MUHAMMAD AL-XORAZMIY AVLODLARI
ILMIY-AMALIY VA AXBOROT-TAHLILIY JURNAL

DESCENDANTS OF MUHAMMAD AL-KHWARIZMI
SCIENTIFIC-PRACTICAL AND INFORMATION-ANALYTICAL JOURNAL

1(27)/2024

ISSN-2181-9211

MUHAMMAD AL-XORAZMIY NOMIDAGI TOSHKENT AXBOROT TEXNOLOGIYALARI UNIVERSITETI

MUHAMMAD AL-XORAZMIY AVLODLARI

Ilmiy-amaliy va axborot-tahliliy jurnal
2017 yilda ta’sis etilgan

1(27)/2024

MUNDARIJA

Muassis:

Muhammad al-Xorazmiy nomidagi Toshkent axborot texnologiyalari universiteti
Manzil:
100084, O‘zbekiston, Toshkent sh., Amir Temur ko‘chasi, 108
Telefon: *71 238-64-38;*
e-mail: *alxorazmiy@tuit.uz*
Jurnal sayti: *http://alxorazmiy.uz*

Bosishga ruxsat etildi:
Qog‘oz bichimi 60x84 1/8
Bosma tabog‘i 15,5. Adadi 100 nusxa
Buyurtma raqami №195 “Fan va texnologiyalar Markazining bosmaxonasi”da chop etildi
Toshkent shahri Olmazor ko‘chasi, 171.
Jurnal O‘zbekiston Matbuot va axborot agentligida 2017 yil
22 iyunda 0921 raqami bilan ro‘yxatdan o‘tgan.
Jurnal yilda 4 marotaba
(har chorakda) chop etiladi.

O‘ZBEKISTONDA AXBOROTLASHGAN JAMIYAT RIVOJLANISHINING IQTISODIY MASALALARI

ILMIY AXBOROTLAR

Elov B.B., Xusainova Z. Y.

TABIIY TILNI QAYTA ISHLASHNING ZAMONAVIY ALGORITMLARI

Tabiiy tilni qayta ishlash (NLP) algoritmlari insonning til ma'lumotlarini, shu jumladan, strukturlanmagan matn ma'lumotlarini qayta ishlashga xizmat qiladi. Bugungi kunda NLP algoritmlari til qoidalariga asoslangan, statistik va sun'iy intellektga asoslangan yondashuvlar asosida ishlab chiqiladi. Til qoidalariga asoslangan yondashuv asosida asosan NLP vazifalari uchun lingvistik bazalarni shakllantirish va til korpuslarida razmetkalash amallari bajariladi. Statistik algoritmlar mashinalarga inson tillarini o'qish, tushunish va ma'no olish imkonini beradi hamda katta hajmdagi (bigdata) matnlarni qayta ishlashga asoslanadi. Statistik algoritmlardan nutqni tanib olish, mashina tarjimasi, hissiyotlarni tahlil qilish, matnlarni tasniflash va tahlil qilish kabi ko'plab NLP vazifalarda qo'llaniladi. Bugungi kunda mashinali o'rganish (ML) algoritmlarining CNN va RNN texnologiyalari asoslangan chuqur o'rganish modellari mavjud NLP tizimlarini "o'rganish" imkonini beradi va katta hajmdagi strukturlanmagan matnlarni yanada aniqroq qayta ishlash imkonini beradi. Ushbu maqolada bugungi kundagi NLPning zamonaviy algoritmlari va konsepsiyalari haqida fikr-mulohaza yuritiladi va o'zbek tilidagi matnlarni ushbu algortimlar asosida qayta ishlash usullari keltiriladi.

Kalit so'zlar: NLP, pipeline konveyeri, Levenshtein masofasi, kosinus o'xshashligi, Bag of words usuli, TF-IDF algoritmi.

Kirish

Bugungi kunda ma'lumotlarni intellektual qayta ishlash tabiiy tilni qayta ishlash (Natural Language Processing, NLP)ning ommabop vazifalaridan biri hisoblanadi. Bunda matematik tenglamalar, formulalar, paradigmalar, shablonlarni matn shaklida ifodalash va uni qayta ishlash uchun matn semantikasini (tarkibini) tushunish uchun maqsadida tasniflash va fragmentatsiya kabi amallar bajarilishi lozim.

Dasturchilar kompyuterlar va odamlarga tabiiy til yordamida o'zaro ma`lumot almashish imkonini beruvchi mexanizmlarini ishlab chiqadilar. Kompyuterlar NLP tufayli inson tilini o'qishi, sharhlashi, tushunishi va javob qaytarishi mumkin. Qoidaga ko'ra, ishlov berish mashinaning aql darajasiga asoslanadi. NLP algoritmlari inson xabarlarini uning uchun mazmunli bo'lgan raqamli shakldagi ma'lumotlarga aylantiradi. NLP jarayonlari va ularning algoritmlari qanday ishlashini tushunish juda muhimdir. Zamon bilan hamnafas bo'lish va bu texnologiyalar imkoniyatlaridan samarali foydalanish lozim.

Hozirda NLPda hal qilinadigan vazifalar orasida eng muhimlari quyidagilar (1-rasm) [1, 2]:

mashina tarjimasi (machine translation) – NLP texnologiyalarini ishlab chiquvchilarga yuklatilgan birinchi klassik vazifa;

imloni tekshirish (grammar and spell checking) – o'zbek tilidagi matnlar imlosini avtomatik tekshirish va tuzatish;

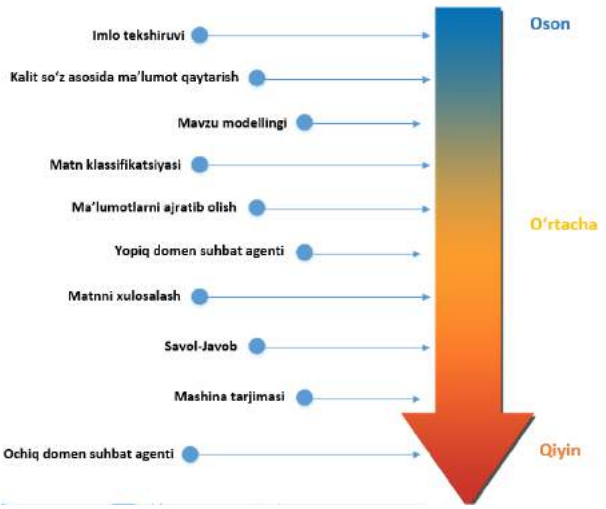
matn tasnifi (text classification) – matn semantikasini aniqlash;

nomlanuvchi obyektlarni aniqlash (named-entity recognition, NER) – ma'noga ega obyektlarni aniqlash;

umumlashtirish (summarization) – matnni soddalashtirilgan shaklga umumlashtirish;

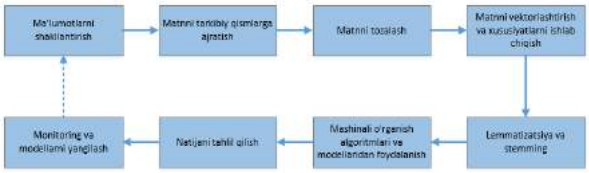
matn generatsiyasi (text generation) – AI tizimlarini yaratishda foydalaniladigan vazifalardan biri;

tematik modellashtirish (topic modeling) – katta hajmdagi matnlardan yashirin mavzularni ajratib olish.



