^e f_{ij}^e(c', x))} \tag{5.3}$$

Мұндағы,

$$f_{ij}^e(c, x) = \begin{cases} 1, \text{ егер } x = z_j^e \,\&\, c = s_i^e \text{ (} x \text{ } c\text{-ның тіркесі)} \\ 0, \text{ кері жағдайда} \end{cases} \tag{5.4}$$

ω^e – *көпмағыналы сөз*,

c – *синонимдер класы*,

z_j^e – *с класының j-шы қасиет сөзі* (s_i^e)

x– *зерттелінетін сөз*,

N^e – ω^e *үшін қасиеттер (features) саны,*

λ_{ij}^e – f_{ij}^e *қасиеттер (feature) салмағы ,*

S^e – ω^e *үшін синонимдер жиынтығы*

Кесте 5.1 – ω^e сөз жиілігі кестесі

	ω^l						…	
S^e	ω^e	$z^e{}_1$	$z^e{}_2$	$z^e{}_3$	$z^e{}_4$	$z^e{}_5$	…	$z^e_{N^e}$
s^e	s_1 f_{1j}	0	1	0	1	0	…	0
	s_1 g_{1j}	0	5	0	6	0		0
s^e	s_2 f_{2j}	1	0	0	0	1	…	0
	s_2 g_{2j}	1	0	0	0	6		0
							…	

Мұнда g_{ij} – жиілік. ω^e көпмағыналы сөздің $s_1{}^e$ синонимі үшін g жиілігін қолданылып және f қасиетін пайдаланылып, λ^e салмағы есептелінді. Ол үшін 5.1-кестедегі мәліметтері қолданылды:

$$\lambda_{12} = \frac{g_{12} f_{12}}{\sum_{j=1}^{N^e} g_{1j} f_{1j}} = 5/11 = 0,45$$

$$\lambda_{14} = \frac{s_1 f_4}{\sum_{j=1}^{N^e} g_{1j} f_{1j}} = 6/11 = 0,54$$

мұндағы, g_{ij} - s_i синоним үшін z_j қасиет сөз жиілігі (s_i үшін сөз тіркес жиілігі); ω^e көпмағыналы сөз бен s_2^e синонимі үшін λ^e салмақтарының есептелуі келесідей болады:

$$\lambda_{21} = \frac{g_{21} f_{21}}{\sum_{j=1}^{N^e} g_{2j} f_{2j}} = 1/7 = 0,14$$

$$\lambda_{25} = \frac{g_{25} f_{25}}{\sum_{j=1}^{N^e} g_{2j} f_{2j}} = 6/7 = 0,85$$

Толық нәтижені 5.2 кестеден көруге болады

Кесте 5.2 – ω^e сөзі үшін есептелінген семантикалық текше кестесі

ω^e	$z\,1$	$z\,2$	$z\,3$	$z\,4$	$z\,5$	…	z_{N^e}
1	2	3	4	5	6	7	8
s_1 f_{1j}	0	1	0	1	0	…	0
s_1 λ_{1j}	0	0,45	0	0,54	0		0

5.2-кестенің жалғасы

1	2	3	4	5	6	7	8
$s_2 \quad f_{2j}$	1	0	0	0	1	…	0
$s_2 \quad \lambda_{2j}$	0,14	0	0	0	0,85		0
			…				

(5.3) -формуласы бойынша *синонимдер класының* ықтималдығын есептеу келесідей болады:

$$P(s_1 \mid x) = \frac{e^{0,45} * e^{0,54}}{e^{0,45} * e^{0,54} + e^{0,14} * e^{0,85}} = \frac{0,243}{0,243 + 0,119} = \frac{0,243}{0,362} \approx 0.67$$

$$P(s_2 \mid x) = \frac{e^{0,14} * e^{0,85}}{e^{0,45} * e^{0,54} + e^{0,14} * e^{0,85}} = \frac{0,119}{0,243 + 0,119} = \frac{0,119}{0,362} \approx 0,33$$

Сосын (5.1-формула) классификация формуласы қолданылады. Яғни, $P(s_i \mid x)$ максимум болатын s_i синонимдік класс таңдалады. Мысалы, таңдалған синоним s_1 болып табылады, себебі $P(s_i \mid x)$ мәні ең үлкен (максимум) көрсеткішті көрсетті.

Алгоритм 3. Қате аударылған қазақ сөзі үшін ықтималдығы жоғары синонимді таңдау алгоритмі

Қате аударылған сөз анықталғаннан кейін мәтінге қатысты ықтималдылығы жоғары сөзді таңдау арқылы жүзеге асырылған алгоритімнің сипаттамасы келесідей:

1. Кіріс деректері: қате аударылған қазақ сөзінің синонимдер тізімі, ағылшын – қазақ жұп корпусы.

2. Қазақ тіліндегі синонимдер семантикалық текшенің қалыптаструы. (Қазақ тіліндегі корпусты оқыту). Қате аударылған сөздер үшін синоним ықтималдықтар семантикалық текшесін құру.

3. $w_j^{каз}$ қате аударылған қазақ сөздің ықтималдығы ең жоғары (*max*) синонимін таңдау .

4. K_l сөйлеміндегі қате аударылған қазақ сөзінің ықтималдығы max синоним мәнімен ауыстыру.

5. Пост-редакияланған сөйлем шығару.

6. Соңы

5.2 Қате аударылған қазақ сөзі үшін ықтималдығы жоғары синонимді таңдау мысалы

Максимум энтропияға негізделген семантикалық текше әдісінің қолданылуының практикалық есептеулерін келесі кестелер мен сипаттамалардан көруге болады [64-66]. 5.3-кестеде ағылшын және қазақ тілдеріндегі синонимдері бар қате аударылған сөз көрсетілген.

Кесте 5.3 – ағылшын және қазақ тілдеріндегі синонимдері бар қате аударылған «*baby*» сөзінің мысалы

ω^e , қате аударыл-ған сөз	Синоним 1	Синоним 2	Синоним3	Синоним4	Синоним5
baby	сәби	ергежейлі	кішкентай	кіші	шағын

Бұл тізім параллель ағылшын – қазақ сөздеріне негізделген аудармалардан құралады. Мәтінмәннен(context) алынған сөздер, қате аударылған сөздер (синонимдер) және олардың жиіліктері сәйкес кестеге жазылады. Мүмкін аудармалар 5.4-кесте арқылы анықталады.

Мәтінмән (контекст) – басында енгізілген сөйлем. Корпуста кездесетін және мәтінмәнде кездесетін сөздер мен қате аударылған синоним сөздердің жиілігін қолданылады.

Кесте 5.4 – Контексттегі қазақша «*baby*» аудармасының нұсқалары

z, корпустан алынған сөздер	ω^e және S^e, қате аударылған сөздер мен синонимдер	g, олардың жиіліктері
мен	*сәби*	201
көйлек	*ергежейлі*	20
мінез	*кішкентай*	189
…	…	…
бойжеткен	*кіші*	31

Екі тілдегі қате аударылған сөздердің леммаларының тізімі параллель ағылшын – қазақ корпустарында кездесетін сөйлемдерден алынды және жадыда сақталды. Қате сөздердің каталогы жаңартылғаннан кейін қолданылу жиілігі бүкіл паралель корпус бойынша есептелді. Қате аударылған сөздердің корпуста кездесетін жиілігін есептеу үшін ағылшын және қазақ тіліндегі сөйлемдердің қиылысы қарастырылады. Тек бір жағы (бағыты, ағылшын не қазақ) қолданылса, қажетті сөздер табылмауы мүмкін. Қазақ және ағылшын корпусындағы қате аударылған сөздер қазақша да, ағылшынша да сөйлемдерде болмауы мүмкін. Егер бір ғана нұсқа алынса, онда сөйлемнің мағыналары әртүрлі болуы мүмкін. Содан кейін семантикалық текше әдісі арқылы сөздердің

қазақ тіліндегі нұсқаларынан қате синоним сөздерден құралған кестелер автоматты түрде құрылады.

Бұрын түсіндірілгендей, мақсатты сөйлемнен қате аударылған сөздер табылады. Содан кейін каталогтан осы сөздердің баламалары (синонимдері) ізделінеді. Егер каталогта осы қате аударылған сөз болса, сол жолға байланысты кесте ашылады және тек кіріс сөйлемдегі сөздердің кездесу ықтималдығы есептелінеді. Ал егер каталогта қажетті қате сөз табылмаса, синонимдердің жаңа тізіміне(каталогқа) қате аударылған жаңа сөздерді қосу алгоритміне көшеді. Осылайша, каталог ұлғая береді. Келесі қарастырылатын семантикалық текшені құру алгоритмі болып табылады.

Семантикалық текшені құру үшін мақсатты сөйлемдегі баламаның дұрыс нұсқасын таңдау үшін ықтималдық үлгісі (probabilistic model) мен корпус пайдаланылды. Параллель (ағылшынша – қазақша) корпустың көлемі 250 000 сөйлемді құрайды. Осы семантикалық текше әдісін қолдануды жақсарту үшін келесідегідей қадамдар орындалды:

- Біріншіден, корпустағы барлық сөйлемдердің леммаларын табу үшін Apertium пайдаланылды.
- Екіншіден, көпмағыналы сөздерге қате аударылған синонимдер каталогынан әрбір қате аударылған қазақ сөзіне синонимдер қолданылды, оның барлық синонимдерімен және контекстпен (корпустан) осы синонимдердің кездесу жиілігімен кесте құрастырылды.
- Әрбір қате аударылған синоним сөз үшін семантикалық текше құрылады. Семантикалық текшені құру алгоритмінің негізгі қадамдары төмендегідей.
    - ❖ Біріншіден, ағылшын корпусынан бұрын табылған қате аударылған сөздерді олардың синонимдерімен бірге ізделінді. Ағылшын синонимдері қолданылып қазақша аудармалары жинақталды. Бұл синонимдер 250 000 сөйлемнен тұратын паралель корпустан ізделінді. Апертиум арқылы сөздердің леммалары табылды. Синонимдердің контексте кездесу ықтималдығы есептелінді. Анықталан қате синоним сөздер кестелерінде контекстерінің (корпустағы) ықтималдықтары қажетті сөздер үшін есептілінді. Ол үшін каталогта табылған синоним сөздердің қатарларының саны (нөмірі) осы синонимдермен байланысты кестенің санына теңестірілді (5.5-кесте). Кестеде корпуста кездесетін барлық сөздер мен қате аударылған сөздердің кездесу жиілігі көрсетілген.

