SUN’IY INTELLEKT KUTUBXONALARI YOQDAMIDA MATNLARNI QAYTA ISHLASH

Berdiyev Jahongir Botir o‘g‘li,
Kompyuter lingvistikasi mutaxassisligi magistranti
berdiyevjahongir94@gmail.com
ToshDO‘TAU

Annotatsiya. Zamonaviy texnologiyalar taraqqiyoti sun’iy intellekt (SI) sohasini jadal rivojlantirmoqda. Ayniqsa, tabiiy tilni qayta ishlash (Natural Language Processing — NLP) yo‘nalishida erishilayotgan yutuqlar turli sohalarda matn bilan ishlash jarayonini avtomatlashtirish va optimallashtirish imkonini bermoqda. Ilgari faqat inson aqli orqali bajarilgan matn tahlili, his-tuyg‘ularni aniqlash, tarjima, xulosa chiqarish kabi vazifalar bugungi kunda sun’iy intellekt yordamida yuqori aniqlikda amalga oshirilmoqda. Matn bilan ishlashda asosiy vazifa uni kompyuter tushunadigan shaklga keltirishdir. Buning uchun maxsus algoritmlar va modellardan foydalaniladi. Shu jarayonda TensorFlow, PyTorch, Scikit-learn kabi SI kutubxonalari muhim o‘rin egallaydi. Ushbu kutubxonalar yordamida neyron tarmoqlarni o‘rgatish, mavjud modellarni sozlash va yangi yondashuvlarni sinovdan o‘tkazish mumkin.

Ushbu maqolada matnlarni qayta ishlash jarayonida sun’iy intellekt kutubxonalarining tutgan o‘rni, ularning imkoniyatlari va amaliy qo‘llanilishi haqida so‘z yuritiladi. Shuningdek, yetakchi tadqiqotchilar ishlanmalari asosida iqtiboslar keltiriladi va turli kutubxonalar taqqoslanadi.

Abstract. The rapid advancement of modern technologies has significantly accelerated the development of artificial intelligence (AI). In particular, the field of Natural Language Processing (NLP) has witnessed substantial progress, enabling the automation and optimization of various text analysis processes. Tasks such as sentiment detection, text classification, summarization, and machine translation, which were once performed solely by human experts, are now being carried out by intelligent systems with high accuracy. One of the core challenges in text processing is converting natural language into a machine-readable format. To achieve this, various algorithms and deep learning models are employed. In this context, open-source AI libraries such as TensorFlow, PyTorch, Scikit-learn play a crucial role. These libraries provide efficient tools for training neural networks, fine-tuning pre-trained models, and experimenting with innovative approaches.

This article explores the role and capabilities of AI libraries in text processing tasks. It further discusses practical applications and comparative analysis of leading frameworks, referencing key contributions from prominent researchers in the field.

Аннотация. Стремительное развитие современных технологий существенно ускорило прогресс в области искусственного интеллекта (ИИ). Особенно заметны достижения в сфере обработки естественного языка (Natural Language Processing — NLP), что позволило автоматизировать и оптимизировать различные задачи анализа текстов. Такие операции, как определение тональности, классификация текста, реферирование и машинный перевод, ранее выполнявшиеся исключительно человеком, сегодня реализуются интеллектуальными системами с высокой точностью. Одной из основных задач при работе с текстами является преобразование естественного языка в формат, понятный для машин. Для этого используются специализированные алгоритмы и модели глубокого обучения. В этом контексте важную роль играют открытые библиотеки ИИ, такие как TensorFlow, PyTorch, Scikit-learn. Эти инструменты позволяют эффективно обучать нейронные сети, адаптировать уже готовые модели и тестировать новые подходы.

В данной статье рассматриваются роль и возможности библиотек искусственного интеллекта при обработке текстов. Также проводится сравнительный анализ популярных инструментов с опорой на ключевые научные исследования и авторитетные источники.

Kalit so‘zlar. *TensorFlow, PyTorch, Scikit-learn, tokenizatsiya, vektorlashtirish, klasslar.*

Kirish. Matnni qayta ishlash sun’iy intellektning tabiiy tilni qayta ishlash (NLP) sohasiga kiradi. Uning asosiy maqsadi mashinalarga inson tilini tushunish, talqin qilish va yaratish imkoniyatini berishdir. Chuqur o‘rganishning rivojlanishi bilan matnga asoslangan vazifalarni avtomatlashtirish va takomillashtirish uchun sun’iy intellekt kutubxonalari muhim rol o‘ynamoqda. Bu tizim matnni tasniflash, hissiyotlarni tahlil qilish, mashina tarjimasi va boshqa ko‘plab NLP vazifalarini bajarish imkonini beradi.

