BIG DATA TEXNOLOGIYALARIDA N-GRAM MODELLAR, INDEKSLASH VA REAL VAQTLI TAHLIL USULLARINING QO‘LLANILISHI

Primova Mastura Hakim qizi,
Katta o‘qituvchi
primovamastura@navoiy-uni.uz
ToshDO‘TAU

Annotatsiya. Ushbu maqolada BigData muhitida keng qo‘llaniladigan zamonaviy texnologiyalar – N-gram modellar, Elasticsearch asosidagi matnni indekslash, Apache Kafka yordamida real vaqtli matn oqimlarini qayta ishlash, NoSQL ma’lumotlar bazalarida o‘zbek matnli lug‘atlarni saqlash hamda bulut muhitida matn tahlili xizmatlarining o‘rni va amaliy ahamiyati yoritilgan. Mazkur texnologiyalar katta hajmdagi matnli ma’lumotlarni tezkor qayta ishlash, qidirish, tahlil qilish va real vaqt rejimida qaror qabul qilish jarayonlarini optimallashtirishda muhim rol o‘ynaydi. Maqolada har bir texnologiyaning ishlash printsipi va ularning BigData ekotizimidagi o‘zaro bog‘liqligi ilmiy jihatdan tahlil qilingan.

Kalit so‘zlar: *BigData, N-gram, Apache Kafka, Elasticsearch, NoSQL.*

Abstract. This article discusses modern technologies widely used in the BigData environment, including N-gram models, text indexing based on Elasticsearch, real-time text stream processing using Apache Kafka, storage of Uzbek text dictionaries in NoSQL databases, and the role and practical significance of cloud-based text analysis services. These technologies play an important role in optimizing the processes of fast processing, searching, analyzing, and making real-time decisions on large-scale textual data. The article provides a scientific analysis of the working principles of each technology and their interconnection within the Big Data ecosystem.

Keywords: *BigData, N-gram, Apache Kafka, Elasticsearch, NoSQL.*

Ushbu maqolada BigData muhitida keng qo‘llaniladigan zamonaviy texnologiyalar – N-gram modellar, Elasticsearch asosidagi matnni indekslash,

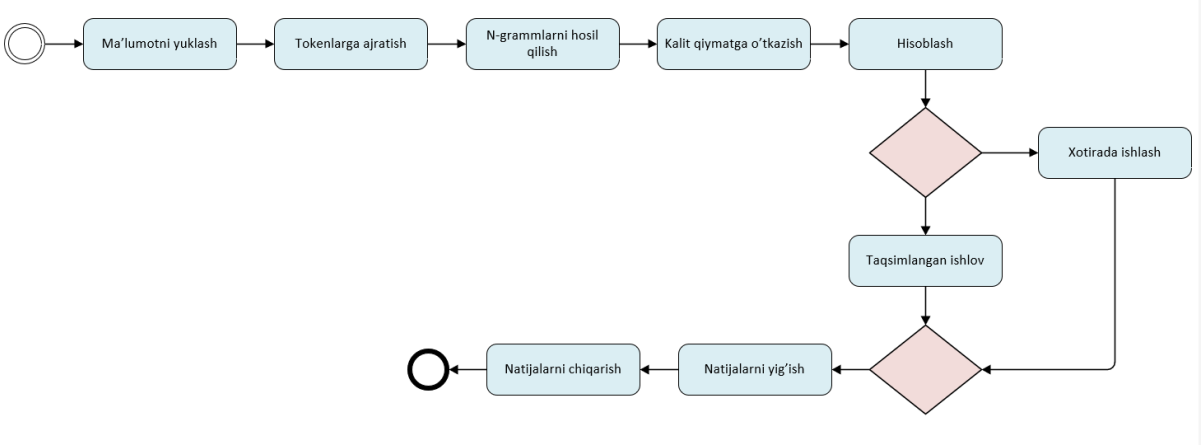
Apache Kafka yordamida real vaqtli matn oqimlarini qayta ishlash, NoSQL ma’lumotlar bazalarida o‘zbek matnli lug‘atlarni saqlash hamda bulut muhitida matn tahlili xizmatlarining o‘rni va amaliy ahamiyati yoritilgan. Mazkur texnologiyalar katta hajmdagi matnli ma’lumotlarni tezkor qayta ishlash, qidirish, tahlil qilish va real vaqt rejimida qaror qabul qilish jarayonlarini optimallashtirishda muhim rol o‘ynaydi.