Кесте 5.5 – «*baby*» деген қате аударылған сөздің синонимдері мен корпуста кездесетін сөздермен жиілігі кестесінің бөлігі

z, корпустағы сөздер	ω^e мен S^e, қате аударылған сөздер мен синонимдер
тіл	сәби : 42

5.5- кестенің жалғасы

1	2
үзінді	сәби : 1
мен	сәби : 201
үст	сәби: 38
ойын	сәби : 1
кеш	сәби : 84
ана	сәби : 113
мен	ергежейлі : 12
бой	кіші : 29
із	кіші : 19
біз	кіші : 45
дым	кіші : 63
ала	кіші: 65
оқырман	кіші : 5
бір	бала : 16
түсін	бала : 3
бері	бала : 5
не	бала : 37
бер	бала : 22
үй	бала : 26
ай	бала : 61
...	...

Егер бұл қате аударма каталогта болса, каталогтан қажетті баламаларын алып, жүйе қате аударылған сөздердің дұрыс нұсқасын таңдап, түзетілген сөйлемді шығарады. Егер ол каталогта жоқ болса, каталог толықтырылып, жүйе синонимдер кестесін (каталогты) және сол синонимдердің корпуста кездесу жиілігінің кестесін құрады. Семантикалық текшенің параметрлерін қарастырып отырған бастапқы *I have a nice baby* сөйлеміндегі *“baby”* сөзінің параметрлерін ғана көрсетілген. Яғни, мысал ретінде берілген қате аударылған сөзді көрсете отырып семантикалық текшенің жаңалану реті түсіндіріліп сипатталған.

5.4-жиілік кестесінен *T* мақсатты сөйлемде кездесетін сөздердің (*мен, сүйкімді, бар*) синонимдерін ғана аламыз. Алдында айтылып кеткендей, тек берілген сөйлемдегі сөздер мен қате аударылған синонимдердің корпустағы жиіліктері қолданылады. Толық ақпаратты 5.5-кестеден көруге болады.

Кесте 5.5 – Мақсатты(target) сөйлемде кездесетін сөздер, қате аударылған *“baby”* сөзі мен оның синонимдер жиілік кестесі

ω^e және S^e / z, контекст-тегі сөз	сәби	ергежейлі	кішкентай	кіші	бала
мен	201	12	104	153	145
сүйкімді	258	15	40	136	23
бар	233	136	199	144	89

Менің сүйкімді балам бар сөйлеміндегі әр сөз үшін қате аударылған сөздердің ықтималдығын (5.1) және (5.2) формулалары арқылы есептелінеді: 250000 сөйлемдер қолданылып, екі тілді корпустың қазақша сөйлемдері қолданылады. Яғни, $P(s_i \mid x)$ максимум мәнге ие болатын s_i синонимдік класы таңдалады. Сол себепті анықталған қате сөз бен оның синонимдарының ықтималдығы есептелінді.

Сәби синоним сөзінің ықтималдығы: $P(s_1 \mid x)$ = (201+258+233)/250000=0.0027

Ергежейлі синоним сөзінің ықтималдығы: $P(s_2 \mid x)$ = (12+15+136)/250000= =0.00065

Кішкентай синоним сөзінің ықтималдығы: $P(s_3 \mid x)$ = (104+40+199)/250000= =0.001372

Кіші синоним сөзінің ықтималдығы: $P(s_4 \mid x)$ = (153+136+144)/250000= =0.001732

Бала синоним сөзінің ықтималдығы: $P(s_5 \mid x)$ = (145+23+89)/250000= =0.001028

(5.4) формулада көрсетілгендей, табылған ықтималдықтардың ішінен ең үлкен мәні таңдалынып, қате аударылған «сәби» сөзінің максимум мәнге ие $P(s_1 \mid x)$ синонимі таңдалынды.

Тек *T* мақсатты сөйлемдердегі сөздердің қолданылу себебі, кестелерде корпустың көптеген сөздері болуы мүмкін. Яғни, сөздердің жалпы жиілігі маңызды емес, берілген сөйлемдегі сөздер мен синонимнің жиілігі ғана маңызды. 5.5-кестеде тек енгізілген бастапқы сөйлемге қатысты баламалары бар қате сөздер көрсетілген.

Бастапқы сөйлемдегі де, мақсатты сөйлемдегі де сөздер әртүрлі формада болуы мүмкін, (бұл қазақ тілінде өте маңызды, өйткені оның морфологиясы өте бай), сондықтан балама іздеу үшін олардың леммаларын қолдану қажет. Содан кейін морфологиялық анализатор мен генератор көмегімен жаңа *t** сөзі бастапқы қате аударылған *t* сөзі сияқты формаға келтіріледі. Бұл жұмыста қазақ және ағылшын тілдерін морфологиялық өңдеу үшін Apertium платформасы қолданылды. Толығырақ ақпаратты 5.6-кестеде көруге болады.

Апертиум платформасы көмегімен сөйлемдегі сөздердің леммалары табылады. Түзетілген толық сөйлемді түзету үшін де қолданылады. Сөзді дұрыс талдау үшін леммаға айналдырылғандықтан, сөйлемдердегі сөздердің

леммаларын Apertium морфологиялық генераторы сәйкес жалғауларын жалғап, түзетілген сөйлеммен аяқталады. «Дұрыс емес» мақсатты сөйлемге морфологиялық талдау жасалып, семантикалық текше әдісі есептеліп қолданылғаннан кейін мағынасы жағынан дұрыс ауыстырылған сөз(дер)ді түзету шығарылады. Апертиум платформасының *Менің сүйкімді сәбиім бар* деген сөйлемге жасалған морфология талдауын 5.6-кесте көруге болады.

Кесте 5.6 – Түзетілген сөйлемнің морфологиялық талдауы

мен <prn><pers><p3><sg><gen>
сүйкімді <adj>
сәби<n><pl><px3sg><nom>
бар<adj>
.<sent>

Көрсетілген кестенің мағынасы (тегтері) келесідей: <prn>pronoun, <pers>person, <p3>person 3, <sg>singular, <gen>genitive, <adj>adjective, <n>noun, <pl>plural, <px3sg> person3singular, <nom>nominative,<sent>-sentence.

Үшінші тапсырманың соңында «*Менің сүйкімді балам бар*» сөйлеміндегі "*балам*" сөзі қате аударылғаны анықталып, зерттеу нәтиже соңында *«Менің сүйкімді сәбиім бар*» деген дұрыс түзетілген сөйлем алынды. Сонымен «*Менің сүйкімді балам бар*» бастапқы сөйлемі «*Менің сүйкімді сәбиім бар*» сөйлеміне түзетілді.

Қорыта келгенде, ұсынылған пост-редакциялау технологиясы нәтижесінде түзетілген сөйлемдердің қысқаша тізімін көруге болады (5.7-кесте).

Кесте 5.7 – Пост-редакциялау технологиясы нәтижесінде түзетілген сөйлемдер тізімі

Бастапқы сөйлем	**Google MA-мен аударылған сөйлем**	**PE-LC технологиясы көмегімен шығарылған сөйлем**	**Қате аударылған сөздердің синонимдер ықтималдық көрсеткіші**
You are the most *beautiful* woman I have ever seen in my life.	Сіз менің өмірімде көрген ең *әдемі* әйелсіз.	Сіз менің өмірімде көрген ең *сұлу* әйелсіз.	$P(s_1 \mid x)$=0,00132 (әдемі) $P(s_2 \mid x)$=0,00471 (сұлу) $P(s_3 \mid x)$=0,00087(тартымды) $P(s_4 \mid x)$=0,00014(әсем)

5.7-кестенің жалғасы

1	2	3	4
			$P(s_5 \mid x)$=0,0014(талғампаз) $P(s_6 \mid x)$=0,0008(сүйкімді) $P(s_7 \mid x)$=0,00017(сымбатты)
Algorithms and *data* structures are central to computer science.	Алгоритмдер мен *деректер* құрылымдары информатика үшін орталық болып табылады.	Алгоритмдер мен *мәліметтер* құрылымдары информатика үшін орталық болып табылады.	$P(s_1 \mid x)$=0,00198(дерек) $P(s_2 \mid x)$=0,00243(мәлімет) $P(s_3 \mid x)$=0,00187(материал) $P(s_4 \mid x)$=0,00063(ақпарат)
This year has become a year of comprehensive transformations and *genuine* renewal.	Биылғы жыл жан-жақты қайта құрулар мен *шынайы* жаңару жылы болды.	Биылғы жыл жан-жақты түрлендірулер мен *нақты* жаңару жылы болды.	$P(s_1 \mid x)$=0,00143(шынайы) $P(s_2 \mid x)$=0,00281(нақты) $P(s_3 \mid x)$=0,00017(заңды)

Осы үшінші тапсырманың нәтижесінде семантикалық текше әдісін қолдану арқылы қате аударылған сөздің мағынасы жағынан жақын нұсқасын тауып, түзетілген дұрыс толық сөйлем көрсетілді.

Негізгі пост-редакциялау кезеңінде келесі қадамдар бар:

- Біріншіден, ағылшын тілінен қазақ тіліне қате аударылған сөзді табу керек.
- Одан кейін екінші кезекте синонимдер каталогын іздеу және қалыптастыру; қате аударылған сөз табылса, синонимдерді табу үшін бұл сөз каталогтан ізделеді.
- Үшінші қадамда жүйе семантикалық текше әдісін қолданып, қолайлы синонимді таңдап, қате аударылған сөзді осы сәйкес синоним арқылы жаңартады.

Ұсынылып отырған ғылыми жұмыстың PE-LC технологиясы жоғарыда аталған қадамдарды іске асырып, жақсы нәтижелер көрсетті. Бұл туралы мағлұматты әртүрлі есептеу көрсеткіштері мен сапа бағалау көрсеткіштері туралы келесі бөлімде толығырақ қарастырылған.

Бесінші тараудың қорытындысы

Бұл бесінші тарауда табылған қате сөздердің синонимдерінің ішінен мағынасы жағынан ең жақын синоним сөзді таңдау үшін семантикалық текше әдісіне негізделген лексикалық таңдау тапсырмасы қарастырылды. Қорыта келгенде бұл тарауда:

- табылған қате сөздерді түзету мақсатында максималды энтропияға негізделген семантикалық текше үлгісі мен алгоритмі қолданылып, мысалдар келтірілді.
- ағылшын – қазақ тіліндегі сөздердің 250000 сөйлемнен тұратын ағылшын-қазақ корпустың леммаларын табу үшін және сөздерге морфологиялық генератор қолдану мақсатында Апертиум платформасы қолданылды.
- ұсынылған технологияның алгоритімінің орындалу реті көрсетілді.