Sun’iy intellekt kutubxonalari yordamida ma’lum bir vazifa bajarish va matnga ishlov berish bir qancha bosqichlarni o‘z ichiga oladi. Quyida bu bosqichlarni ko‘rib chiqamiz.

Tokenizatsiya va to‘ldirish(padding).

Tabiiy tilni qayta ishlash (NLP) texnologiyalarining rivojlanishi bilan matnni raqamli formatga aylantirish usullari muhim ahamiyat kasb etmoqda. “Tokenizatsiya” va “to‘ldirish” kabi jarayonlar NLP algoritmlarining samaradorligi va ishlash tezligini ta’minlovchi asosiy tarkibiy qismlardir. Matnni qayta ishlashning birinchi bosqichi xom matnni mashinani o‘qitish modeli tushunadigan shaklga aylantirishni o‘z ichiga oladi. Bu jarayon *tokenizatsiya* bilan boshlanadi, u matnni

tokenlar deb ataladigan kichikroq birliklarga, odatda, so‘zlar, bo‘g‘inlar yoki harflarga ajratadi.

Tokenizatsiya – bu matnni kichik bo‘laklarga ajratish jarayonidir. Bu kichik bo‘laklar **tokenlar** deb ataladi. Tokenlar, odatda, so‘zlar, bo‘g‘inlar yoki boshqa ma’lumotli segmentlar bo‘lishi mumkin. Tokenizatsiya tabiiy tilni qayta ishlashning birinchi va eng muhim bosqichlaridan biridir. Tokenizatsiya jarayoni matn ustida bajariladigan amallarga bog‘liq holda bir qancha turlarga bo‘linishi mumkin (Qarang: 1-jadval). Masalan, *bolalar bugun maktabga ketishdi* gapini turli xil tokenlarga ajratish mumkinligini quyidagi jadvalda ko‘rish mumkin:

1-jadval. Token turlari

Token turlari	**Token misoli**
Gap shaklida	Bolalar bugun maktabga ketishdi
So‘zshakl	Bolalar, bugun, maktabga, ketishdi
Asos va qo‘shimcha	Bola, lar, bugun, maktab, ga, ket, ish, di
Bo‘g‘in shaklida	Bo, la,lar, bu, gun, mak, tab, ga, ket, ish, di
Harf shaklida	B, o, l, a, l, a, r, b, u, g, u, n, m, a, k, t, a, b, g, a, k, e, t, i, sh, d, i

Yuqoridagi jadvaldan ko‘rinib turibdiki, matnlarni turli shaklda tokenizatsiya qilish mumkin ekan. Bu, albatta, matn ustida bajariladigan vazifaga bog‘liq bo‘ladi. Masalan, harf shaklidagi tokenizatsiya fonetik tahlil, yozuv qoidalari va shu kabi xususiyatlarni o‘rganishda ishlatiladi. Yoki so‘zshakl darajasidagi tokenizatsiya matn tasnifi mashina tajrimasi kabi vazifalar uchun ishlatiladi.

TensorFlow kutubxonasida tokenizatsiya modullari va ularning bir qancha klasslari mavjud. Quyida ularning bir qismini ko‘rib chiqamiz.

tf.keras.preprocessing.text moduli. Bu modul **TensorFlow** kutubxonasida matnlarni qayta ishlash uchun taqdim etilgan yordamchi funksiyalardan iborat. Tokenizer klassi matnni butun sonlar ketma-ketligiga aylantirish orqali matnni oldindan qayta ishlash uchun ishlatiladi, bu erda matndagi har bir so‘z yoki token ma’lumotlar to‘plamidagi chastotasi asosida noyob butun songa ko‘rsatiladi[1]. Bu modulda matnlarni **tokenizatsiya qilish, ketma-ketlikka aylantirish** va **NLP** modellar uchun tayyorlash jarayonini osonlashtirish uchun ko‘plab funksiyalar mavjud. Undagi “Tokenizer” klassi tokenizatsiya uchun keng qo‘llaniladi. “Tokenizer” matn korpusini butun sonlar ketma-ketligiga aylantiradi, bunda har bir butun son alohida so‘z yoki tokenni ifodalaydi. Masalan, “TensorFlow mashinali o‘qitishda mashhur kutubxona” degan oddiy jumla berilgan bo‘lsa, tokenizatsiya uni [“TensorFlow”, “mashinali”, “o‘qitishda”, “mashhur”, “kutubxona”] so‘zlariga bo‘ladi, so‘ngra ular korpusdagi chastotasiga qarab mos keladigan butun son