Yuqorida nazariy jihatdan ko‘rib chiqilgan texnologiyalar amaliyotda turli kombinatsiyalarda qo‘llanilib, matnlarni qayta ishlash bo‘yicha keng ko‘lamli loyihalarni amalga oshirishga imkon yaratmoqda. Quyida har bir texnologiyani matnli ma’lumotlar bilan bog‘liq aniq vazifalarda qo‘llashga doir misollar keltiriladi.

1. **Apache Spark va N-gram modellar.** Katta hajmdagi matnlardan N-gramlarni ajratib chiqarish va ularning chastotasini hisoblash – til modellari qurishda muhim bosqich[1]. Masalan, o‘zbek tilida keng korpusdan ikki so‘zli birikmalar (bigrammalar) statistikasi kerak bo‘lsa, Spark’da buni juda tez bajariladi. Dastlab, matnlar RDD yoki DataFrame’ga yuklanadi, so‘ngra Spark API yordamida har bir gap tokenlarga ajratiladi va flatMap operatsiyasi bilan n-gramlar generatsiya qilinadi. Keyin reduceByKey orqali bir xil n-gramlar soni jamlanadi. Spark’ning in-memory ishlashi va tarqatilgan hisoblash tufayli millionlab hujjatlardagi n-gramlar bir necha daqiqada hisoblanishi mumkin. Amaliy misol sifatida, Spark NLP kutubxonasida maxsus NGramGenerator transformatori mavjud bo‘lib, u matnli ustundan n-gramlarni hosil qilib beradi. Masalan, ushbu vosita yordamida o‘zbek tilidagi matnlardan trigram (3-li birikmalar) jadvalini olish uchun bir necha qatorda kod yozish kifoya bo‘ladi – kutubxona ostida Spark klasteri orqali bu hisob-kitobni taqsimlab bajaradi. Google kompaniyasining mashhur **N-Gram Corpus (Google Books Ngrams)** ham aslida Hadoop MapReduce yordamida kitoblar korpusidan

chiqarib olingan bo‘lib, unda milliardlab N-gramlar mavjud – bu hajmdagi ishni faqat BigData texnologiyalari bilan ado etish mumkinligi yaqqol misoldir[2].

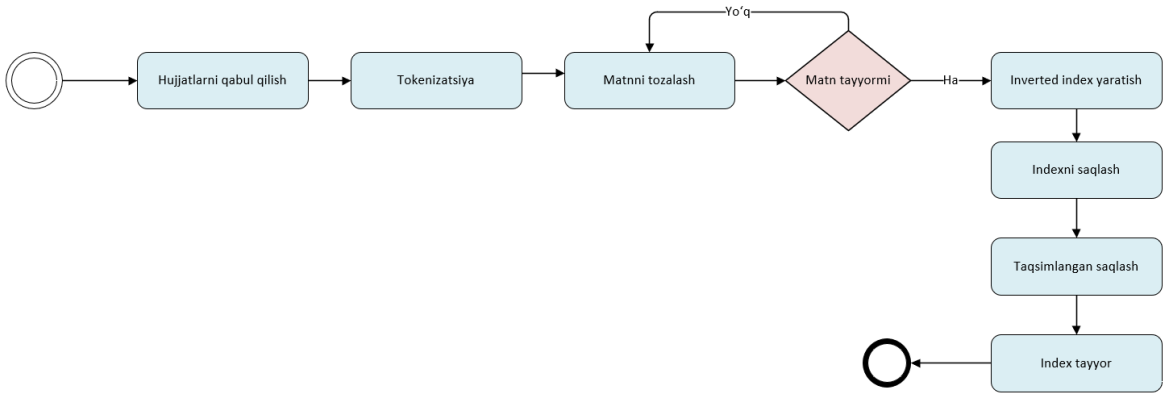
Quyidagi 1-rasmda matnni qayta ishlash jarayoni blok-sxemasi keltirilgan.