6 ҰСЫНЫЛҒАН PE-LC ТЕХНОЛОГИЯСЫН БАҒАЛАУ

6.1 PE-LC технологиясына әртүрлі бағалау көрсеткіштерін қолдану

Ұсынылған PE-LC технологиясын бағалауды орындау үшін 25 000 сөйлемді қамтитын сынақ жинағы жүргізілді. Жұмыстың мақсаты мәтінді ағылшын тілінен қазақ тіліне аудару болғандықтан кіріс сөйлем ретінде тек ағылшын тіліндегі сөйлемдер қолданылды. Сөйлемдер ағылшын тіліндегі әртүрлі жаңалықтар порталдары[6], грамматикалық сайт[7] және әдеби дереккөздер[8] сайттардан алынды. Ертегілер, үкімет, жаңалықтар[9], статистика[10], тарих[11] және құқық туралы сайттардан да сөйлемдер жинастырылды. Сәйкес баламаларды анықтауда каталогты толық ету үшін сөйлемдердің әртүрлі түрлері қосылды. Бұл екі тілді дайын дереккөздер емес, өйткені кез-келген сөйлемнің дұрыс анық сілтемесі бар аудармаларын алу қиындық тудырады. Сондықтан тек ағылшын тіліндегі сөйлемдер алынып, ұсынылған әдіс бойынша қате аударылған сөздер табылды.

2017 мен 2019 жылдардағы мәліметтер бойынша Google, Prompt МА-лар мен PE-LC технологиясының BLEU сапа көрсеткіші 6.1 – кестеде көрсетілген. Ұсынылған PE-LC технологиясының сапа көрсеткіші нәтижесінде 2017 жылы Google МА-дан +6%, Prompt МА-дан +11% жақсы нәтиже көрсетті. 2019 жылы PE-LC технологиясының эксперимент нәтижесінде сапа көрсеткіші +3,88% және +8,37%-ға жетті (6.1-сурет). Бұл жылдары тестілеу үшін ағылшын – қазақ қарапайым, синоним сөздері жиі кездеспейтін сөйлемдері қолданылды.

Кесте 6.1 – 2017, 2019 жылдардағы BLEU сапа бағалау көрсеткіштер нәтижелері

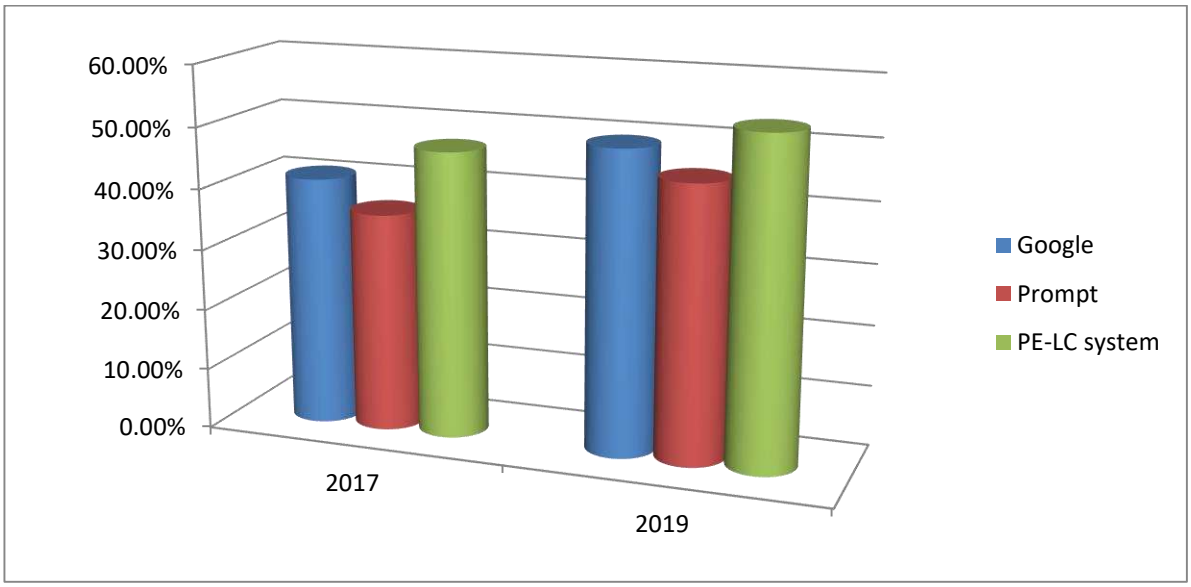
Машиналық Аударма құралы	BLEU көрсеткіші
Google 2017ж.	41,01%
Prompt 2017ж.	36%
PE-LC system, 2017 ж.	47,01%
Айырмашылығы 2017 ж.	+6% Google-ден +11% Prompt-тен
Google 2019	49,52%

[6] https://qazaqtv.com/en/news
[7] https://englishgrammarhere.com/example-sentences/50-examples-of-simple-sentences/Kazakh
[8] https://americanliterature.com/
[9] https://www.akorda.kz/en, Tengrinews.kz
[10] https://www.resourcedata.org/dataset/
[11] https://e-history.kz/ru/back translation

6.1-кестенің жалғасы

1	2
Prompt 2019	45,03%
PE-LC system, 2019 ж	53,4%
Айырмашылығы 2019 ж.	+3,88% Google-ден +8,37% Prompt-тен

2020-2022 жылдар аралығындағы тестілеу кезінде WER (сөз қателік деңгейі) [67], TER (аударма қателік деңгейі) [68] және BLEU [69] көрсеткіштерін есептеу үшін сынақ мазмұны әртүрлі салалардағы жаңа жаңалықтарды[12], синонимдері[13] бар әртүрлі сөйлемдерді және ертегілерді қамтыды(6.2-кесте). Тестілеу кезінде қолданылған мәтіндерді мына сілтемеден көруге болады. *https://github.com/assem7shormak/Data-set.*



Сурет 6.1 – 2017, 2019 жылдардағы BLEU сапа бағалау көрсеткіштерінің нәтижелер диаграммасы

Осы жұмыстың нәтижесінде ағылшын тілінен қазақ тіліне аударылған мәтін МА арқылы талданды. Бір МА жүйесімен ұсынылған жүйенің BLEU, TER және статистикалық маңыздылығын есептеуге көмектесетін онлайн құрал қолданылды. Мұнда нәтижелерді алу үшін Sacrebleu

[12] https://astanatimes.com/

[13] https://sentence.yourdictionary.com/task, https://englishgrammarhere.com/synonyms/50-examples-of-synonyms-with-sentences/, https://kk.opentran.net/, https://parenting.firstcry.com/articles/10-popular-fairy-tale-stories-for-kids/

(https://github.com/mjpost/sacrebleu) бағдарламалық құралы пайдаланылды (6.2-кесте). Нәтижелер 6.2-кестеде көрсетілген. Сынақ корпусының дереккөзі жаңалықтар порталдарынан алынды.

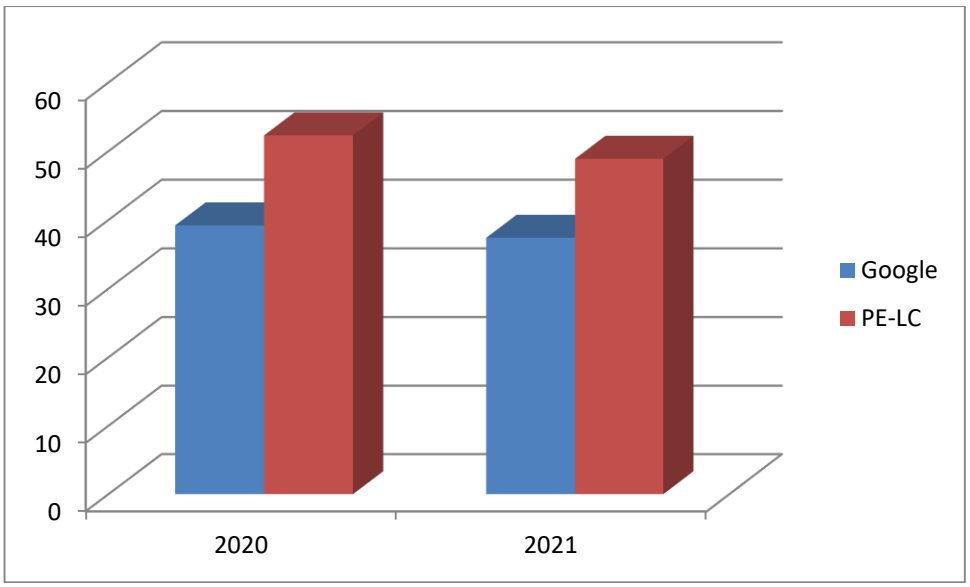
Кесте 6.2 – Екі жүйеге арналған сапа бағалау көрсеткіштер нәтижелері

Машиналық Аударма құралы	BLEU көрсеткіші	Аударма қателік деңгейі, TER	Сөз қателік деңгейі, WER
Google Translate, 2020ж.	39,2%	-	-
PE-LC system, 2020 ж.	52,3%	-	-
Айырмашылығы 2020 ж.	+13,1%		
Google Translate, 2021ж.	37,4%	-	-
PE-LC system, 2021 ж.	48,9%	-	-
Айырмашылығы 2021 ж.	+11,5%		
Google Translate, 2022ж.	41,01%	40,63%	29,59%
PE-LC system, 2022 ж.	47,48%	35,97%	25,31%
Айырмашылығы 2022 ж.	+6,47%	-4,66	-4,28

2020-2021 жылдары алынған зерттеу нәтижелерінің сапа көрсеткішін келесі 6.2-суретте көруге болады. Бұл жылдары Google MA мен PE-LC технологиясының сапа көрсеткіштері салыстырылды. Зерттеу нәтижесінде PE-LC технологиясы Google MA-дан +13,1% жақсы көрсеткіш көрсетті. 2020-2022 жылдар аралығындағы тестілеу кезінде WER (сөз қателік деңгейі) [67], TER (аударма қателік деңгейі) [68] және BLEU [69] көрсеткіштерін есептеу үшін сынақ мазмұны әртүрлі салалардағы жаңа жаңалықтарды[14], синонимдері[15] бар әртүрлі сөйлемдерді және ертегілерді қамтыды (6.3-сурет). Тестілеу кезінде қолданылған мәтіндерді мына сілтемеден көруге болады. *https:// github.com/ assem7shormak/Data-set.*