indekslariga aylantiriladi, ya’ni [[2, 3, 1, 4, 5]] shaklidagi natijani beradi. Bu modelga xom matn emas, balki raqamli tasvirlar bilan ishlash imkonini beradi. UnicodeCharTokenizer klassi matnni belgilarga bo‘lib tokenlashtiradi. Ya’ni matnni individual belgilar (harflar, raqamlar, maxsus belgilar) shaklida tokenlarga ajratadi. Bunda gap, so‘zlar yoki bo‘g‘inlar emas, balki har bir belgi alohida token sifatida ishlatiladi. WhitespaceTokenizer klassi bo‘sh joylarni ajratuvchi element sifatida ko‘rib, har bir bo‘sh joydan keyin yangi token yaratadi. Bu tokenizatsiya turi, asosan, so‘zlar orasidagi bo‘sh joylar bo‘yicha ajratish uchun ishlatiladi.

torchtext moduli. Bu modul PyTorch kutubxonasida matnlarni qayta ishlash uchun mo‘ljallangan maxsus vositalarni o‘z ichiga oladi. torchtext yordamida matnlarni tokenizatsiya qilish, lug‘at (vocabulary) yaratish, embedding (semantik vektorlar) hosil qilish va matnlar bilan ishlovchi datasetlar yaratish imkoniyati mavjud. torchtextdan foydalanib, siz matnli ma’lumotlarni qayta ishlash jarayonini avtomatlashtirishingiz mumkin. Bu jarayon tokenizatsiya, so‘zlarni raqamlarga aylantirish, lug‘at yaratish va hokazolarni o‘z ichiga oladi. Bu xususiyat bu amalni bajarish uchun murakkab kodlarni yozish shart emasligini bildiradi[2]. Bu moduldagi get_tokenizer() funksiyasi yordamida ingliz tilida keng qo‘llaniladigan tokenizatorlar, jumladan basic_english, spacy, moses va boshqa variantlar bilan matnni bo‘lish mumkin. Vocab klassi esa barcha tokenlarni indekslash va ularning embedding ifodalarini saqlash uchun ishlatiladi. Masalan, "mashina o‘rganish sohasida PyTorch juda samarali" degan jumla get_tokenizer() yordamida [“mashina”, “o‘rganish”, “sohasida”, “PyTorch”, “juda”, “samarali”] shaklida tokenlarga ajratiladi, so‘ngra ular Vocab yordamida indekslanadi va nn.Embedding orqali neyron tarmoqlar uchun tayyorlanadi. torchtext shuningdek, maxsus TextClassificationDataset, DataLoader va BucketIterator funksiyalarini taqdim etadi, bu esa yirik matnlar ustida o‘qitish jarayonini samarali tashkil etish imkonini beradi. PyTorch tokenizatsiyani fleksib tarzda sozlash imkonini berishi bilan boshqa kutubxonalardan ajralib turadi, ayniqsa LSTM, GRU, Transformer modellar bilan ishlaganda bu juda muhimdir.

scikit-learn.feature_extraction.text moduli. SciKit Learn - bu mashinani o‘rganish loyihalari uchun keng kutubxona bo‘lib, bir nechta tasniflagich va tasniflash algoritmlari, o‘qitish va ko‘rsatkichlarni yig‘ish usullari va kiritilgan ma’lumotlarni oldindan qayta ishlash uchun mo‘ljallangan[3]. Bu modul Scikit-learn kutubxonasida matnlarni raqamlashtirish va oldindan ishlashga ixtisoslashgan komponentlarni o‘z ichiga oladi. Asosan, CountVectorizer va TfidfVectorizer klasslari matnni vektor ko‘rinishiga keltirishda keng qo‘llaniladi. CountVectorizer matndagi har bir so‘zni indekslab, ularning chastotasiga qarab vektor yaratadi. Masalan, "Scikit-learn matnli tahlil uchun foydali vosita" jumlasi quyidagicha vektorga aylanishi mumkin: [“scikit”, “learn”, “matnli”, “tahlil”, “uchun”, “foydali”, “vosita”] → [1, 1, 1, 1, 1, 1, 1]. TfidfVectorizer esa so‘z chastotasi va