1-rasm. Matnni qayta ishlash jarayoni blok-sxemasi

2. **Elasticsearch va matnni indekslash.** Matnli axborotni foydalanuvchi uchun qulay qidirish imkoniyatini yaratish uchun Elasticsearch kabi qidiruv motorlari qo‘llanadi[3]. Masalan, **O‘zbek tilining elektron korpusini** (bir necha yuz million so‘zdan iborat matnlar to‘plamini) qidiruvchan qilish vazifasini olaylik. An’anaviy yondashuvda bunday qidiruv juda sekin bo‘lar edi (har safar butun korpusni ko‘zdan kechirish talab qilinadi). Elasticsearch esa indekslash jarayonida har bir hujjatni tahlil qilib, invertlangan indeksga yozib boradi: so‘zlar asosiy shaklga keltiriladi, masalan, *“kutubxonalarimizdan”* → *“kutubxona”,* so‘ngra ushbu lemma hujjat IDsi bilan indeksga joylanadi. Natijada qidiruv vaqtida foydalanuvchi *“kutubxona”* so‘zini kiritganda, tizim indeks orqali darhol ushbu so‘z qaysi hujjatlarda ekanligini topadi (bu amaliyot matnni oldindan tokenizatsiya va stemming qilish evaziga juda tezkor bo‘ladi). Ko‘pgina veb-qidiruv tizimlari va sayt ichidagi qidiruv funksiyalari xuddi shu prinsipada ishlaydi. O‘zbek tilida ham Elasticsearch’ga o‘zbek tili uchun morfologik analiz pluginlari ulangan holda foydalanish mumkin – bu holda qidiruv tizimi so‘zning turli grammatik shakllarini bitta kalit ostida birlashtirib qidirishni yanada aqlliroq qiladi. Amaliy misol: masalan, Wikipedia O‘zbekcha bo‘limi uchun qidiruv Elasticsearch asosida ishlaydi,

unda foydalanuvchi qidiruvga so‘zning istalgan grammatik shaklini kiritsa ham, tegishli maqola topiladi (bunda Lucene asosidagi tahlilchilar ishlaydi). Bundan tashqari, Elasticsearch klasteri katta yuk ostida ham barqaror ishlay oladi – bir vaqtning o‘zida minglab qidiruv so‘rovlarini ko‘rib chiqib, ularga mos darajasida javob qaytaradi. Bu, albatta, uning BigData texnologiyalariga xos taqsimlangan ishlash va parallel qayta ishlash xususiyati samarasidir. Quyidagi 2-rasmda “Matnli hujjatlarni qayta ishlash va teskari indeks yaratish jarayoni blok-sxemasi” keltirilgan bo‘lib, unda hujjatlarni qabul qilishdan boshlab tokenizatsiya, matnni tozalash va yakuniy bosqichda teskari indeksni yaratish jarayonini bosqichma-bosqich ifodalaydi.