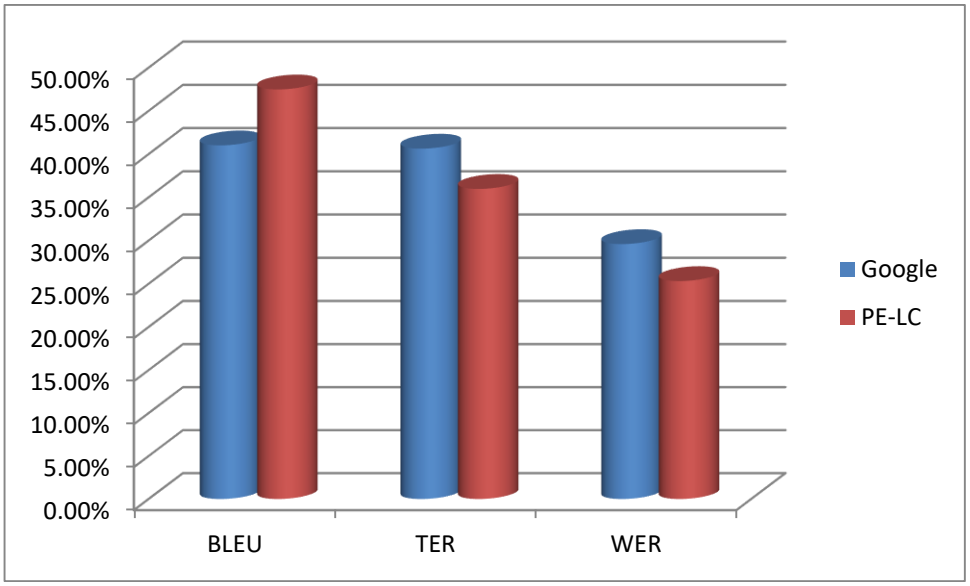
[14] https://astanatimes.com/

[15] https://sentence.yourdictionary.com/task, https://englishgrammarhere.com/synonyms/50-examples-of-synonyms-with-sentences/, https://kk.opentran.net/, https://parenting.firstcry.com/articles/10-popular-fairy-tale-stories-for-kids/



Сурет 6.2 – 2020, 2021 жылдардағы BLEU сапа бағалау көрсеткіштерінің нәтижелер диаграммасы

МА мен сөйлемнің пост-редакциялау сапасы TER, WER, BLEU жүйесі арқылы анықталды (Papineni K. et al. 2002). Салыстыру үшін Google аударма жүйесі де қолданылды. 6.2-кестеге сәйкес, қазақ тіліндегі пост-редакцияланған мәтін мен Google аударған мәтін арасындағы BLEU метрикасының айырмашылығы 2022 жылы 6,47 пайызды құрады. Бұл деректер 2022 жылдың 18 ақпанында жиналды.



Сурет 6.3 – 2022 жылдардағы BLEU,TER,WER сапа бағалау көрсеткіштерінің нәтижелер диаграммасы

Сондай-ақ, аударма қателік деңгейі (TER) метрикасы қолданылды, бұл әдісті МА мамандары пост-редакциялаудың қажеттілік көлемін анықтау үшін

қолданады. Ал ұсынылған PE-LC жүйесінен кейін Google Translate жүйесі PE-LC жүйесінен 4,66 пайызға артық көрсеткішті көрсетті. TER көрсеткішінің азаюы – бұл жақсырақ нәтиже дегенді білдіреді; PE-LC жүйесінде TER кездеседі, ол 4,66 пайызға төмен.

WER сөз қателік деңгейін есептеу үшін дұрыс мәтін мен Google шығарған мәтін қарастырылды. Мысалы, бір аударылған сөйлемде екі қате сөз табылса, ол санды сегізге (бастапқы сөйлемдегі сөздердің жалпы саны) бөлу арқылы сөз қатесінің шамамен 25 пайыздық деңгейін көруге болады. Бірақ бұл бір сөйлем үшін ғана. Зерттеу жұмысында 100 сөйлем үшін WER, TER және BLEU метрика мәнін Sacrebleu онлайн бағдарламаны қолданылып, мәндері есептелінді. Жүйелердің пайыздық көрсеткіштерін есептеу үшін синонимдері бар сынақ корпусындағы сөйлемдерде лексикалық таңдау мәселесінің қаншалықты шешілгенін көру үшін пайдаланылды. Google МА-ның орташа өлшенген WER көрсеткіші нәтижесі 29,59 пайызды құрады. WER көрсеткіші ағылшын тілінен қазақ тіліне аударылған мәтінге арналған PE-LC технологиясы Google Translate аудармасының мәтінін 4,28 пайызға жақсырақ екенін көрсетті.

Статистикалық маңыздылық екі жүйе (Google мен PE-LC) үшін де есептелінді. МА бағалау көрсеткішіндегі жақсарту – байқалған жақсартудың *p* кездейсоқ нәтиже болу ықтималдығы (нөлдік гипотезаның ықтималдығы) *p* мәнінен аз болғанда статистикалық маңызды болып табылады. Мұнда экстремалды – сынақ статистикасының нөлдік гипотезадан ауытқу дәрежесі. BLEU сенімділігін жұптастырылған жүктеу жолағын қайта үлгілеу (bootstrap resampling) арқылы есептедік. Содан кейін Google және PE-LC жүйелері арасында статистикалық маңыздылық тексерілді. 6.3-кестеде n-грамм сөздері үшін BLEU, TER және chrF2(статистикалық маңыздылық тест көрсеткіші) нәтижесі алынды. Google үшін PE-LC үлгісінде статистикалық маңыздылық тесті жасалынды. 6.3-кестеде берілген нәтижелерді алу үшін Sacrebleu қолданылды.

Кесте 6.3 – 100 сөйлемнің Google , PE-LC технологиясы үшін статистикалық маңыздылық көрсеткіш кестесі (2022 ж.)

BLEU (μ ± 95% CI)	TER (μ ± 95% CI)	chrF2 (μ ± 95% CI)
Google 41.0103 (40.9405 ± 6.0153)	Google 40.6349 (40.6292 ± 4.9062)	Google 70.3059 (70.2557 ± 3.4295)
PE-LC model 47.4809 (47.4171 ± 5.8691) **(p = 0.0010)***	PE-LC model 35.9788 (35.9826 ± 4.8104) **(p = 0.0010)***	PE-LC model 75.6458 (75.6220 ± 2.9849) **(p = 0.0010)***

Ұсынылып отырған ғылыми жұмыстың соңында алынған нәтижелер Google МА қызметін 6 пайызға жақсартты. Үш метрикасының (көрсеткішінің)

нәтижелеріне сәйкес, Bleu бұл ұсынылған PE-LC технологиясы аударма сапасын 6 пайызға жақсартатынын, ал қалған екі (WER, TER) метрикада кездесетін қателер Google MA-дан әлдеқайда аз екенін көруге болады.

Алтыншы тараудың қорытындысы

Бұл алтыншы тарауда ұсынылған PE-LC технологиясының бағалау көрсеткіштері сипатталған:

- сынақ жүргізу мақсатында түрлі онлайн ресурстардан ағылшын сөйлемдері алынды.
- 2017 , 2019-2022 жылдардағы Google, Prompt MA-лар мен PE-LC технологиясының BLEU сапа көрсеткіші көрсетілді. Ұсынылған PE-LC технологиясының 2017 жылғы сапа көрсеткіші Google MA-дан +6%, Prompt MA-дан +11% жақсы нәтиже, 2019 жылы +3,88% және +8,37%, 2020 жылы Google MA-дан +13,1%, 2021 жылы Google MA-дан +11,5% нәтижелер алынғаны жазылған.
- PE-LC технологиясын және Google MA-сына WER (сөз қатесінің деңгейі), TER (аударма қатесінің деңгейі) және BLEU көрсеткіштерін есептеу үшін мазмұны әртүрлі салалардағы мәтіндер алынып сынақ жүргізілді. PE-LC технологиясы 2022 жылы Google MA мәтінінің BLEU метрикасының көрсеткіші 6,47 пайызға жоғары көрсеткіш, TER 4,66 және WER 4,28 пайызға төмен көрсеткіш көрсетті.
- 100 сөйлемнен тұратын сынақ жүргізіліп, PE-LC технологиясының статистикалық маңыздылығы анықталды.

ҚОРЫТЫНДЫ

Қазіргі заманғы қарқынды даму талабына сай МА саласы да қарқынды дамуда. Зерттеу жұмысының барысында ағылшын тілі мен қазақ тілінің ерекшеліктері ескеріліп, қазақша мәтініне лексикалық таңдау әдісі негізінде (PE-LC) пост-редакциялау технологиясы жасалды. Бұл диссертациялық жұмыста Apertium платформасы қазақ тілінің морфологиялық ерекшеліктері ескеріліп түбірлерін табу үшін қолданылды.

Ұсынылып отырған зерттеу жұмысының соңында келесі ғылыми нәтижелер алынды:

1. Алғаш рет ағылшын – қазақ машиналық аударманың PE-LC пост-редакциялау технологиясы әзірленді.
2. Ағылшын тілінен қазақ тіліне қате аударылған сөздерді табу әдісі кері аудармамен жетілдірілді.
3. Қате аударылған қазақ тіліндегі сөздер синонимдерінің каталогын автоматты түрде қалыптастыру сұлбасы мен алгоритмі жасалды.
4. Қате аударылған сөздің ықтималдығы жоғары синоним сөзді таңдау семантикалық текше әдісінің үлгісі мен алгоритмі бейімделді.