umumiy korpusdagi kamyoblik darajasini hisobga olib, har bir so‘zga mos og‘irlik (weight) beradi. Bu metod ayniqsa soxta yangiliklarni aniqlash, ijtimoiy tarmoq postlarini tahlil qilish, yoki hujjatlarni kategoriyalarga ajratishda juda foydalidir. Scikit-learn kutubxonasi shuningdek, tayyorlangan vektorlarni MultinomialNB, SVM, LogisticRegression va boshqa klassifikatorlarga uzatish imkonini beradi. Bu modul matnlarni qayta ishlashni chuqur o‘rganish tarmoqlarisiz bajarishni xohlagan foydalanuvchilar uchun yengil va tezkor yechim taqdim etadi. Shuningdek, Pipeline va GridSearchCV kabi vositalar orqali butun ishlov jarayonini avtomatlashtirish mumkin.

tf.keras.preprocessing.sequence moduli TensorFlow Keras kutubxonasining bir qismi bo‘lib, ketma-ketlik ma’lumotlari bilan ishlash vositalarini taqdim etadi. U mashinani o‘qitish modellari uchun matnli yoki ketma-ket ma’lumotlarni tayyorlash uchun yordamchi dasturlarni, ayniqsa, tabiiy tilni qayta ishlash (NLP) vazifasini o‘z ichiga oladi. Bu moduldagi TimeseriesGenerator klassi ketma-ket ma’lumotlarni qayta ishlash uchun, xususan, prognozlash, anomaliyalarni aniqlash yoki tasniflash kabi davriy vazifalar uchun yordamchi dasturdir. Bu LSTM, GRU yoki CNN kabi modellarga vaqt seriyasidagi ma’lumotlarni uzatishni osonlashtirib, siljish oynalari ketma-ketligini va ularga mos keladigan maqsadlarni yaratishga yordam beradi. Ya’ni bu klass kutilayotgan natijani bashorat qilishga yordam beradi. NgramTokenizer klassi esa ketma-ket elementlardan tashkil topgan n-gramm tokenlarini yaratuvchi tokenizatorni chaqiradi. N-gramm tokeni ketma-ketlikdagi n ta elementdan tashkil topgan tokendir.

torch.nn.utils.rnn va **torchtext** modullari PyTorch kutubxonasida ketma-ketlik ma’lumotlari bilan ishlashda muhim rol o‘ynaydi. Har qanday chuqur o'rganish modeli torch.nn modulining quyi sinfi yordamida ishlab chiqilgan bo'lib, u chiqishni qaytaruvchi oldinga (kirish) kabi usuldan foydalanadi[4]. Bu modullar yordamida NLP va vaqt ketma-ketligi asosidagi vazifalar uchun matnli ma’lumotlarni samarali tayyorlash mumkin. torch.nn.utils.rnn.pad_sequence funksiyasi turli uzunlikdagi ketma-ketliklarni bitta batchga joylashtirish uchun ularni maksimal uzunlikka tenglashtirib (padding), tensor holatiga keltiradi. Bu esa LSTM yoki GRU kabi modellar uchun bir xil o‘lchamdagi kiritish (input) yaratishga imkon beradi. pack_padded_sequence va pad_packed_sequence funksiyalari esa modelga joylashtirilgan (packed) yoki asl shakliga qaytarilgan (unpacked) ma’lumotlar bilan ishlash imkonini beradi. Shuningdek, torchtext.data moduli ichida mavjud bo‘lgan BucketIterator va Field klasslari yordamida ma’lumotlar optimallashtirilgan tarzda modelga uzatiladi. Bu komponentlar ayniqsa, vaqt seriyalari, til modellari yoki ketma-ketlik asosida klassifikatsiya vazifalarida muhim ahamiyatga ega. torchtext.transforms.NGramTransform esa matndan n-gram tokenlar yaratish imkonini beradi, bu esa murakkab til tuzilmalarini modelga tushuntirishda foydalidir.