2-rasm. Matnli hujjatlarni qayta ishlash va teskari indeks yaratish jarayoni blok-sxemasi

3. **Apache Kafka va real vaqtli matn oqimlari.** Kafka’dan foydalangan holda real vaqt rejimida keluvchi matn ma’lumotlarini qayta ishlash juda keng tarqalgan amaliyotdir. Masalan, ijtimoiy tarmoqlardagi postlarni tahlil qilish yoki chatbotlar uchun foydalanuvchi xabarlarini oqimda qayta ishlashda Kafka asosiy komponent bo‘la oladi. Bir real loyiha misoli: Twitter API orqali ma’lum kalit so‘zlar bo‘yicha tweetlarni olib, ular ustida real-time sentiment analysis bajarish. Bunda arxitektura quyidagicha tuziladi: Python’da yozilgan dastur (producer) Twitter’dan kelgan har bir yangi tweet matnini Kafka topig‘iga yuboradi. Kafka bu xabarlarni navbatda saqlab, Spark Structured Streaming dasturiga uzatadi. Spark

oqim dasturi har bir tweet matnini olishi bilan, unda tasniflash modelini (masalan, oldindan o‘qitilgan neyron tarmoq) qo‘llab, shu tweetning sentimentini (ijobiy, salbiy, neytral) aniqlaydi. Olingan natija (matn va uning sentiment belgisi) yana Kafka’ning boshqa bir topig‘iga yoki bevosita biror NoSQL bazaga yoziladi. Yakunda, masalan, MongoDBda jamlangan ushbu tahlil natijalarini veb-interfeys orqali kuzatish mumkin – unda foydalanuvchi real vaqt rejimida, oqim shaklida yangilanib boruvchi sentiment statistikasini ko‘radi. Bu misolda Kafka asenkron va decoupled (bog‘lanmagan) arxitekturani ta’minlab bermoqda: ya’ni, ma’lumot manbai (Twitter) bilan uning iste’molchilari (Spark, yoki ba’zan bir nechta mustaqil tahlil servislari) Kafka orqali ayri bog‘lanadi. Oqimdagi ma’lumotlarni bunday uzatish va tarqatish faqat tarqatilgan tizim yordamida erishish mumkin bo‘lgan ishonchlilik va masshtablanish xususiyatini talab etadi – Kafka shu bois deyarli barcha yirik real vaqt analitika loyihalarining ajralmas qismiga aylangan[4-5].

4. **NoSQL va o‘zbek matnli lug‘atlar.** Katta hajmdagi lingvistik resurslarni saqlash uchun NoSQL bazalari qulay yechim bo‘la oladi. Masalan, o‘zbek tilining elektron lug‘atini (bir necha yuz ming leksema va ularning barcha shakllari bilan) yaratishda MongoDB hujjat bazasi ishlatilgan deylik. Har bir hujjat bitta so‘z va unga oid ma’lumotlarni (asosiy ma’no, sinonimlar, tarjimalar, qo‘shimcha ma’lumotlar) JSON shaklida saqlaydi. Bunday bazada biror so‘z bo‘yicha qidiruv juda tez amalga oshadi (indekslar tufayli), qolaversa, yangi so‘zlarni qo‘shish yoki ma’lumotni o‘zgartirish ham oddiy JSON operatsiyasi bilan bajariladi. Cassandra kabi tizimlar esa, masalan, katta korpusdagi N-gram chastotalar jadvalini saqlash uchun ishlatilishi mumkin: jadvalning kaliti – N-gramning o‘zi, ustunlarda esa matn to‘plamlari bo‘yicha uchrashuv sonlari. Cassandra millionlab qatorlardan iborat bunday jadvalni tarqatilgan holda ushlab, minglab so‘rovlarni bir sekundda amalga oshira oladi. Bu esa masalan, interaktiv lug‘at ilovasida foydalanuvchi kiritayotgan so‘zning tezkor autokomplit (auto-

complete) takliflarini shakllantirishga yordam beradi – tizim bir nechta harf bo‘yicha boshlanuvchi eng tez-tez uchraydigan so‘zlarni N-gram jadvalidan darhol chiqarib beradi. NoSQL bazalarning bunday masshtablanuvchanligi tufayli, o‘zbek tiliga oid katta hajmdagi ma’lumotlar (lug‘atlar, tez-tez uchraydigan so‘zlar ro‘yxati, kollokatsiyalar bazasi) samarali boshqarilishi va xizmat qilishi mumkin.