Ұсынылған PE-LC технологиясының 2017 жылғы BLEU сапа көрсеткіші Google МА-дан +6%, Prompt МА-дан +11% жақсы нәтиже, 2019 жылы +3,88% және +8,37%, 2020 жылы Google МА-дан +13,1%, 2021 жылы +11,5% нәтижелер алынды. 2022 жылы эксперимент нәтижелері бойынша PE-LC пост-редакциялау технологиясының BLEU сапа көрсеткіші Google МА-дан +6,47% жақсы нәтиже көрсетті. PE-LC технологиясының сапа көрсеткіштері Google МА-дан TER -4,66% және WER -4,28% төмен көрсеткіш көрсетті, яғни бұл ұсынылып отырған технологияның салыстырмалы түрде жақсырақ нәтиже алынғанын көрсетеді.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. Комиссаров В. Н. (1990). Теория перевода (лингвистические аспекты). – М.: Высшая школа, - 253 с

2. Новожилова А.А. (2014). "Машинные системы перевода: качество и возможности использования". *Вестник Волгоградского государственного университета.* Сер. 2, Языкознание, № 3. 67-73с.

3. Пиванова Э. В. Теория и практика машинного перевода : учебное пособие / авт.-сост. Пиванова Э. В. - Ставрополь: Изд-во СКФУ, 2014. - 114 с.

4. Абеустанова(Шормакова) А.Н. "Машиналық аударманың нарықтағы және Қазақстандағы күйі". *ҚазҰТУ хабаршысы* № 6(106), 2014. 150-152–б.

5. Shormakova A. "Machine translation and post-editing". *Материалы международной конференции студентов и молодых ученых «Мир науки»*, 17-19 апреля 2013г. – Алматы: Қазақ университеті, 2013. – с. 222

6. Шормакова А.Н. "Информатика терминдерінің мемлекеттік тілге аудару ерекшеліктері". *Материалы III международного конгресса студентов и молодых ученых «Мир науки»*, 23-28 апреля, 2009г.-Алматы: Қазақ университеті, - с. 249.

7. Андреева А. Д. Обзор систем машинного перевода // Андреева А. Д., Меньшиков И. Л., Мокрушин А. А. — М.: Молодой ученый, 2013. — № 12. — с. 64– 66.

8. Кулагина О. С. Исследования по машинному переводу / Кулагина О.С., - Москва: Наука, 1979. – 320 с.

9. Леонтьева Н. Н. Автоматическое понимание текстов: системы, модели, ресурсы: учеб. пособие для студ. лингв, фак. вузов / Леонтьева Н. Н. — М.: Издательский центр «Академия», 2006. - 304 с.

10. Новожилова А. А. Информационные технологии в переводе : учебнометодическое пособие / Новожилова А. А., Степанова Е. В., Шовгенина Е. А. - Волгоград : Изд-во ВолГУ, 2012. - 159 с.

11. Bing машиналық аударма [Электрондық ресурс] https://www.bing.com/ Сұраныс күні: 18.05.22.

12. Translatedict машиналық аударма [Электрондық ресурс] https:// www.translatedict.com/ Сұраныс күні: 18.05.22.

13. Microsoft машиналық аударма [Электрондық ресурс] https:// translator.microsoft.com/ Сұраныс күні: 18.05.22.

14. Яндекс машиналық аударма [Электрондық ресурс] https:// translate.yandex.ru/ Сұраныс күні: 18.05.22.

15. Deepl машиналық аударма [Электрондық ресурс] https:// www.deepl.com/translator Сұраныс күні: 18.05.22.

16. Хроменков П. Н. Современные системы машинного перевода: учеб. пособие / Хроменков П. Н. - Москва : Изд-во МГОУ, 2005. - 159 с.

17. Hutchins J.W. "The development and use of machine translation systems and computer-based translation tools". *International Symposium on Machine Translation and Computer Language Information Processing*. Beijing, China, 26-28 June 1999.

18. Bowker L. Computer-aided Translation Technology: A Practical Introduction, University of Ottawa Press, 2002

19. Максютина О. В. "Редактирование перевода как неотъемлемая часть современного стандарта качества". *Вестник ТГПУ*. - 2014. - №4 (145). - с. 106-111.

20. Нечаева Н.В., Светова С.Ю. "Постредактирование машинного перевода как актуальное направление подготовки переводчиков в вузах" Вопросы методики преподавания в вузе. - 2018. - №7 (25). - с. 64-72.

21. Koponen M. (2016). "Machine translation post-editing and effort: Empirical studies on the post-editing process". (Doctoral thesis, University of Helsinki, Helsinki, Finland). Retrieved from http://hdl.handle.net/10138/160256

22. Шормакова А.Н., Тукеев У.А. "Технология машинного перевода с обучением английского языка на казахский язык". *Материалы международной конференции студентов и молодых ученых «Мир науки»*, 23-26 апреля 2012г. – Алматы: Қазақ университеті, – с. 154.

23. Papineni K., Roukos S., Ward T., Henderson J. and Reeder F. 2002. Corpus-based comprehensive and diagnostic MT evaluation: Initial Arabic, Chinese, French, and Spanish results. *In Proceedings of Human Language Technology* 2002, San Diego, CA.

24. Barrault L., Bojar O.,Costa-jussà M. R., Federmann C., Fishel M., Graham Y., Haddow B., Huck M., Koehn P., Malmasi S., Monz C. (August 2019). "Findings of the 2019 Conference on Machine Translation (WMT19)". *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1). Florence, Italy: Association for Computational Linguistics:* 1–61. doi:10.18653/v1/W19-5301

25. Carmo F., Shterionov D., Moorkens J., Wagner J., Hossari M., Paquin E., Schmidtke D., Groves D., Way A. A review of the state-of-the-art in automatic post-editing. Machine Translation (2021) 35:101–143. https://doi.org/10.1007/s10590-020-09252-y

26. Mundt J. Learning to Automatically Post-Edit Dropped Words in MT. *Association for Machine Translation in the Americas* (AMTA 2012), Columbia San Diego, California, USA. https://aclanthology.org/2012.amta-wptp.5/

27. Simard M., Foster G. (2013). Pepr: post-edit propagation using phrase-based statistical machine translation. *In: Proceedings of the XIV machine translation summit*

28. Ortiz-Martínez D., Casacuberta F. (2014). The new hot toolkit for fully-automatic and interactive statistical machine translation. *In: Proceedings of the demonstrations at the 14th conference of the European chapter of the association for computational linguistics*, Association for Computational Linguistics, Gothenburg, Sweden, pp 45–48. https ://doi.org/10.3115/v1/E14-2012, https ://www.aclweb.org/anthology/E14-2012

29. Lagarda A.L, Ortiz-Martínez D., Alabau V., Casacuberta F. (2015). Translating without in-domain corpus: machine translation post-editing with online learning

techniques. Comput Speech Lang 32(1):109– 134. https ://doi.org/10.1016/j.csl.2014.10.004

30. Chatterjee R., Gebremelak G., Negri M., Turchi M. (2017). Online automatic post-editing for MT in a multi-domain translation environment. *In: Proceedings of the 15th conference of the European chapter of the association for computational linguistics*: volume 1, long papers. Association for Computational Linguistics, Valencia, Spain, p. 525–535. https ://www.aclwe b.org/anthology/E17-1050

31. Pal S., Naskar S. K., Vela M., Genabith J. "A Neural Network based Approach to Automatic Post-Editing". In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, (Berlin, Germany, August 7-12, 2016) , p. 281–286.

32. Junczys-Dowmunt M. and Grundkiewicz R. 2016. "Log-linear combinations of monolingual and bilingual neural machine translation models for automatic post-editing". In *Proceedings of the First Conference on Machine Translation. Association for Computational Linguistics*, Berlin, Germany, p. 751–758. http://www.aclweb.org/anthology/W16- 2378.

33. Pérez-Ortiz J. A., Torregrosa D. and Forcada M. "Black-box integration of heterogeneous bilingual resources into an interactive translation system", in *EACL 2014 Workshop on Humans and Computer-assisted Translation* (Gothenburg, April 26, 2014)

34. Hokamp C. "Ensembling Factored Neural Machine Translation Models for Automatic Post-Editing and Quality Estimation". *Computation and Language (cs.CL)*, arXiv preprint arXiv: 1706.05083, 2017.

35. Negri M, Turchi M, Bertoldi N, Federico M (2018b). Online neural automatic post-editing for neural machine translation. In: Cabrio E, Mazzei A, Tamburini F (eds) *Proceedings of the fifth Italian conference on computational linguistics* (CLiC-it 2018), Torino, Italy, December 10–12, 2018. CEURWS.org, *CEUR Workshop Proceedings*, vol 2253, http://ceur-ws.org/Vol-2253/paper 63.pdf

36. Chatterjee R., Negri M., Rubino R. and Turchi M. 2018. Findings of the WMT 2018 Shared Task on Automatic Post-Editing. *In Proceedings of the Third Conference on Machine Translation, Volume 2: Shared Task Papers, Brussels, Belgium. Association for Computational Linguistics*.

37. Negri M., Turchi M., Chatterjee R., Bertoldi N. "eSCAPE: a Large-scale Synthetic Corpus for Automatic Post-Editing". In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation* (LREC 2018), Miyazaki, Japan, May 7-12, 2018. *CoRR abs/1803.07274 (2018)*

38. Ortega J., Sánchez-Martínez F., Turchi M., Negri M. (2019). Improving translations by combining fuzzy match repair with automatic post-editing. *In: Proceedings of machine translation summit XVII Volume 1: Research Track, European Association for Machine Translation*, Dublin, Ireland, p. 256– 266. https ://www.aclwe b.org/anthology/W19-6625

39. Gois A., Cho K., Martins A. "Learning Non-Monotonic Automatic Post-Editing of Translations from Human Orderings". In *Proceedings of the 22nd Annual*

Conference of the European Association for Machine Translation (EAMT 2020), p. 205-214 Lisboa, Portugal, November 2020.

40. Bérard A., Pietquin O., Besacier L. (2017). LIG-CRIStAL System for the WMT17 Automatic Post-Editing Task. *In: Proceedings of the second conference on machine translation* (WMT 2017), Copenhagen, Denmark, vol 2, p. 623–629

41. Carmo F., Shterionov D., Wagner J., Hossari M., Paquin E, Moorkens J. (2020). A review of the state-ofthe-art in automatic post-editing. Mach Transl 34.