sklearn.feature_extraction.text va **sklearn.model_selection** modullari Scikit-learn kutubxonasida matn va vaqt ketma-ketligi asosidagi ma’lumotlarni qayta ishlash uchun foydalaniladi. CountVectorizer va TfidfVectorizer klasslari yordamida matnli ketma-ketliklarni vektorlashtirish, ya’ni raqamli ifodalarga aylantirish mumkin. Ayniqsa TfidfVectorizer N-gram parametrlarini belgilash orqali ketma-ketlikdagi tuzilmalarni (masalan, bigram yoki trigram) aniqlash imkonini beradi. Bu matnlardagi kontekstni chuqurroq tushunishga xizmat qiladi. Vaqt asosidagi vazifalarda TimeSeriesSplit klassi yordamida modelni o‘qitish va test qilish uchun vaqt tartibida bo‘lingan datasetlar yaratish mumkin. Bu ayniqsa prognozlash va vaqtga bog‘liq klassifikatsiya vazifalari uchun foydalidir. Scikit-learn modullari LSTM yoki GRU kabi chuqur tarmoqlarni to‘g‘ridan-to‘g‘ri taqdim etmasa-da, statistik asoslangan modellarga tayyorlangan ketma-ketliklarni berish orqali NLP vazifalarini muvaffaqiyatli bajarishga xizmat qiladi. Bu kutubxona o‘zining soddaligi va mustahkam poydevori bilan mashhur.

tf.keras.preprocessing.image moduli rasm ma’lumotlarini oldindan ishlash uchun ishlatiladi. U rasmlarni disk yoki veb saytlardan yuklash, uning o‘lchamlarini o‘zgartirish, normalizatsiya qilish va modelga tayyorlash uchun bir qator yordamchi funksiyalar va sinflarni keltiradi. ImageDataGenerator klassi, asosan, tasvirlarni oldindan ishlash va real vaqt rejimida ma’lumotlarni boyitish vazifasini bajaradi. Bu klass mashinani o‘rgatish jarayonida tasvirlar bilan ishlashni osonlashtiradi va modelning umumlashgan qobiliyatini oshirishga yordam beradi. Bu klassning asosiy vazifalari tasvirlarni neyron tarmoqqa kiritishdan oldin qayta ishlaydi, tasvirlarni kerakli o‘lchamlarga keltiradi, modelni turli xil sharoitlarga moslashtirish uchun real vaqt rejimida tasvirlarni o‘zgartiradi, ma’lumotlarni boyitadi (ya’ni aylantirish, kesish, o‘zgartirish, gorizontal/vertikal shaklda aylantirish va shu kabilar), katta hajmdagi tasvirlar bilan ishlash (diskdagi katta hajmdagi tasvirlarni yuklamasdan, real vaqt rejimida guruhlab yuklab o‘rganish imkonini beradi).

Biroq real dunyo matn ma’lumotlari uzunligi bo‘yicha farq qilishi mumkin va neyron tarmoqlar, odatda, belgilangan o‘lchamdagi kirish ma’lumotlarini kutishadi. Buni hal qilish uchun barcha ketma-ketliklarning bir xil uzunlikka ega bo‘lishini ta’minlash uchun *to‘ldirish*(padding) ishlatiladi. To‘ldirish juda muhim, chunki mashinani o‘qitish modellari, ayniqsa, neyron tarmoqlari, odatda, ma’lum hajmdagi kirishlarni talab qiladi. To‘ldirish turli uzunlikdagi ketma-ketliklarni bir xil uzunlikka keltirishni ta’minlaydi, bu esa partiyalarni qayta ishlashga imkon beradi. TensorFlow eng uzun ketma-ketlikning uzunligiga mos keladigan qisqaroq ketma-ketliklarni nol (yoki boshqa qiymatlar) bilan to‘ldirish uchun “pad_sequences” funksiyasini taqdim etadi. Bu partiyalarni qayta ishlash va samarali o‘qitish uchun juda muhimdir.