1-rasm. NLP vazifalari murakkabligi

Shuni ta'kidlash kerakki, zamonaviy va dolzarb NLP vazifalarning barchasi ko'pincha interaktiv AI-tizimlarini yaratishda umumlashtiriladi: chat-botlar [3, 4]. Bu inson so'rovlarini dasturiy ta'minot bilan birlashtirishga yordam beradigan muhit (tizim) hisoblanadi. Umuman olganda, NLPdan foydalanadigan tizimlarning ishlashini quydagi konveyer sifatida tasvirlash mumkin [5, 6, 7]:



2-rasm. NLP pipeline konveyeri

– *Matnni (yoki matnga aylantirilgan ovozni) kiritish;*

– *Matnni tarkibiy qismlarga ajratish (segmentatsiya va tokenizatsiya);*

– *Matnni tozalash (keraksiz elementlarni olib tashlash);*

– *Matnni vektorlashtirish va xususiyatlarni ishlab chiqish;*

– *Lemmatizatsiya va stemming;*

– *Mashinali o‘rganish algoritmlari va modellarini o‘qitish usullaridan foydalanish;*

– *Natijani tahlil qilish.*

B.B.Elov, Sh.M.Hamroyeva va Z.Xusainovalar tomonidan o‘zbek tilidagi matnlarni bosqichma-bosqich qayta ishlashning pipeline jarayoni (konveyeri) uchun zarur texnologiyalar, usullar va algoritmlar ishlab chiqilgan va amaliyotga tadbiq etilgan [8]. Matnni qayta ishlash konveyerini rejalashtirish va ishlab chiqish har qanday NLP loyihasini yaratishing boshlang‘ich nuqtasi sifatida qaraladi. Ushbu maqolada zamonaviy NLPda qo‘llaniladigan eng mashhur texnologiyalar, usullar va algoritmlarni tavsiflanadi.

Satrlar o‘xshashligini aniqlash

Tabiiy tilni qayta ishlash odatda matn yoki matnga asoslangan ma’lumotlarni (audio, video) qayta ishlashni taqozo etadi. Bu jarayondagi muhim qadam so‘z va so‘zshakllarni bitta nutq shakliga aylantirishdir. Shunigdek, ko‘p hollarda satrlarning qanchalik o‘xshash yoki farqli ekanligini aniqlash kerak. Odatda, so‘zlar orasidagi farqni ko‘rsatadigan turli ko‘rsatkichlar (metrikalar)dan foydalaniladi [9]. Hozirda starlarning o‘xshashlikni hisoblashning quyidagi usullaridan foydalaniladi:

– Levenshtein masofasi;
– Kosinus o‘xshashligi;
– Hamming masofasi.

Levenshtein masofasi

Oddiy va, ayni paytda, keng miqyosida foydalanish mumkin bo‘lgan metrikalardan biri bu masofani o‘lchashning Levenshtein masofasi deb nomlanadi [10,11,12]. Levenshtein masofasi – ikkita satr qiymatining (so‘z, so‘zshakl, so‘z tarkibi) minimal sonini solishtirish (tahrirlash) orqali o‘xshashligini baholash algoritmi. Levenshtein masofasi matn terish xatolarini aniqlashda uchun ishlatiladi. Quyida a (a = “matematika”) va b (b = “maktab”) so‘zlari orasidagi Levenshtein masofasini aniqlash jarayoni keltirilgan:

M	A		T	E	M	A	T	I	K	A
M	A	K	T			A	B			

3-rasm. Levenshtein masofasini aniqlash jarayoni

Ikki so‘z orasidagi Levenshtein masofasi bir so‘zni boshqasiga almashtirish uchun zarur bo‘lgan bir belgidan iborat tahrirlarning (qo‘shish, o‘chirish yoki almashtirish) minimal sonini ifodalaydi. Shunday qilib, ushbu algoritm quyidagi matn amallarini o‘z ichiga oladi:

– satrga belgi qo‘shish (insert);
– satrdab belgini o‘cherish (delete);
– belgilarni almashtirish (replace).

Ikkita satr o‘rtasidagi Levenshtein masofasini aniqlashning algoritmi, psevdokodi va Python tilidagi tadbig‘i quyida keltirilgan. Levenshtein masofasini aniqlash algotirmi:

$$lev_{a,b}(i,j) == \begin{cases} max(i,j) & Agar\ min(i,j) = 0, \\ min\begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 & aks\ holda \\ lev_{a,b}(i-1,j-1)+1 & (a_i \neq b_j) \end{cases} \end{cases} \quad (1)$$

Levenshtein masofasini aniqlashning psevdokodi va Python tilidagi tatbig‘i quyida keltirilgan:

```
Function levdist(a,b)
   if a.size() == 0 then
      return b.size();
   end
   if b.size() == 0 then
      return a.size();
   end
   indicator ← a[0] 0 ≠ b[0] ? 1 : 0;
   return Minimum(
    levdist(a.substr(1),b)+1,                    //
o‘chirish;
     levdist(b.substr(1),a)+1,                   //
qo‘shish;
     levdist(a.substr(1),b.substr(1))+indicator);    //
almashtirish;
   end
```

```
def levdist(str_1, str_2):
    n, m = len(str_1), len(str_2)
    if n > m:
        str_1, str_2 = str_2, str_1
        n, m = m, n
    current_row = range(n + 1)
    for i in range(1, m + 1):
        previous_row, current_row = current_row, [i]
+ [0] * n
        for j in range(1, n + 1):
            add, delete, change = previous_row[j] + 1,
current_row[j - 1] + 1, previous_row[j - 1]
            if str_1[j - 1] != str_2[i - 1]:
                change += 1
            current_row[j] = min(add, delete, change)
    return current_row[n]
```

Masofani o‘lchash uchun mashhur NLP ilovalari quyidagilar:

– *imloni avtomatik tekshirish (tuzatish) tizimlari;*

– *bioinformatikada DNK ketma-ketliklarining o‘xshashligini miqdoriy aniqlash tizimlari;*

– *matnni qayta ishlash‑ba’zi matn obyektlariga yonma-yon joylashgan so‘zlarning yaqinligini aniqlash tizimlari.*

```
import Levenshtein
print(Levenshtein.distance('fidokor', 'havaskor'))
```

Kosinus o‘xshashligi

Kosinus o‘xshashligi – turli hujjatlardagi matn o‘xshashligini o‘lchash (aniqlash) uchun ishlatiladigan ko‘rsatkichdir. Ushbu ko‘rsatkich uchun hisob-kitoblar vektorning kosinus vektorlari formulasi bo‘yicha o‘xshashligi o‘lchovlariga asoslanadi [10,13]:

$$similarity(A,B) = cos(\theta) = \frac{\bar{A}\cdot\bar{B}}{||\bar{A}||\cdot||\bar{B}||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2}\cdot\sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (2)$$

Bu yerda,

- θ – vektorlar orasidagi burchak;
- $\bar{A}\cdot\bar{B}$ – A ba B ning skalyar ko‘paytmasi bo‘lib, quyidagi formula orqali hisoblanadi:

$$\bar{A}\cdot\bar{B} = A^T B = \sum_{i=1}^{n} A_i B_i == A_1 B_1 + A_2 B_2 + \cdots + A_n B_n \quad (3)$$