42. Chatterjee R., Freitag M., Negri M., Marco. (2020). Findings of the WMT 2020 Shared Task on Automatic Post-Editing. *In: Proceedings of the WMT 2020 Automatic Post-Editing Shared Task*. https://www.statmt.org/wmt20/ pdf/2020.wmt-1.75.pdf

43. Yang H., Wang M., Wei D., Shang H., Guo J., Li Z., Lei L., Qin Y., Tao S., Sun S., Chen Y. (2020). HW-TSC's Participation at WMT 2020 Automatic Post Editing Shared Task. *In: Proceedings of the WMT 2020 Automatic Post Editing Shared Task.* https://aclanthology.org/2020.wmt-1.85.pdf

44. Sharma A., Gupta P., Nelakanti A. (2021). Adapting Neural Machine Translation for Automatic Post-Editing. *In: Proceedings of the WMT 2020 Automatic Post Editing Shared Task.* https://aclanthology.org/2021.wmt-1.35.pdf

45. Schwenk H., Chaudhary V., Sun S., Gong H. and Guzmán F. (2021). WikiMatrix: Mining 135M Parallel Sentences in 1620 Language Pairs from Wikipedia. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1351–1361, Online. Association for Computational Linguistics.

46. Rakhimova, D., Karyukin V., Karibayeva A., Turarbek A., Turganbayeva A. The Development of the Light Post-editing Module for English-Kazakh Translation In *Proceeding:The 7th International Conference on Engineering & MIS 2021*, 1-5

47. Bao X. L. (2015). Analysis on Lexical Errors in College English Writing. Canadian Social Science, 11(12), 127-130. Available from: http:// www.cscanada.net/index.php/css/article/view/7775 DOI: http://dx.doi.org/10.3968/ 7775

48. Esplà-Gomis M., Sánchez-Martínez F., Forcada M.L. (2012). "A Simple Approach to Use Bilingual Information Sources for Word Alignment", *Procesamiento del Lenguaje Natural*, 49, 93-100

49. Esplà-Gomis M., Sánchez-Martínez F., Forcada M.L. (2015). "Using Machine Translation to Provide Target-Language Edit Hints in Computer Aided Translation Based on Translation Memories", *Journal of Artificial Intelligence Research*, 53, 169-222

50. Tukeyev U., Amirova D., Karibayeva A., Sundetova A., Abduali B. "Combined Technology of Lexical Selection in Rule-Based Machine Translation", in *Lecture Notes of Artificial Intelligence* (LNAI) vol. 10449, Part 2, Springer, 2017, *The International Conference on Computational Collective Intelligence* , p. 491-500, DOI: 10.1007/978-3-319-67077-5_47

51. Tyers F.M., Sánchez-Martínez F., Forcada M.L. , "Unsupervised training of maximum-entropy models for lexical selection in rule-based machine translation", in *Proceedings of EAMT 2015, The Eigtheenth Annual Conference of the European Association for Machine Translation* (Antalya, May 11-13, 2015) , p. 145-152

52. Forcada M. L., Ginestí-Rosell M., Nordfalk J., O'Regan J., Ortiz-Rojas S., Pérez-Ortiz J. A., Sánchez-Martínez F., Ramírez-Sánchez G., Tyers F.M. "Apertium: a free/open-source platform for rule-based machine translation", *Machine Translation*, (Special Issue on Free/Open-Source Machine Translation) 25:2, 127-144

53. Шормакова А.Н., Айткулова А. "Добавление новой англо-казахской языковой пары в платформу машинного перевода Апертиум". *51-я Международная научная студенческая конференция «Студент и научно-технический прогресс»*, Новосибирск, 12-18 апреля 2013, Секция "Информационные технологии". - с. 241.

54. Sundetova A., Forcada M.L., Shormakova A., Aitkulova A. "Structural transfer rules for English-Kazakh machine translation in the free/open-source platform Apertium", in *Proceedings of the I International Conference on Computer Processing of Turkic Languages (TurkLang-2013)* (Astana, 3-4 oct. 2013) , p. 322-331.

55. Тукеев У.А., Абеустанова (Шормакова) А.Н., Сундетова А. "Ағылшын – қазақ тілдік жұбы үшін Apertium платформасындағы сөйлемді синтаксистік құрылымдық түрлендіру ережелері және мәселелері». *IV международная научно-практическая конференция:* (секция «Искусственный интеллект»). Қоғамды ақпараттандыру IV Халықаралық ғылыми-практикалық конференция еңбектері, Астана 2014 , 127-129 б.

56. Шормакова А.Н. "Екі табиғи тілдегі аударылған мәтінді туралау". *ҚазҰТУ хабаршысы*, №4(128), 2018. 344-349–б.

57. Miller G. A. 1995. Wordnet: A lexical database for English. *Communications of the ACM,* 38(11):39–41.

58. Онлайн сөздік [Электрондық ресурс]: https://www.thesaurus.com/ Сұраныс күні: 18.02.22.

59. Питон кітапханасы [Электрондық ресурс]: https://www.crummy.com/software/BeautifulSoup/ Сұраныс күні: 18.02.22.

60. Абеустанова А.Н. "Ағылшын тілінен қазақ тіліне аударылған қазақша қате сөздерді анықтау және баламалар каталогын құру". *ҚазҰТУ хабаршысы* №6 2017. 313-317–б.

61. Berger A. L., Pietra S. A. D. & Pietra V. J. D. (1996). "A maximum entropy approach to natural language processing". *Computational Linguistics*, 22, 39–71

62. Jurafsky D. and Martin J.H. (2007). Automatic Speech Recognition. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* Prentice Hall, Upper Saddle River.p.213-220

63. Абеустанова(Шормакова) А.Н. "Қазақ тіліндегі көпмағыналы сөздердің бірін анықтаудың бір болжамы". *ҚазҰТУ хабаршысы* №4(110) 2015. 625-628–б.

64. Abeustanova (Shormakova) A., Tukeyev U. “Automatic Post-editing of Kazakh Sentences Machine Translated from English” in *Studies in Computational Intelligence/Advanced Topics in Intelligent Information and Database Systems*, vol. 710 – Springer International Publishing, in *Asian Conference on Intelligent Information and Database Systems* (2017), p. 283-295.

65. Shormakova A., Zhumanov Z.H., Rakhimova D. "Post-editing of words in Kazakh sentences for information retrieval". *Journal of Theoretical and Applied Information Technology,* 2019, 97(6), p. 1896–1908.

66. Ali A. and Renals S. 2018. Word Error Rate (WER) Estimation for Speech Recognition: e-WER. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, p. 20–24, Melbourne, Australia. Association for Computational Linguistics.

67. A Study of Translation Edit Rate with Targeted Human Annotation (TER) [Электрондық ресурс] https://www.cs.umd.edu/~snover/pub/amta06/ter_amta.pdf Сұраныс күні: 18.05.22.

68. Papineni K., Roukos S., Ward T., Zhu W.J. (2002). BLEU: a method for automatic evaluation of machine translation. *In: Proceedings of the 40th annual meeting of association for computational linguistics* (ACL 2002), Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, p. 311–318.

69. Post M. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, p. 186–191, Brussels, Belgium. Association for Computational Linguistics.