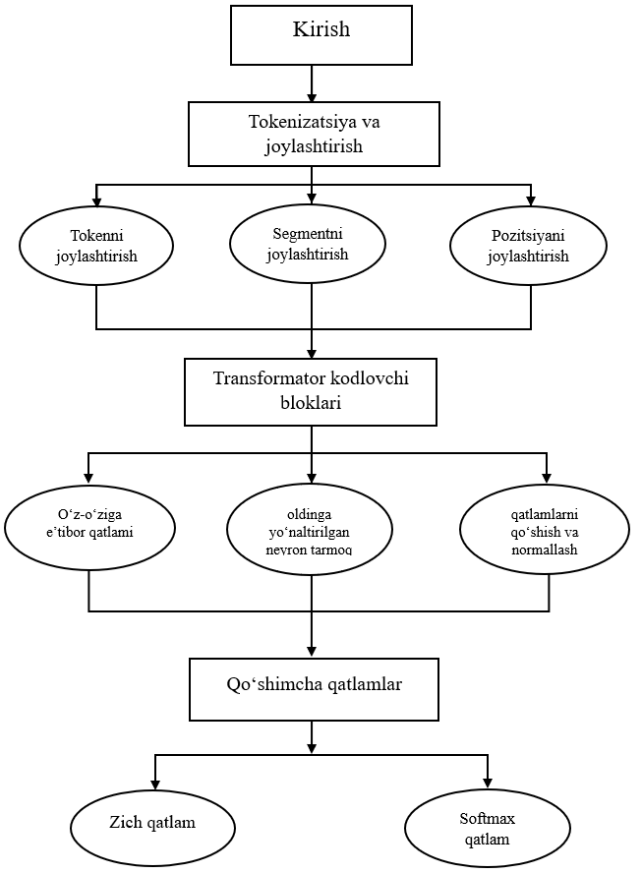
Matnni vektorlashtirish: Matnni raqamli tasvirlarga aylantirish.

Matn tokenizatsiya qilingandan so‘ng, keyingi qadam bu tokenlarni raqamli vektorlarga aylantirishdir. Ushbu transformatsiya matnning semantik ma’nosini saqlab qolgan holda neyron tarmoqqa matnli ma’lumotlar bo‘yicha hisob-kitoblarni amalga oshirish imkonini beradi. Keng tarqalgan yondashuvlardan biri *so‘zlarni joylashtirish* bo‘lib, ular har bir so‘zni uzluksiz vektor fazosida zich vektor sifatida ifodalaydi. Matnni vektorlashtirish juda muhim, chunki kompyuterlar faqat raqamlar bilan ishlaydi. Matnni raqamlarga aylantirish orqali, kompyuter modeliga matnni o‘rganish, uning xususiyatlarini aniqlash va tasniflash mumkin bo‘ladi. Bu jarayonni amalga oshirishda TensorFlow kuchli vosita hisoblanadi. TensorFlowda matnni vektorlashtirish uchun ko‘plab usullar mavjud bo‘lib, ulardan eng ko‘p ishlatiladigan usul TextVectorization qatlamidan foydalanishdir. Bu qatlam matnni sonli vektorlarga aylantiradi va matnni modelga kiritish uchun tayyorlaydi. Shuningdek, bu qatlam matnni tokenizatsiya qilish, vektorlashtirish, to‘ldirish va oldindan ishlov berish kabi vazifalarni amalga oshiradi. Bu esa kodni ixsham va soda qiladi. Bundan tashqari, matnni vektorlashtirishda so‘zlarning ma’nosini yaxshiroq ifodalash uchun *so‘z joylashtirish* funksiyasidan foydalaniladi. Joylashtirishlar har bir so‘zni yuqori o‘lchamdagi (masalan, 100 yoki 300 o‘lchamli) vektor bilan ifodalaydi. Bu usul so‘zlarning semantik ma’nosini saqlashga yordam beradi. Joylashtirish so‘zlar orasidagi semantik munosabatlarni qamrab oladi, masalan, o‘xshash ma’noga ega so‘zlar o‘xshash vektor ko‘rinishlariga ega bo‘ladi.