$||\bar{A}||$ – L2 norma bo‘lib, quyidagi formula orqali hisoblanadi:

$$||\bar{A}|| = \sqrt{\sum_{i=1}^{n}(A_i)^2} \quad (4)$$

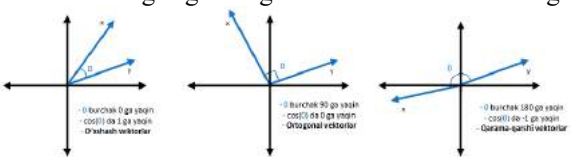
Kosinus o‘xshashlik [-1,1] oralig‘idagi qiymatlarni qabul qilishi mumkin. Vektorlar orasidagi kichikroq burchaklar kattaroq kosinus qiymatlarini hosil qiladi. Bu esa kattaroq qiymatdagu kosinus o‘xshashligini ko‘rsatadi. Masalan:

– Ikki vektor bir xil yo‘nalishga ega bo‘lsa, ular orasidagi burchak 0 ga, kosinus o‘xshashligi esa 1 ga teng.

– Perpendikulyar vektorlar o‘rtasida 90 gradusga va kosinus o‘xshashligi 0 ga teng.

– Qarama-qarshi vektorlar orasidagi burchak 180 gradusga va kosinus o‘xshashligi -1 ga teng.

Quyida 1 ga yaqin, 0 ga yaqin va -1 ga yaqin o‘xshashliklarga ega bo‘lgan ikkita vektorlar keltirilgan:



4-rasm. Kosinus o‘xshashlik vektorlari

Kosinus o‘xshashligi usuli Data Science va Machine Learning ilovalarida keng qo‘llaniladi. Shuningdek, kosinus o‘xshashligi usuli orqali quyidagi turdagi obyektlar o‘zaro taqqoslanadi:

– tabiiy tilda ishlov berishdagi hujjatlar;
– filmlar, kitoblar, videolar;
– grafikli tasvirlar.

Matnni tavsiflovchi vektor sifatida turli xil matn elementlari yoki xususiyatlaridan (masalan, matnni vektorlashtirish usullari) foydalanish mumkin. O‘xshashligini hisoblash lozim bo‘lgan barcha “gaplar”dan vektor maydonini shakllantirish lozim bo‘ladi. Ushbu vektor maydoni barcha gaplarda birlashtirilgan noyob (unikal) so‘zlardan iborat ko‘p o‘lchamga ega bo‘ladi. Kosinus o‘xshashligi vektor fazo modelida quyida ko‘rsatilgan vektorlar orasidagi farqlarni uchta shart uchun hisoblab chiqadi.

Kosinus o‘xshashligini hisoblash natijasi matnning o‘xshashligini tavsiflaydi va kosinus yoki burchak qiymatlari sifatida taqdim etilishi mumkin. Berilgan uchta matnni solishtirgandagi kosinus masofasini hisoblash natijalari (1-jadval) matnlar mos kelganda kosinus qiymatining birga va burchakning nolga yaqinlashishini ko‘rsatadi. Shunday qilib, olingan kosinus o‘xshashlik qiymatlari oddiy semantik matnni oldindan qayta ishlash uchun ishlatilishi mumkin.

```
# Berilgan korpus
str1 = "Adirlar ham bahorda lola bilan go‘zal, chunki lola – bahorning erka guli."
str2 = "Lola ham shifokorlik kasbini tanladi."
str3 = "Bahorda daraxtlar kurtak ochadi."
str4 = "Shifokorlik jamiyatda kerakli kasb."
```

Tahlil natijasini sifatini oshirsih maqsadida quyidagi funksiyani ishlab chiqish lozim:

– *berilgan satrdan tinish belgilarini olib tashlash;*
– *kichik harflarga aylantirish;*
– *nomuhim so‘zlarini olib tashlash.*

```
def clean_string(text):
    text = ''.join([word for word in text if word not in string.punctuation])
    text = text.lower()
    text = ' '.join([word for word in text.split() if word not in stopwords])
    return text
```

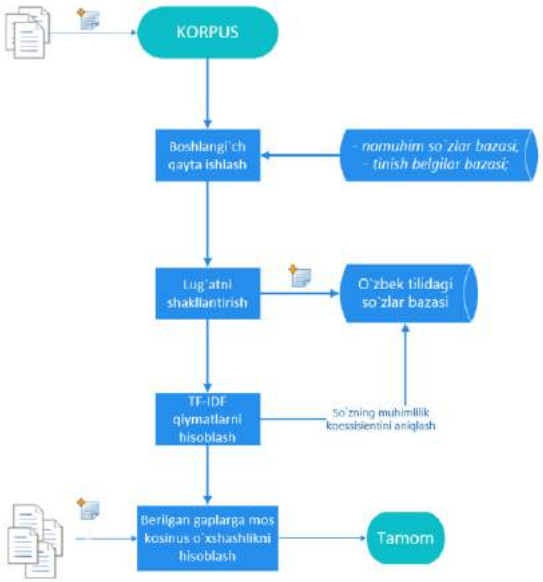
O‘zbek tilidagi so‘z birikmalari yoki gaplarni (3 ta gap) berilgan korpus asosida kosinus o‘xshashligini aniqlash masalasini ko‘rib chiqamiz:

```
# Berilgan gaplar (so‘z birikmalari)
check_str1 = "Lola doim kerak."
check_str2 = "Shifokorlar bahorda dam oladilar."
check_str3 = "Lola – erkatoy go‘zal qizaloq."
```

1-jadval.

Berilgan korpus va matnlarning kosinus o‘xshashligini aniqlash

Matnlar	str1	str2	str3	str4
check_str1	0.55	0.46	0	0
check_str2	0.27	0	0.47	0
check_str3	0.61	0.32	0	0



5-rasm. Kosinus o‘xshashlik qiymatlari aniqlash algoritmining blok-sxemasi

Ushgbu tahlil natijasidan "Lola doim kerak" gapining berilgan korpusdagi str1 = "Adirlar ham bahorda lola bilan go‘zal, chunki lola – bahorning erka guli." matni bilan kosinus o‘xshashlik koeffisienti 0.55 ga teng ekanligini anglash mumkin. Xuddi shu usulda berilgan korpusdagi barcha matnlar bilan kalit so‘zlarini yoki so‘z birikmalarini 5-rasmga keltirilgan algortim asosida kosinus o‘xshashlik qiymatini hisoblash mumkin.

Raqamli vektorlar.

Raqamli vektorlar – matndan muhim atributlarni (xususiyatlarni) aniqlash va mashinali o‘rganish algoritmlarida ulardan foydalanish uchun so‘zlarni sonli formatga o‘tkazish jarayoni [14]. Bugungi kunda NLP vazifalarini hal qilishda **“So‘zlar sumkasi” (Bag of words)** va **“TF-IDF”** kabi raqamli vektorlarni hosil qilish usullaridan keng miqyosida foydalaniladi [15,16]. O‘zbek tilidagi matnlarning kosinus o‘xshashligini aniqlash algoritmida ham berilgan gapga mos TF-IDF qiymatlarni hisoblash boshqichidan foydalanilgan [17].