5. **Bulut muhitida matn tahlili xizmatlari.** Bugungi kunda amaliy loyihalarda ko‘pincha Big Data texnologiyalari bulut infratuzilmasida joriy etiladi[6]. Masalan, O‘zbek tilida nutqni matnga aylantirish yoki matndan nutq sintetizatsiyasi kabi servislarni olaylik – ularning ortida katta hajmdagi ma’lumotlarda o‘qitilgan modellari turadi va foydalanuvchilarga real vaqt rejimida xizmat ko‘rsatadi. AWS, Google Cloud yoki Azure kabi platformalarda bunday xizmatlarni tayyor API ko‘rinishida olish mumkin (masalan, Google Cloud Speech-to-Text O‘zbek tilini ham qo‘llab-quvvatlaydi). Agar esa o‘zimiz mustaqil tarzda biron matnni qayta ishlash pipeline’ini qurmoqchi bo‘lsak, bulutda buning uchun kerakli barcha qurilmalar mavjud: AWS Glue orqali matnlarni ETL (extract-transform-load) qilish, Amazon EMRda Spark klasterini ko‘tarib morfologik tahlilni tarqatilgan holda bajarish, Amazon Elasticsearch Service (OpenSearch) orqali natijalarni qidiruvchan qilish va API Gateway + Lambda orqali foydalanuvchiga HTTP API ko‘rinishida ulash. Xuddi shunday konstruktsiyani Azure yoki GCPda ham tuzish mumkin. Bulutlarning afzalligi – dastlab kichik masshtabda boshlab, foydalanuvchilar ko‘paygan sari avtomatik ravishda klaster quvvatini oshirish (auto-scaling) va xizmatni uzluksiz ravishda boshqarish[7]. O‘zbekistonda ham so‘nggi vaqtlarda bulut xizmatlari joriy etila boshlagan; masalan, ba’zi mahalliy IT kompaniyalar matnni tahlil qilish uchun Yandex Cloud yoki Oracle Cloud’dan foydalanmoqda. Bulut infratuzilmasi Big Data va NLP tutashgan nuqtada tez va samarali yechim yaratish vaqtini qisqartiradi – developerlarga tayyor platformada faqat o‘zining algoritmlarini yo‘lga qo‘yish va natijani olish qoladi xolos.

Xulosa. Ushbu maqolada, zamonaviy BigData texnologiyalari va vositalari yordami bilan o‘zbek tilidagi ulkan matnlar ustida ilgari bajarilishi qiyin bo‘lgan vazifalar ham amalga oshirilmoqda. Katta hajmdagi korpuslardan til modellari o‘qitish, real vaqt chatlardan ma’noli ma’lumot ajratish, matnlarni semantik indekslash va qidiruv tizimlarini yaratish – bularning barchasi endilikda Hadoop, Spark, NoSQL, Kafka, Elasticsearch singari platformalar integratsiyasi orqali hayotga tatbiq etilmoqda. O‘zbek tili lug‘aviy boy til bo‘lib, uni kompyuterda *“tushunish”* uchun ko‘p ma’lumot va kuchli hisoblash zarur.

Foydalanilgan adabiyotlar ro‘yxati

1. Leskovec J., Rajaraman, A., Ullman,J. D. Mining of Massive Datasets. Cambridge University Press, 2020.
2. White T. Hadoop: The Definitive Guide. O’Reilly Media, 2015.
3. Elasticsearch B.V. Elasticsearch Guide. https://www.elastic.co/guide/
4. Zaharia M. et al. Apache Spark: A Unified Engine for Big Data Processing. Communications of the ACM, 2016.
5. Apache Software Foundation. Apache Kafka Documentation. https://kafka.apache.org/
6. Google Cloud. Natural Language API Documentation. https://cloud.google.com/natural-language
7. Amazon Web Services. Amazon Comprehend Developer Guide. https://docs.aws.amazon.com/comprehend/