TensorFlow-da “tf.keras.layers”dagi “Embedding(Joylashtirish)” qatlami trening davomida ushbu joylashtirishlarni o‘rganish uchun ishlatiladi. Bu qatlam butun sonlar ketma-ketligini (tokenlashtirilgan so‘zlar) oladi va ularni ma’lum o‘lchamdagi zich vektorlar bilan taqqoslaydi. Joylashtirish qatlami, odatda, xom ma’lumotlarni, ya’ni so‘zlar, matnlar, tasvirlar yoki shunga o‘xshash elementlarni tahlil qilish va model uchun tushunarli shaklga o‘tkazish uchun xizmat qiladi. Joylashtirish qatlamlari juda katta hajma ega bo‘lgan xom ma’lumotlarning hajmini qisqartirib, modelga samarali tarzda ishlatish imkonini beradi. Misol uchun, model “shoir” so‘zini joylashtirish maydonida “yozuvchi” so‘ziga yaqin vektor sifatida ko‘rsatadi. Chunki bu so‘zlar bir-biriga o‘xshash konterkstlarda ishlatiladi.

Tokenizatsiya va vektorlashtirish jarayonlari tugagandan so‘ng qayta ishlangan matn mashinani o‘qitish modellariga kiritishga tayyor bo‘ladi.

TensorFlowda matnli modellarni yaratish. TensorFlow matnli ma’lumotlar uchun kuchli modellar yaratishga imkon beruvchi ochiq manbali mashinani o‘qitish kutubxonasidir. Oldindan ishlov berish va vektorlashtirishdan so‘ng, qayta ishlangan ma’lumotlar mashinani o‘qitish modeliga kiritilishi mumkin. TensorFlowda matnli modelni yaratish uchun bir qancha ilg‘or arxitekturalardan foydalanish mumkin. Shulardan biri Hugging Face Transformatorlar kutubxonasida ishlatiladigan BERT modeli arxitekturasidir(Qarang: 2.1.1-chizma).



1-chizma. BERT modeli arxitekturasi

Diagramma transformator modellarining ishlash jarayonini tasvirlaydi. Matn avval tokenizatsiya qilinib, token, segment va pozitsiya embeddinglari joylashtiriladi. So‘ngra, o‘z-o‘ziga e’tibor qatlami va oldinga yo‘naltirilgan neyron tarmoq orqali kodlanadi. Qatlamlarni qo‘shish va normallashtirish barqarorlikni oshiradi. Yakunda zich qatlam va Softmax qatlami natijani hisoblab, tasniflashni amalga oshiradi. Bu model NLP vazifalari uchun samarali ishlatiladi.

Model arxitekturasi aniqlangandan so‘ng, TensorFlowning yuqori darajali API “tf.keras” modelni kompilyatsiya qilishni, yo‘qotish funksiyasini va optimallashtiruvchini belgilashni va modelni ma’lumotlar to‘plamida o‘rgatishni osonlashtiradi. TensorFlow, shuningdek, aniqlik yoki F1 balli kabi standart ko‘rsatkichlar yordamida modelning ish faoliyatini oson baholash imkonini beradi.

Xulosa

Xulosa qilib aytganda, matnni qayta ishlash, NLPning asosiy yo‘nalishlaridan biri bo‘lib, mashinalarga inson tilini tushunish va qayta ishlash imkonini beradi.

Sun’iy intellektning turli ochiq manbali kutubxonalari matnni tasniflash, hissiyotlarni tahlil qilish va mashina tarjimasi kabi vazifalarda muhim ahamiyatga ega.

Matnni qayta ishlashda tokenizatsiya, to‘ldirish va vektorlashtirish asosiy jarayonlardir. Tokenizatsiya matnni kichik bo‘laklarga ajratib, modelga tahlil qilish imkonini beradi. To‘ldirish matn uzunliklarini tenglashtiradi, vektorlashtirish esa matnni raqamli formatda ifodalaydi. Sun’iy intellekt kutubxonalari bu jarayonlarni samarali amalga oshirish uchun kerakli vositalarni taqdim etadi.

BERT modeli va joylashtirish usullari semantik munosabatlarni yaxshilaydi, so‘zlar orasidagi o‘xshashlikni aniqlashga yordam beradi, bu esa matnni tahlil qilishda samarali natijalar beradi.

Foydalanilgan adabiyotlar:

1. Bilal Sharma. Tokenizer and TextVectorization? Difference, 2024.
2. https://www.shirdell.ir/index.php/machine-learning/138-what-is-torchtext-and-what-features-does-it-provide-to-developers
3. Sebastian. NLP: Text vectorization methods using Scikit learn. Berlin, 2019.
4. https://www.educba.com/torch-dot-nn-module/