BOW usuli

Matn ma’lumotlarini vektorlashtirishning eng oddiy usuli quyidagilarni o‘z ichiga oladi [18,19,20]:

– *har bir so‘zga unikal butun son indeksini belgilash;*

– *har bir so‘zning paydo bo‘lish sonini hisoblash va qiymatni tegishli indeks orqali saqlash.*

Natijada, biz matndagi so‘zlarning har biri uchun unikal indeks qiymati va takrorlanish chastotalariga ega vektorni olamiz.

1. Adirlar ham bahorda lola bilan go‘zal, chunki lola – bahorning erka guli.
2. Lola ham shifokorlik kasbini tanladi.

Tokenizatsiya

Adirlar	ham	bahorda	lola	bilan	go‘zal	chunki	bahorning	erka	guli	shifokorlik	kasbini	tanladi
1	1	1	2	1	1	1	1	1	1	0	0	0

Nomuhim so‘zlarsiz lemmatizatsiya

adir	bahor	lola	go‘zal	erka	gul	shifokor	kasb	tanla
1	2	2	1	1	1	0	0	0

6-rasm. BOW vektorini shakllantirish (tokenizatsiya, nomuhim so‘zlarsiz lemmatizatsiya)

Matnni BOW vektor shaklida ifodalashda so‘zlar to‘plamida (*korpusda*) bir nechta unikal so‘zlar (*n_features*) mavjudligini anglatadi. Mualliflar tomonidan yozilgan “Tabiiy tilni qayta ishlashda Bag of Words algoritmidan foydalanish” nomli maqolada BoW modellashtirish algoritmidan foydalanib, o‘zbek tilidagi matnni raqamli matritsalarga aylantirish va qayta ishlash usullari keltirilgan [18].

TF-IDF usuli

TF-IDF usuli orqali so‘zning matndagi barcha boshqa so‘zlarga nisbatan ahamiyati baholanadi. NLP vazifalarini hal qilishda ushbu usuldan matndagi **so‘zlar chastotasi** va **teskari hujjat chastotasi**ni aniqlashda foydalaniladi. Agar biron bir matnda so‘z *ko‘proq*, boshqa matnda *kamroq* uchraydigan bo‘lsa, bu so‘z birinchi matn uchun ko‘proq ahamiyatga ega hisoblanadi. Ushbu usuldagi **TF** – *matndagi so‘zlarning umumiy soniga nisbatan matndagi so‘zning chastotasi*ni, **IDF** – *matndagi so‘zlarning teskari chastotasi*ni ifodalaydi. Berilgan matn uchun TF-IDF qiymatli hisoblash ketma-ketligi quyidagicha:

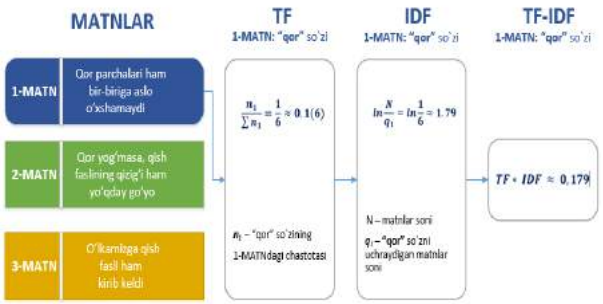
1.Har bir so‘z uchun TF qiymatlarni hisoblash;

2.Ushbu so‘zlar uchun IDF qiymatlarni hisoblash;

*3.Har bir so‘z uchun TF-IDF qiymatini hisoblash (TF*IDF);*

4.Har bir so‘z uchun hisoblangan TF-IDF orqali lug‘atni shakllantirish.

Quyidagi 7-rasmda bitta so‘z uchun TF-IDF qiymatni hisoblash algoritmi ko‘rsatilgan:



7-rasm. TF-IDF algortimini qo‘llash

Vektorizatsiya usulining afzalliklari quyidalar:

– *Nomuhim so‘zlar TF-IDF qiymatda past vazniga ega bo‘ladi (chunki ular barcha matnlarda juda ko‘p marotaba uchraydi) va muhim so‘zlar – yuqori qiymatlarni qabul qiladi.*

– *Matndagi muhim so‘zlar va nomuhim so‘zlarni baholash oson.*

Mualliflar tomonidan yozilgan “O‘zbek tili korpusi matnlari uchun tf-idf statistik ko‘rsatkichni hisoblash” nomli maqolada o‘zbek tili korpusidagi hujjatlarni TF-IDF usulidan foydalanib, kalit so‘zga mos tarzda tartiblash usullari keltirilgan [17]. Amalga oshirilgan tadqiqot natijasida korpusidagi matnlarga mos TF-IDF qiymatlarni hisoblashda matnni normallashtirishning muhim bosqichi hisoblangan lemmatizatsiya jarayonini amalga oshirish orqali tahlil samaradorligini oshishini qayd etilgan.

Matnni normallashtirish

Odatda, NLP masalasi bir necha kichik qismlarga ajratilib, bosqichma-bosqich hal qilinishi lozim. NLPda matnni bosqichma-bosqich qayta ishlash **pipeline jarayoni** (konveyeri) deb yuritiladi [8]. B.Elov, Sh.M.Khamroeva va Z.Y.Xusainovalar tomonidan o‘zbek tili matnlari uchun NLPning pipeline konveyeri amalga oshirish uchun bajariladigan qadamlar va ularning NLP vazifalarini hal qilishdagi ahamiyati haqida mulohaza yuritilgan. Pipeline jarayonining muhim bosqichlaridan biri **matnni normallashtirish** hisoblanadi.

NLPda matnni normallashtirish (yoki so‘zlarni normallashtirish) usullari *matnlarni, so‘zlarni* va *hujjatlarni* **boshlang‘ich qayta ishlash** uchun ishlatiladi. Bunday amallar, odatda, yuqori aniqlikka ega NLP modellarini ishlab chiqish uchun *matnni* (*so‘zlarni yoki nutqni*) to‘g‘ri talqin qilish uchun ishlatiladi.

Bugungi kunda NLP masalalarini hal qilishda barcha holatlar uchun ishlaydigan normalizatsiya vazifalarining “to‘g‘ri” to‘plami yoki ro‘yxati mavjud

emas. Ammo hozirda umumiy maqsadli vositalar to‘plami va oldindan tayyorlangan pipline tizimlaridan ko‘pgina NLP masalalarida foydalanib kelinmoqda. Matnni normallashtirish usullaridan foydalangan holda amalga oshirirlishi lozim bo‘lgan qadamlarni aniq belgilab olish lozim [21].

Matnni normallashtirishda biz **nimani** va **nima uchun** normallashtirishimiz lozimligini aniq bilishimiz kerak. Normallashtirish maqsadi aniq bo‘lgach, qo‘llashimiz kerak bo‘lgan qadamlarni shakllantirish zarur. Ko‘p hollarda normallashtirishdan manfaatdor bo‘lgan ikkita narsa bor:

– **Gap tuzilishi (strukturasi):** Gaplar har doim tinish belgilari bilan tugashi kerakmi? Tinish belgilari takrorlanishi mumkinmi? Barcha tinish belgilarini olib tashlashimiz kerakmi? Muayyan shablon (struktura)dan foydalanish mumkinmi?