ҚОСЫМША А. Бағдарлама коды.

```
#!/usr/bin/python
# coding=utf-8
# -*- encoding: utf-8 -*-
# -*-coding:cp1251-*-
import sys, os, re
#from PyDictionary import PyDictionary
from bs4 import BeautifulSoup
import requests
#==========read from terminal and find root of words from google translator and apertium==========
input_str = sys.argv[1:]
if(len(input_str)==0):
	print "Enter correct sentence!!!"
else:
	source_file=open("source_file.txt", "w")
	source_str=""
	for i in range(len(input_str)):
		if(i==len(input_str)-1):
			source_str=source_str+input_str[i]
		else:
			source_str=source_str+input_str[i]+" "
	source_file.write(source_str)
	source_file.close()
	os.system("cat source_file.txt|apertium -d. eng-kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d '<'>root_source.txt")
	os.system("python translate.py")
```

```
os.system("python reverse.py")
os.system("cat translatedToEnglish.txt|apertium -d. eng-kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d '<'>root_target.txt")
check_list=[]
equal_list=[]
another_list=[]
source_file=set(open("root_source.txt").read().splitlines())
translated_file=set(open("root_target.txt").read().splitlines())
for my_words in source_file:
    for mword in translated_file:
        if my_words==mword and my_words not in equal_list and (my_words!='.'):
            equal_list.append(my_words)
            print "Correct: "+my_words+" "+mword
        #=====================================find different words in sentences==========================================
for my_words in source_file:
    if (my_words not in check_list) and (my_words not in equal_list) and(my_words!='.'):
        print("Different1: "+my_words)
        check_list.append(my_words)

for my_line in source_file:
    for  mword in translated_file:
        if (mword!=my_line) and (mword!='.') and (mword not in another_list) and (mword not in equal_list):
            print "Different2: "+mword
            another_list.append(mword)
```

```
                                check_list.append(mword)
#func with target and source words to analyze them
def func_targetSource(synonyms_list):

        os.chdir("/home/apertium/apertium-kaz")
        kazakh_synonyms=[]
        for i in synonyms_list:
                s_file=open("source_file.txt", "w")
                s_file.write(i)
                s_file.close()
                os.system("python translate.py")
                os.system("cat translated.txt|apertium -d. kaz-morph|sed -e
's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d '<'>root.txt")

                r_file=open("root.txt")
                local_word=r_file.readline().strip()
                r_file.close()
                if("$"  in local_word) or( "*" in local_word) and
(local_word[1:len(local_word)-1] not in kazakh_synonyms):
                        kazakh_synonyms.append(local_word[1:len(local_word)-1])
                else:
                        if(local_word not in kazakh_synonyms):
                                kazakh_synonyms.append(local_word)
        writeToTable(kazakh_synonyms, synonyms_list)

def corpus(kazakh_synonyms, synonyms_list, fileName):
        os.chdir("/home/apertium/apertium-kaz")
```

```
list_count=[]
count1=0
title_file=open("CorpusReady-eng.txt")
r_corpus=[line.decode('utf-8').strip() for line in title_file.readlines()]
for line in r_corpus:
        count1=count1+1
        for i in synonyms_list:
            if(i in line):
                list_count.append(count1)

count2=0
w_result=open("result_kaz.txt", "w")
with open("CorpusReady-kaz.txt", "r") as r_kazCorpus:
    for line in r_kazCorpus:
        count2=count2+1
        for number in list_count:
            if(count2==number):
                w_result.write(line.strip()+"\n")

w_result.close()

r_result=open("result_kaz.txt", "r").readlines()
w_result=open("result_kaz.txt", "a")
with open("CorpusReady-kaz.txt", "r") as r_kazCorpus:
    for line in r_kazCorpus:
        for i in kazakh_synonyms:
            if(i in line and line not in r_result):
```

```
                    w_result.write(line.strip()+"\n")

        w_result.close()

        #==========finding root of result-KAZ=======================

        #os.chdir("/home/apertium/apertium-kaz")

        os.system("cat result_kaz.txt |apertium -d. kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d '<'| cut -f2 -d'.'| cut -f2 -d','| cut -f2 -d'*'| cut -f2 -d'-'|cut -f2 -d'!'|cut -f2 -d':'|cut -f2 -d';'|cut -f2 -d'?'|cut -f2 -d')'|cut -f2 -d('|awk '{print tolower($0)}'|sort -u>table_root.txt")

        func_writeTable(fileName,kazakh_synonyms)
def func_writeTable(fileName, wordList):
                count_list=[]
                w_table=open(fileName, "w")
                #title_file1=open("table_root.txt", "r")
                #title_file2=open("CorpusReady-kaz.txt", "r")
                data=set(open("CorpusReady-kaz.txt").read().splitlines())
                r_rootTable=set(open("table_root.txt").read().splitlines())
                #with open("CorpusReady-kaz.txt") as data:

                for line in data:
                        #with open("table_root.txt") as r_rootTable:

                        for tword in r_rootTable:
                                for aword in wordList:
                                        if("$" in tword):
```

```
                    tword=tword[:len(tword)-1]
                #for aword in list_syn:
                #print("ttword"+ttword+" and aword "+aword)
                if((tword in line) and (aword in line)):
                    count_list.append(tword+" - "+aword)

                #else:
                        #print("There is no coorporate
values in file"+fileName)
                        #w_table.write(aword+" -
"+ttword+"\n")

        for i in count_list:
            w_table.write(i+" : "+str(count_list.count(i))+"\n")
            #print(i+" : "+str(count_list.count(i))+"\n")
        w_table.close()
        #=======write to list or check=========================

def func_found(w):
        with open("tableList", "r") as r_list:
            found=False
            for line in r_list:

                if w in line:
                    found=True
                    return found
            if not found:
```

```
                return found
def synonyms(word):
    res = requests.get('https://www.thesaurus.com/browse/{}'.format(word), verify=False)
    soup = BeautifulSoup(res.text, 'lxml')
    #soup.find('section', {'class': 'css-191l5o0-ClassicContentCard e1qo4u830'})
    slist = [span.text for span in soup.findAll('a', {'class': 'css-1kg1yv8 eh475bn0'})]
    if slist:
        return slist
    return [span.text for span in soup.findAll('a', {'class': 'css-1gyuw4i eh475bn0'})]


def writeToTable(kazakh_synonyms, synonyms_list):
    myStr=""
    fileName="table"
    check=False
    isTableExist=False
    for w in kazakh_synonyms:
        case=func_found(w)
        print("kazakh_synonyms: "+w)
        if(case==False):
            check=True
            myStr=myStr+w+" "
            isTableExist=False
        else:
            isTableExist=True
```

```
        if(isTableExist):
                print "Table exist!!!"
        else:
                w_tableList=open("tableList","a")
                w_tableList.write(myStr+"\n")
                w_tableList.close()

                if(check==True):
                        num_lines = sum(1 for line in open('tableList','r'))
                        open(fileName+str(num_lines),"w").close()
                        corpus(kazakh_synonyms, synonyms_list,
fileName+str(num_lines))



if(len(input_str)!=0):

        #============Find synonyms of the words==================

        #removing articles a, an, the from list
        articles=["a", "an", "the"]
        for article in articles:
                if(article in check_list):
                        check_list.remove(article)
        if(len(check_list)==0):
                        print("There is no any bad words!")
        else:
                #dictionary=PyDictionary()
```

```
#for word in check_list:
        #check=""
        #synonym_list=[]
        #a=[]
        #try:
                #for i in(dictionary.synonym(word)):
                        #i=i.encode("utf-8")
                        #synonym_list.append(i)
                        #a.append(i)
                        #if(i in check_list):
                                #check=word
        #except:
                #print("Error with word: "+word)
                #continue

        #if(check!=""):
                #synonym_list.append(check)
        #if(word not in synonym_list):
                #synonym_list.append(word)
        #print synonym_list
        #func_targetSource(synonym_list)
for word in check_list:
        check=""
        synonym_list=[]

        try:
                for i in(synonyms(word)):
```

```
                encodedString=i.encode("utf-8")
                synonym_list.append(encodedString)

                if(encodedString in check_list):
                        check=word
except:
        print("Error with word: "+word)
        continue

if(check!=""):
        synonym_list.append(check)
if(word not in synonym_list):
        synonym_list.append(word)
print synonym_list
func_targetSource(synonym_list)
```