– **Lug‘at:** Ko‘pincha so‘z boyligimiz imkon qadar kichikroq bo‘lishini xohlaymiz. Buning sababi shundaki, NLPda so‘zlar bizning asosiy xususiyatlarimiz hisoblanadi. Agar bizda kamroq hajmdagi so‘zlar bilan NLP masalasi hal qilishga imkon bo‘lsa natijaga tezroq erishiladi.

Amalda, NLP masalasini yuqoridahi ikki jihatni inobatga olgan holda matnni normallashtirishni oddiyroq (kichikroq) muammolarga ajratib hal qilish lozim. Quyuda eng keng tarqalganlar NLP amallari ro‘yxati keltirilgan:

– *takrorlanuvchi bo‘shliq (probel)lar va tinish belgilarini olib tashlash;*

– *tutuq belgisini olib tashlash (agar sizning ma’lumotlaringizda “xorijiy” tillardagi diskritik belgilar bo‘lsa – ushbu amal kodlash turi bilan bog‘liq xatolarni kamaytirishga yordam beradi);*

– *katta harflarni kichikka almashtrish (ko‘pincha kichik harflar bilan ishlash yaxshi natijalar beradi. Biroq, ba’zi hollarda, ismlar va joy nomlari kabi ma’lumotlarni olish uchun katta harflar juda muhim);*

– *maxsus belgilarni/emojilarni olib tashlash yoki almashtirish (masalan: heshteglarni olib tashlash);*

– *qisqartirishlarni almashtirish;*

– *so‘z birikmasi shaklidagi sonlarini sonli ko‘rinishga aylantirish (masalan: ‘yigirma uch’→‘23’);*

– *qiymatlarni ularning turiga almashtirish (masalan: ‘$50’→‘pul’);*

– *qisqartmalarni normallashtirish (masalan:‘US’→‘United States’/‘U.S.A’) va abbreviaturalarni normallashtirish (masalan: ‘va h.z,’→‘va hokazo‘);*

– *sana formatlarini yoki standart formatga ega bo‘lgan boshqa ma’lumotlarni normallashtirish;*

– *imloni tuzatish (so‘zni cheksiz yo‘llar bilan noto‘g‘ri yozish mumkin, shuning uchun imlo tuzatishlar “tuzatish” orqali lug‘at o‘zgarishini kamaytiradi) — agar siz tvitlar, online va elektron pochta xabarlari kabi ochiq foydalanuvchi kiritishlari bilan ishlayotgan bo‘lsangiz, bu juda muhim;*

– ***stemming** yoki **lemmatizatsiya** bilan turli variantlarni normallashtirish;*

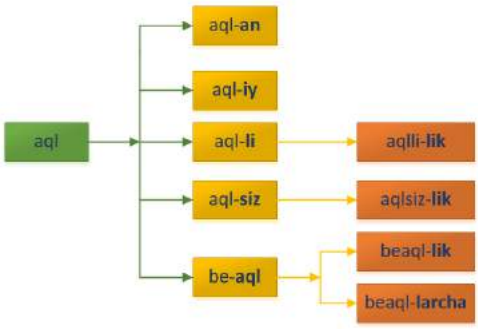
– *kamdan-kam uchraydigan so‘zlarni keng tarqalgan sinonimlar bilan almashtirish;*

– *nomuhim so‘zlarni o‘chirish.*

O‘zbek tili korpusi matnlarini normallashtirishdagi keng tarqalganlar NLP amallari tahlili mualliflarning ishida batafsil keltirilgan. Ushbu maqolada o‘zbek tilidagi matnlaridagi *stemming* va *lemmatizatsiya* jarayonlari va ularning farqlari haqida ma’lumot beriladi [22,23,24,25,26].

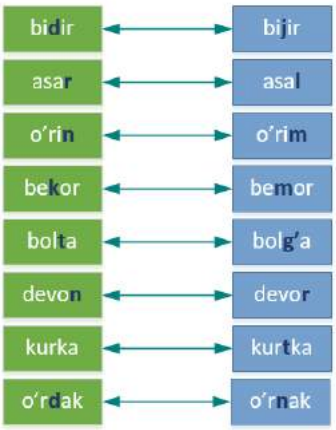
Stemming va lemmatizatsiya

Odatda, matnli hujjatlarda turli xil so‘zshakllardan foydalaniladi, masalan:



8-rasm. “aql” so‘zining so‘zshakllari

Turli ma’noli o‘xshash so‘zlar (**similar definitions**).



9-rasm. Turli ma‘noli o‘xshash so‘zlar

Stemming va lemmatizatsiya jarayoning maqsadi turli so‘zshakllarini, ba’zan esa o‘zakdosh so‘zlarni umumiy asosini aniqlashdan iborat. **Stemming** – bu so‘zlarni ularning asos (o‘zak) shakliga qisqartirish usuli (so‘zning kanonik shaklini hosil qilish). Stemming jarayonida, odatda, so‘zlardagi qo‘shimchalarini kesib tashlaydigan evristik amaldan foydalaniladi (2-jadval).

2-jadval.

Stemming va lemmatizatsiya jarayonlari

so‘zshakl	stem	lemma	asos
kelajagimiz	kelajak	kelajak	kelajak
o‘quvchilar	o‘quv	o‘quvchi	o‘qi
borib ketdi	bor ket (2ta)	borib ketmoq	bor ket (2ta)
ega bo‘lishdi	ega bo‘l (2ta)	ega bo‘lmoq	ega bo‘l (2ta)
taqillatdi	taqilla	taqillamoq	taq
undami	un	u	u

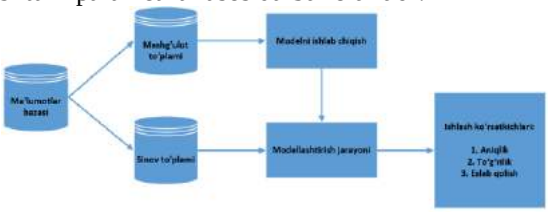
keldilar	kel	kelmoq	kel
uyda	uy	uy	uy
har birimiz	har bir (2ta)	har bir	har bir (2ta)

Lemmatizatsiya so‘zshaklni (yoki so‘zni) asosiy shakli – **lemmaga** aylantiradigan matnni o‘zgartirish jarayoni. Odatda, bu jaratonda *lug‘atlar, morfologik tahlil va so‘z turkumlariga ajratish* kabi amallardan foydalaniladi.

Stemming va lemmatizatsiya o‘rtasidagi farq shundaki, oxirgisi kontekstni oladi va so‘zni lemmaga aylantiradi, stemming esa so‘nggi (ba’zan so‘z boshidagi) bir nechta belgilarni kesib tashlaydi. Bu esa ko‘p hollarda noto‘g'ri ma’no va imlo xatolariga olib keladi. O‘zbek tilidagi matnlarni qayta ishlashning dastlabki ishlov berish bosqichlari hisoblangan *tokenizatsiya, stemming va lemmatizatsiya* jarayonlar haqida [27,28,29] maqolalarda batafsil ma’lumot berilgan.

Naive Bayes algortimi asosida matnlarni tasniflash

Naive Bayesian Analysis (NBA) – matnlarni **tasniflash algoritmi** bo‘lib, xususiyatning mustaqilligi gipotezasiga asoslangan Bayes teoremasi tatbig‘i hisoblanadi. Matnni tasniflash ikki bosqichdan: **o‘rganish bosqichi** va **baholash bosqichidan** iborat. O‘rganish bosqichida klassifikator o‘z modelini berilgan ma’lumotlar to‘plamiga o‘rgatadi. Baholash bosqichida esa klassifikatorning ishlashini sinab ko‘rilradi. Ishlab chiqilgan **model** aniqlik, xatolik va moslashuvchanlik kabi turli parametrlar asosida **baholanadi**.



10-rasm. Matnni tasniflash bosqichlari

NBA sinfdagi biron-bir xususiyatning mavjudligi boshqa hech qanday xususiyat bilan bog‘liq emas deb hisoblaydi. Shuning uchun bunday yondashuv **“Naïve”** deb ataladi. Ushbu tasniflagichning afzalligi *modelni o‘qitish, parametrlarni baholash* va *tasniflash* uchun kichik ma’lumotlarni talab qiladi.

Naive Bayes klassifikatori sinfdagi ma’lum bir xususiyatning ta’siri boshqa xususiyatlardan mustaqil ekanligini taxmin qiladi. Bu xususiyatlar o‘zaro bog‘liq bo‘lsa-da, xususiyatlar mustaqil ravishda ham ko‘rib chiqiladi. Bu taxmin hisoblashni soddalashtiradi va shu sababli **sodda** deb yuritiladi. Ushbu faraz sinfning **shartli mustaqilligi** deb ataladi. Naive Bayes klassıfikatorı har bir xususıyat uchun ehtimolliklarni hisoblab chiqadi va eng yuqori ehtimollik bilan natijani tanlaydi.

$$P(class|data) = \frac{P(data|class)\cdot P(class)}{P(data)} \quad (5)$$

– **P(class):** *h gipotezasining rost bo‘lish ehtimoli (ma’lumotlardan qat’i nazar). Bu h ning aprior ehtimoli sifatida qo‘llaniladi.*

– **P(data):** *ma’lumotlarning ehtimolligi (gipotezadan qat’i nazar). Bu aprior ehtimollik sifatida qo‘llaniladi.*

– **P(class|data):** *D ma’lumotlar asosida h gipoteza ehtimoli. Bu aposterior ehtimollik sifatida qo‘llaniladi.*

– **P(data|class):** *h gipotezasi to‘g‘ri bo‘lganligi sharti asosida ma’lumotlarning d ehtimoli. Bu aposterior ehtimollik sifatida qo‘llaniladi.*

$P(A|B)$ –“**B** hodisa amalga oshgandagi **A** data hodisasi sodir bo‘lish ehtimoli” deb o‘qiladi. Tenglamaning o‘ng tomondagi ifoda *ikkala hodisaning birgalikda sodir bo‘lish ehtimolini B hodisasining yuzaga kelishi ehtimoliga* bo‘lish orqali hisoblanadi.

Naive Bayes klassifikatori

Matnli ma’lumotlarni klassifikator orqali diskret razmetkalarga tasniflashimiz sababli, Naive Bayes algoritmi uchun *kirish funksiyalari to‘plami* hamda *ularga mos tegishli chiqish sinfiga* ega bo‘lamiz. Naive Bayes klassifikatori quyidagi formula yordamida ehtimollikni hisoblaydi:

$$P(y_1|x_1,x_2,x_3) == \frac{P(y_1)\cdot P(x_1|y_1)\cdot P(x_2|y_1)\cdot P(x_3|y_1)}{P(x_1)\cdot P(x_2)\cdot P(x_3)} \quad (6)$$

Ushbu tenglamadagi $P(y_1|x_1,x_2,x_3)$ qabul qilingan $\{x_1,x_2,x_3\}$ ma’lumotlar asosida, chiqish sifatida y_1 bo‘lish ehtimolini anglatadi. Faraz qilaylik, bizning NLP masalamiz jami 2 ta sinfga ega bo‘lsin, ya’ni $\{y_1, y_2\}$. Endi yuqoridagi formuladan avval y_1 sodir bo‘lish ehtimolini, keyin esa y_2 sodir bo‘lish ehtimolini hisoblash lozim bo‘ladi. Qaysi bir ehtimoli yuqori bo‘lsa, bizning taxmin qilingan sinfimiz shu hisoblanadi. Naive Bayes algortimi orqali tasniflash yuqorida keltirilgan qoidalar asosida amalga oshiriladi.

Statistik tasniflash – tasniflangan matnlar ustida o‘tkazilgan kuzatuv natijalaridir. Matnni tasniflashni amalga oshirish uchun, birinchi qadamda muammoni tushunish, potentsial xususiyatlar va razmetkalarni aniqlashdir. Xususiyatlar – belgilash natijalariga ta’sir qiladigan xususiyatlar yoki atributlar to‘plami. Ushbu xususiyatlar modelga matnlarni tasniflashga yordam beradigan xususiyatlar sifatida aniqlanadi. Matnni tasniflash ikki bosqichdan: **o‘rganish bosqichi** va **baholash bosqichi**dan iborat. O‘rganish bosqichida klassifikator o‘z modelini berilgan ma’lumotlar to‘plamiga o‘rgatadi. Baholash bosqichida esa klassifikatorning ishlashi sinab ko‘riladi. Ishlab chiqilgan **model** aniqlik, xatolik va moslashuvchanlik kabi turli parametrlar asosida **baholanadi**.

Naive Bayes klassifikatori yordamida grammatik jihatdan o‘xshash bo‘lgan so‘z turkumlari orasida omonimiya hosil qiluvchi so‘zlarni farqlash masalasini yechilishini ko‘rib chiqsak. Buning bizga so‘z turkumlarni tasniflash xususiyatlar zarur. Buni so‘z turkumlarini grammatik xususiyatlaridan kelib chiqib aniqlash mumkin.

Faraz qilaylik, ot yoki sifat so‘z turkumlari orasidagi omonimiyani aniqlash masalasi qo‘yilgan bo‘lsin. Dastlab ot va sifat so‘z turkumlari uchun tasniflash parametrlari aniqlanishi lozim. Kuzatishlar shuni ko‘rsatadiki, ot va sifat so‘z turkumlari asosan o‘zak va

tarkibida qo‘shimcha mavjud bo‘lgan holatlarda bo‘lishi mumkin, ya’ni **o‘zak va o‘zak +aff (affix)**. Misol tariqasida ot yoki sifat so‘z turkumlari doirasida omonimiya hosil qiluvchi issiq so‘zini keltiramiz.