ҚОСЫМША Б. Каталог пен семантикалық текше құру бағдарлама бөлігі

```
#!/usr/bin/python
# coding=utf-8
# -*- encoding: utf-8 -*-
# -*-coding:cp1251-*-
## -*- coding: utf-8 -*-
import sys, os, re

#Finds which table we need by the kazakh word
def findTable(word):
  my_line=""

  with open("tableList", "r") as r_list:
  #r_list=open("tableList").readlines()
        count=0
        number=0
        for line in r_list:
              count=count+1
              if(word in line):
                    #print(word+"SSS"+str(count))
                    my_line=line
                    number=count
  return number, my_line


#counting each word of sentence by the table
def func_count_general(word):

  sumNumber=0
  with open("tableResult") as r_result:
        for line in r_result:
              split=line.split()
              for i in range(0,len(split), 5):
                    if(word==split[i+2]):
                          sumNumber+=int(split[i+4])
  return sumNumber

#function of morphology
def find_between( s, first, last ):
  try:
        start = s.index( first ) + len( first )
        end = s.index( last, start )
        return s[start:end]
```

```
  except ValueError:
        return ""

def main_morph():
        #Finding morphology syntaxys of given sentence
        os.chdir("/home/apertium/apertium-kaz")
        os.system("cat translated.txt |apertium -d. kaz-morph >morph_file.xml")
        arr=[]
        file_xml = open('morph_file.xml','r')
        file_a=open('main_morph', 'w')

        for vals in file_xml:
                val = vals.split('$')

                for w in val:
                        arr.append(w+"null")
        for v in arr:

                res=find_between(v, "/", "null" )
                if '>/' in v:
                        res=find_between(v, "/", ">" )

                        if len(res)>1:
                                file_a.write(find_between(v, "/", "/" )+"\n")
                elif('>null' in v):

                        res=find_between(v, "/", ">" )

                        if len(res)>1:
                                file_a.write(find_between(v, "/", "null" )+"\n")

        file_a.close()

def morph(my_list, maxString, maxNumber):
        maxLine=""
        #putting max string into morphology syntax
        res=[]
        main_morph()
        with open("main_morph", "r") as r_morph:
                for line in r_morph:

                        mLine=line.strip()
                        check=False
```

```
                        myWord=""
                        myString=""
                        for word in my_list:
                                #print("line "+word+" "+line)
                                if(word in mLine):

                                        myWord=word
                                        if(maxString+mLine[len(word):] not in res):
                                                if(maxString!="nothing"):
                                                        print("Max String is: "+maxString+" and
its number: "+str(maxNumber))
                                                        maxLine                                 =
maxString+mLine[len(word):]

  res.append(maxString+mLine[len(word):])
                                else:
                                        myString=mLine
                        if(maxString+mLine[len(myWord):] not in res):
                                res.append(myString)


        #Result of work. Shows the correct sentence

        if(len(res)!=0):
                if(maxLine not in res):
                        print ("The sentense is the same as sourse sentense!")
                w_main_morph = open("main_morph", "w")
                for index in range(len(res)):
                        #print          predWord+"          and          "+res[number]+"NN
"+str(number)+"AAAA"
                        if(index > 0):
                                predWord = res[index-1]
                                if(predWord != res[index]) :
                                        #print     predWord+"     and     "+res[number]+"NN
"+str(number)
                                        w_main_morph.write(predWord+"\n")
                                        os.system("echo     '^"+predWord+"$'|hfst-proc     -g
/home/apertium/apertium-eng-kaz/eng-kaz.autogen.hfst")
                                        if(index == len(res)-1):
                                                w_main_morph.write(res[index]+"\n")
                                                os.system("echo '^"+res[index]+"$'|hfst-proc -g
/home/apertium/apertium-eng-kaz/eng-kaz.autogen.hfst")
  #print str(number)
```

```
                w_main_morph.close()
                print
("================================================================
================================================")

#main process
def func_count(fileName, my_line, badWordKz):#my_line is line of synonyms from
tablelist
  print("You opened file "+fileName)
  sfile=open("source_file.txt", "w")
  sfile.write(main_text)
  sfile.close()
  os.system("python translate.py")
  os.system("cat translated.txt|apertium -d. kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'| cut -f2
-d'/'|cut -f1 -d '<'>root.txt")
  r_table=set(open("root.txt").read().splitlines())

  #readTableList and counts each line
  w_result=open("tableResult", "w")
  tableResult=[]
  myWord=""
  my_list=my_line.split()
  #my_list.append(badWordKz)
  myTable=set(open(fileName).read().splitlines())
  for line in myTable:
        split=line.split()
        for line2 in my_list:
              for line3 in r_table:
                    word=line3.strip()
                    if(len(split)==5):
                          for i in range(0,len(split), 5):
                                if(word==split[i] and word!=split[i+2]):
                                      if((split[i]+" - "+split[i+2]+" : "+str(split[i+4]))
not in tableResult):
                                            tableResult.append(split[i]+"
-
"+split[i+2]+" : "+str(split[i+4]))
                                            w_result.write(split[i]+" - "+split[i+2]+"
: "+str(split[i+4])+"\n")

  w_result.close()


  #finding Maximum value from couting and string name
```

```
  lMax=[]
  for k in my_list:
        count=func_count_general(k)
        print(k+" - "+str(count))
        lMax.append(k+" - "+str(count))
  maxNumber=0
  maxString="nothing"
  for line in lMax:
        myLine=line.split()
        #print(str(len(myLine[2])))

        for i in(0, len(myLine), 3):
            if(int(myLine[2])>maxNumber):
                maxNumber=int(myLine[2])
                maxString=myLine[i]

  #w_maxWords=open("maxWords", "a")
  #w_maxWords.write(my_line.strip()+", "+maxString+"\n")
  #w_maxWords.close()

  morph(my_list, maxString, maxNumber)

#works by list of synonym words if yes, then call findMax function
def base_func(word):

        #finding root of words
        os.chdir("/home/apertium/apertium-kaz")

        s_file=open("source_file.txt", "w")
        s_file.write(word)
        s_file.close()
        os.system("python translate.py")
        os.system("cat translated.txt|apertium -d. kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'|
cut -f2 -d'/'|cut -f1 -d '<'>root.txt")
        r_root=open("root.txt", "r")
        #finding translate to kazakh
        kazakh_word=r_root.readline().strip().lower()
        r_root.close()

        #finds which table we need
        count, my_line=findTable(kazakh_word)

        print ("English word is "+word)
```

```
            print ("Kazakh word is "+kazakh_word)

            if(count==0):
                    print("There is no any table of word: "+kazakh_word)
                    badWords.append(word)

            else:
                    func_count("table"+str(count), my_line, kazakh_word)

def roundToTwoDecimalPlace(num):
  return round(num, 2)

#================ Write To File ==============
def writeToFile(file, source):
  source_file=open(file, "w")
  source_file.write(source)
  source_file.close()

  #================ Append To File ==============
def appendToFile(file, source):
  source_file=open(file, "a")
  source_file.write(source)
  source_file.close()

#================ Write To File ==============
def checkSentenceSeparately(sentence, length):
  if(len(sentence)<length):
          return []
  words = sentence.split()
  groupedWords = [ ]
  for i in range(0, len(words), 1):
          newWord = ' '.join(words[i:i+length])
          lengthOfNewWords = len(newWord.split(' '))
          if(lengthOfNewWords >= length):
                  groupedWords.append(newWord)
  return groupedWords
def combineRootsWithSections(sections, roots):
  lastIndex = 0
  tempList = list()
  for section in sections:
          for line in section:
                  print(line)
                  wordLength = len(line.strip().split())
```

```
                tempSection = ' '.join(roots[lastIndex: lastIndex+wordLength])
                tempList.append(tempSection)
  return tempList
def countWordsInSections(wordsList, mainSection):
  res = dict()
  for mainWord in mainSection:
        genCount = 0
        for words in wordsList:

                for word in words:
                        count = 0
                        splittedLine = word.split()
                        count+=splittedLine.count(mainWord)*len(splittedLine)
                        if(count!=0):
                                genCount=genCount+(1.0/count)
        res[mainWord]= roundToTwoDecimalPlace(genCount)
  return res

#================ Write To File ==============
def checkSentenceSeparately(sentence, length):
  if(len(sentence)<length):
        return []
  words = sentence.split()
  groupedWords = [ ]
  for i in range(0, len(words), 1):
        newWord = ' '.join(words[i:i+length])
        lengthOfNewWords = len(newWord.split(' '))
        if(lengthOfNewWords >= length):
                groupedWords.append(newWord)
  return groupedWords

def combineRootsWithSections(sections, roots):
  lastIndex = 0
  tempList = list()

  for section in sections:
        for line in section:
                wordLength = len(line.strip().split())
                tempSection = ' '.join(roots[lastIndex : lastIndex+wordLength])
                tempList.append(tempSection)
                lastIndex += wordLength
  return tempList
```

```
def countWordsInSections(wordsList, mainMorphsList):
  # source_file=open("response.txt", "w")
  res = dict()

  for mainWord in mainMorphsList:
        genLengthList = list()
        for i in range(len(wordsList)):
                word = wordsList[i]
                temp = word.split()
                count = temp.count(mainWord)
                genLengthList.append(count*len(temp))
                #       source_file.write(mainWord+"       "+str(word)+":
"+str(genLengthList)+"\n")
        res[mainWord]= genLengthList
  # source_file.write(str(res))
  # source_file.close()
  return res


def confusionMatrixOfWords(kazakhWords, englishWords, mainValuesMorphKaz,
mainValuesMorphEng):
  kazakhWordsDict = countWordsInSections(kazakhWords, mainValuesMorphKaz)
  englishWordsDict = countWordsInSections(englishWords, mainValuesMorphEng)

  combinedAllDict = dict()
  for keyEn, listEn in englishWordsDict.items():
        for keyKz, listKz in kazakhWordsDict.items():
            count = 0
            for i in range(len(listEn)):
                if(i<len(listKz)):
                    mltply = listEn[i]*listKz[i]
                    if(mltply>0):
                        count += 1/float(mltply)
            combinedAllDict[keyKz+"  -  "+keyEn]  =
roundToTwoDecimalPlace(count)
  return combinedAllDict

def secondFormula(mainKazakhRootwords, translatedEnMainMorphWords,
mainEnglishRootwords, confusionMatrixResult):

  confusionMatrixNewDict = {}
  diffWords = []
  sameWords = []
```

```
  for i in range(len(translatedEnMainMorphWords)):

        enRoot = translatedEnMainMorphWords[i].lower()
        if("*" in enRoot):
            enRoot = enRoot.replace("*", "")
        if("$" in enRoot):
            enRoot = enRoot.replace("$", "")
        if(i< len(mainEnglishRootwords) and enRoot == mainEnglishRootwords[i].lower()):
            sameWords.append(enRoot)

  for i in range(len(mainKazakhRootwords)):
        kzRoot = mainKazakhRootwords[i]
        numerator = 0
        denomenator = 0
        result = 0
        for key, value in confusionMatrixResult.items():
            if(kzRoot in key):
                denomenator +=value
                if(key[str(key).rfind(" "):].strip()  in sameWords):
                    numerator+=value
                if(denomenator > 0):
                    result =roundToTwoDecimalPlace(numerator/float(denomenator))
                    diff = mainEnglishRootwords[i]
                    if(result <= 0.5 and diff not in diffWords):
                        diffWords.append(diff)
        confusionMatrixNewDict[mainEnglishRootwords[i]] = result
  if(len(diffWords)>0):
        print("Different words are "+str(diffWords))
  writeToFile("response.txt", str(confusionMatrixNewDict))
  return diffWords


#===========read from terminal and find root of words from google translator and apertium===========
inputArgument = sys.argv[1:]
inputString = ' '.join(inputArgument)


if(len(inputString)==0):
  print ("Enter correct sentence!!!")
else:
```

```
  #if('I' in inputString):
        #inputString = inputString.replace("I", "i")
  writeToFile("source_file.txt",inputString )

  os.system("cat source_file.txt|apertium -d. eng-kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'|
cut -f2 -d'/'|cut -f1 -d'<' > english_roots.txt")
  with open("english_roots.txt") as file:
        mainEnglishRootwords = [line.rstrip() for line in file]

  if('I' in inputString):
        inputString = inputString.replace("I", "i")
  main_text=inputString

  writeToFile("source_file.txt",inputString )
  os.system("python translate.py")
  #writeToFile("source_file.txt", mainKazakhSentence)
  #os.system("python translate.py")


  os.system("cat translated.txt|apertium -d. kaz-eng-morph|sed -e 's/\$\W*\^/$\\n^/g'|
cut -f2 -d'/'|cut -f1 -d'<'>kazakh_roots.txt")

  with open("kazakh_roots.txt") as file:
        mainKazakhRootwords = [line.rstrip() for line in file]
  translatedKazakhFIle = open("translated.txt", "r")
  mainKazakhSentence = translatedKazakhFIle.readline()

  writeToFile("source_file.txt", mainKazakhSentence)
  os.system("python translate.py")


  os.system("python reverse.py")
  translatedEnMainSentenceFile = open("translatedToEnglish.txt", "r")
  translatedEnMainSentence = translatedEnMainSentenceFile.readline()
  os.system("cat translatedToEnglish.txt|apertium -d. kaz-eng-morph|sed -e
's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d'<'>english_roots.txt")
  with open("english_roots.txt") as file:
        translatedEnMainMorphWords = [line.rstrip() for line in file]


  #========== separate with grouped words of sentence ==========
  writeToFile("source_file.txt", "")
  englishGroupedSections = list()
```

```
  kazakhRootWords = list()
  kazakhMorphWords = list()
  englishMorphWords = list()
  # lengthList = list()
  badWords = []
  for i in range(1, 4):
        groupedWords = checkSentenceSeparately(inputString, i)
        # lengthList.extend([i for y in range(len(groupedWords))])
        toStringGroupedWords = ' \n'.join(groupedWords)
        englishGroupedSections.append(groupedWords)

        #========== Get translation and morph of grouped words ==========
        appendToFile("source_file.txt", toStringGroupedWords+"\n")
  os.system("python translate.py")



  os.system("cat translated.txt|apertium -d. kaz-eng-morph|sed -e 's/\$\W*\^/$\\n^/g'|
cut -f2 -d'/'|cut -f1 -d'<'>kazakh_roots.txt")
  os.system("cat source_file.txt|apertium -d. eng-kaz-morph|sed -e 's/\$\W*\^/$\\n^/g'|
cut -f2 -d'/'|cut -f1 -d'<' > english_roots.txt")



  with open("kazakh_roots.txt") as file:
        kazakhRootWords = [line.rstrip() for line in file]

  with open("english_roots.txt") as file:
        tempList = [line.rstrip() for line in file]
        englishMorphWords = combineRootsWithSections(englishGroupedSections,
tempList)



  with open("translated.txt") as file:
        tempList = list()
        for line in file:
              tempList.append([line.rstrip()])
        kazakhMorphWords = combineRootsWithSections(tempList,
kazakhRootWords)

  confusionMatrixResult = confusionMatrixOfWords(kazakhMorphWords,
englishMorphWords, mainKazakhRootwords, mainEnglishRootwords)
  diffWords = secondFormula(mainKazakhRootwords,
translatedEnMainMorphWords,mainEnglishRootwords, confusionMatrixResult)
```

```
  if(len(diffWords)==0):
                print("There is no any bad words!")
  else:

                for i in diffWords:
                        '''
                        s_file=open("source_file.txt", "w")
                        s_file.write(i)
                        s_file.close()
                        os.system("cat source_file.txt|apertium -d. eng-kaz-morph|sed -e
's/\$\W*\^/$\\n^/g'| cut -f2 -d'/'|cut -f1 -d '<'>root.txt")
                        root_file = open("root.txt")
                        root_word=root_file.readline().strip().lower()
                        root_file.close()
                        '''
                        base_func(i)

  articles=["a", "an", "the"]
  for article in articles:
        if(article in badWords):
                badWords.remove(article)
  if(len(badWords)>0):
        print("Go to train1.py =====>> ...")
        os.chdir("/home/apertium/apertium-eng-kaz")
        for badword in badWords:
                os.system("python train1.py "+badword)
```