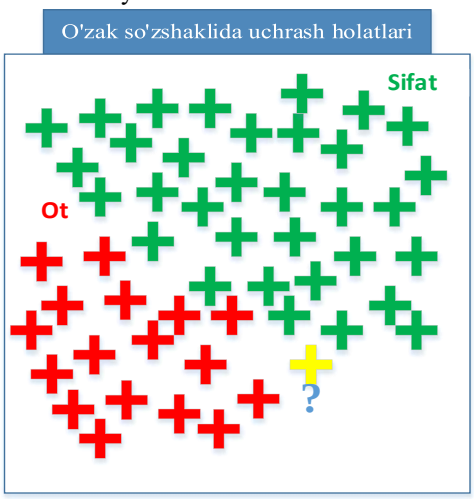
O‘zbek tili korpusi ma’lumotlari orasida *issiq* so‘zini izlaganimizda 30782 ta o‘rinda ishtiroki kuzatildi. Kuzatuvlarda issiq so‘zi

- *Uy **issiq** bo‘lgani uchun tutun yuqoriga ko‘tarilmaydi.*
- *Onamning **issiqqina** bag‘rini, mehrini baribir hech narsa bosolmaydi-da.*
- *Jazirama **issiqda** sun’iy suv havzalarida cho‘milish bolalarga bir olam zavq berishi shubhasiz.*
- *Charchadim o‘qishdanmi, o‘ylovlardanmi, **issiqdanmi**, sovuqdanmi bilmayman.*
- *Metro tomondan ishxonaga intilayotgan, **issiqda** biroz toliqqan Ustozga ko‘zim tushdi.*

kabi so‘zshakllarda uchrash holatlari kuzatildi. Kuzatuvlar natijasida ot yoki sifat so‘z turkumlari orasida omonimlik hosil qiluvchi so‘zlarni tasniflovchi parametrlar ularni *o‘zak* va *o‘zak+aff* ekanligi aniqlandi. Demak x_1= *o‘zak*, x_2= *o‘zak+aff*, y_1=*sifat va* y_2= *ot*. Dastlab o‘zak shaklida uchrash holatlari tahlil qilindi. Tahlillarga ko‘ra issiq so‘zi o‘zak shaklida asosan sifat so‘z turkumiga oid bo‘lishi kuzatildi.

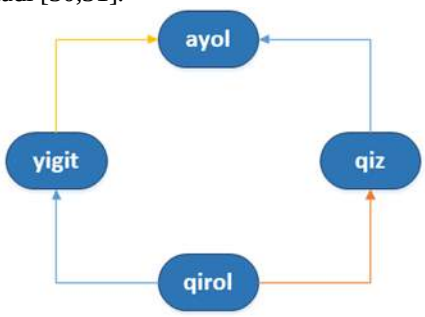
Kuzatuvlar natijasidan olingan statistik ma’lumotlar asosida *issiq* so‘zining aprior va aposterior ehtimolliklarni hisoblash jarayonini soddalashtirish uchun *chastota* va *ehtimollik jadvali*dan foydalanish mumkin. Ushbu jadvallarning ikkalasi ham aprior va aposterior ehtimolliklarni hisoblashda yordam beradi. Chastotalar jadvali barcha xususiyatlar uchun teglarni o‘z ichiga oladi.

Gap tarkibida uchragan ot yoki sifat so‘z turkumlari doirasiga omonimlik hosil qiluvchi so‘zlarning sifat so‘z turkumiga doir bo‘lish ehtimolini hisoblash jarayonini ko‘rib chiqamiz. Uy ***issiq*** bo‘lganligi sababli tutun yuqoriga ko‘tarilmaydi.



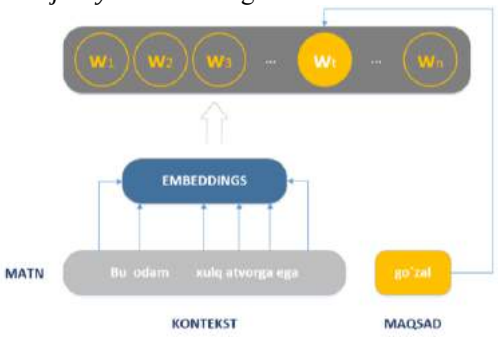
11-rasm. Yangi uchragan issiq so‘zini teglash

So‘zlarni o‘zaro bog‘lash (Word embedding). So‘zlarni o‘zaro bog‘lash – *so‘zlarni, so‘zshakllarni* yoki *so‘z birikmalarini* sonli vektorlari bilan bog‘laydigan NLP modellarini yaratish uchun turli xil usullar va yondashuvlar to‘plami. So‘zlarni o‘zaro bo‘glash tamoyillari: bir xil kontekstda paydo bo‘lgan so‘zlar o‘xshash ma’noga ega. Bunda, **o‘xshashlik** faqat o‘xshash so‘zlar kontekstda joylashishi mumkinligi tushuniladi [30,31].



12-rasm. So‘zlarni o‘zaro bog‘lash

Model so‘zning ehtimolini kontekstga ko‘ra bashorat qiladi. Shunday qilib, NLP modeli so‘z vektorlari bo‘yicha shunday o‘qitiladiki, model tomonidan so‘zga tayinlangan ehtimollik ma’lum kontekstda uning mos kelish ehtimoliga yaqin bo‘ladi (**Word2Vec modeli**). So‘zning kontekst bo‘yicha o‘xshashlik ehtimoli **softmax formulasi** orqali hisoblanadi. Bu NLP-modelni *xatolarni orqaga qaytarish jarayoni* bilan o‘rgatish uchun ishlatiadi.



13-rasm. So‘zning kontekst bo‘yicha o‘xshashlik ehtimolini aniqlash

Eng mashhur ilovalar:

– ***Word2Vec*** *– so‘z konteksti asosida so‘zlarni joylashtirishni hisoblash uchun neyron tarmoqlardan foydalanadi.*

– ***GloVe*** *– matnda so‘zlarning birgalikda paydo bo‘lish ehtimolini tavsiflovchi so‘z vektorlarining kombinatsiyasidan foydalanadi.*

– ***FastText*** *– Word2Vec ga o‘xshash usul bo‘liv, faqat so‘zlar o‘rniga ularning qismlari va belgilaridan foydalanadi hamda natijada so‘z uning kontekstiga aylanadi.*

Xulosa

Ushbu maqolada matnni tahlil qilish va bir qator NLP amaliy vazifalarni hal qilish imkonini beradigan zamonaviy algoritmlar va konseptsiyalar ko‘rib chiqildi. Oddiy o‘xshashlik ko‘rsatkichlari, matnni normallashtirish, vektorlashtirish, so‘zlarni o‘zaro bog‘lash, Naive Bayes va LSTM algortimlari haqida ma’lumot berildi. Keltirilgan usul va algortimlar NLP

masalalarini hal qilish uchun muhim hisoblanadi va ushbu sohani o‘rganishga yordam beradi.

Foydalanilgan adabiyotlar

[1] Zhou, M., Duan, N., Liu, S., & Shum, H.-Y. (2020). Progress in Neural NLP: Modeling, Learning, and Reasoning. Engineering, 6(3), 275–290. https://doi.org/10.1016/j.eng.2019.12.014

[2] Bulusu, A., & Sucharita, V. (2019). Research on machine learning techniques for POS tagging in NLP. International Journal of Recent Technology and Engineering, 8(1 Special Issue 4).

[3] Ghorpade-Aher, J., Kontamwar, R., Kukreja, S., Karpe, T., & Kakkad, S. (2019). An overview of NLP based Chatbots. Universal Review, VIII(II).

[4] Bhagwat, V. A. (2018). Deep Learning for ChatBots. Scholarworks.Sjsu.Edu.

[5] Becquin, G. (2020). End-to-end NLP Pipelines in Rust. https://doi.org/10.18653/v1/2020.nlposs-1.4

[6] Tenney, I., Das, D., & Pavlick, E. (2020). BERT rediscovers the classical NLP pipeline. ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference. https://doi.org/10.18653/v1/p19-1452

[7] Attardi, G. (2015). DeepNL: A deep learning NLP pipeline. 1st Workshop on Vector Space Modeling for Natural Language Processing, VS 2015 at the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2015. https://doi.org/10.3115/v1/w15-1515

[8] B.Elov. Tabiiy tilni qayta ishlash (NLP)da spacy modulidan foydalanish. Илм-фан ва инновацион ривожланиш, 2022 (4). 41-55.

[9] N.Xudayberganov, Sh.Hasanov. Tabiiy tilni qayta ishlashda so‘zlar orasidagi masofani aniqlash algoritmidan foydalanish // O‘zbekiston: til va madaniyat. Amaliy filologiya masalalari. 2022-yil 5 (2) son. − B. 69-83.

[10] Yujian, L., & Bo, L. (2007). A normalized Levenshtein distance metric. IEEE Transactions on Pattern Analysis and Machine Intelligence, 29(6). https://doi.org/10.1109/TPAMI.2007.1078

[11] Zhao, C., & Sahni, S. (2019). String correction using the Damerau-Levenshtein distance. BMC Bioinformatics, 20. https://doi.org/10.1186/s12859-019-2819-0

[12] Behara, K. N. S., Bhaskar, A., & Chung, E. (2020). A novel approach for the structural comparison of origin-destination matrices: Levenshtein distance. Transportation Research Part C: Emerging Technologies, 111. https://doi.org/10.1016/j.trc.2020.01.005

[13] Darmalaksana, W., Slamet, C., Zulfikar, W. B., Fadillah, I. F., Maylawati, D. S. adillah, & Ali, H. (2020). Latent semantic analysis and cosine similarity for hadith search engine. Telkomnika (Telecommunication Computing Electronics and Control), 18(1). https://doi.org/10.12928/TELKOMNIKA.V18I1.14874

[14] Stecanella, B. (2019). What is TF IDF? MonkeyLearn.

[15] Jalilifard, A., Caridá, V. F., Mansano, A. F., Cristo, R. S., & da Fonseca, F. P. C. (2021). Semantic Sensitive TF-IDF to Determine Word Relevance in Documents. Lecture Notes in Electrical Engineering, 736 LNEE. https://doi.org/10.1007/978-981-33-6987-0_27

[16] Pietro, M. di. (2020). Text Classification with NLP: Tf-Idf vs Word2Vec vs BERT. Medium.

[17] B. Elov, Z. Husainova, N.Xudayberganov. (2022). O‘zbek tili korpusi matnlari uchun TF-IDF statistik ko‘rsatkichni hisoblash. International scientific journal of “Science and Innovation”. Issue 8, https://doi.org/10.5281/zenodo.7440059

[18] B.Elov, Z.Xusainova, N.Xudayberganov. (2022). Tabiiy tilni qayta ishlashda Bag of Words algoritmidan foydalanish. Til va madaniyat, Amaliy filologiya masalalari. Vol.2(5). − B. 35-50.

[19] Zhang, Y., Jin, R., & Zhou, Z. H. (2010). Understanding bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics, 1(1–4). https://doi.org/10.1007/s13042-010-0001-0

[20] Yadav, A. K., & Borgohain, S. K. (2015). Sentence generation from a bag of words using N-gram model. Proceedings of 2014 IEEE International Conference on Advanced Communication, Control and Computing Technologies, ICACCCT 2014.
https://doi.org/10.1109/ICACCCT.2014.7019414

[21] Ariffin, S. N. A. N., & Tiun, S. (2020). Rule-based text normalization for malay social media texts. International Journal of Advanced Computer Science and Applications, 11(10).
https://doi.org/10.14569/IJACSA.2020.0111021

[22] Webster, J. J., & Kit, C. (1992). Tokenization as the initial phase in NLP. https://doi.org/10.3115/992424.992434

[23] Rai, A., & Borah, S. (2021). Study of various methods for tokenization. In Lecture Notes in Networks and Systems (Vol. 137). https://doi.org/10.1007/978-981-15-6198-6_18

[24] Hafsa Jabeen. (2018). Stemming and Lemmatization in Python. Towardsdatascience.

[25] Balakrishnan, V., & Ethel, L.-Y. (2014). Stemming and Lemmatization: A Comparison of Retrieval Performances. Lecture Notes on Software Engineering, 2(3). https://doi.org/10.7763/lnse.2014.v2.134

[26] Khyani, D., S, S. B., M, N. N., & M, D. B. (2021). An Interpretation of Lemmatization and Stemming in Natural Language Processing. Journal of University of Shanghai for Science and Technology, 22(10).

[27] B.Elov, Sh.Hamroyeva, D.Elova. (2022). Morfologik analizatorni yaratish usullari, O‘zbekiston: til va madaniyat (Amaliy filologiya), 2022, 5(1).

[28] B.Elov, Sh.Hamroyeva, X.Axmedova. (2022). Methods for creating a morphological analyzer, 14th International Conference on Intellegent Human Computer Interaction, IHCI 2022, 19-23 October 2022, Tashkent.

[29] Z.Xusainova. Tokenizatsiya algoritmlari. Ilmiy innovatsion jurnal: " Filologik tadqiqotlar:Til,adabiyot, ta'lim". 2022/№5-6. b.73-76

[30] Rudkowsky, E., Haselmayer, M., Wastian, M., Jenny, M., Emrich, Š., & Sedlmair, M. (2018). More than Bags of Words: Sentiment Analysis with Word Embeddings. Communication Methods and Measures, 12(2–3). https://doi.org/10.1080/19312458.2018.1455817

[31] Wang, B., Wang, A., Chen, F., Wang, Y., & Kuo, C. C. J. (2019). Evaluating word embedding models: Methods and experimental results. In APSIPA Transactions on Signal and Information Processing (Vol. 8). https://doi.org/10.1017/ATSIP.2019.12

Elov Botir Boltayevich
Texnika fanlari bo‘yicha (PhD), dotsent
Alisher Navoiy nomidagi Toshkent davlat o‘zbek tili va adabiyoti universiteti
E-mail: elov@navoiy-uni.uz

Xusainova Zilola Yuldashevna
Alisher Navoiy nomidagi Toshkent davlat o‘zbek tili va adabiyoti universiteti stajor-o‘qituvchi.
E-mail: xusainovazilola@navoiy-uni.uz

Elov B.B., Xusainova Z.Y.

Modern algorithms of natural language processing

Natural language processing (NLP) algorithms serve to process human language data, including unstructured text data. Today, NLP algorithms are developed based on language rule-based, statistical and artificial intelligence approaches. Based on the approach based on language rules, the formation of linguistic bases for NLP tasks and the operations of classification in language corpora are performed. Statistical algorithms allow machines to read, understand and derive meaning from human languages and are based on processing large volumes of (bigdata) texts. Statistical algorithms are used in many NLP tasks such as speech recognition, machine translation, sentiment analysis, text classification and analysis. Today, deep learning models of machine learning (ML) algorithms based on CNN and RNN technologies allow to "learn" existing NLP systems and allow more accurate processing of large volumes of unstructured texts. This article discusses modern algorithms and concepts of NLP today, and methods of processing texts in the Uzbek language based on these algorithms are presented.

Keywords: NLP, pipeline pipeline, Levenshtein distance, cosine similarity, Bag of words method, TF-IDF algorithm.