ISSN 2181-922X 69—83

Tabiiy tilni qayta ishlashda so‘zlar orasidagi masofani aniqlash algoritmlaridan foydalanish

Nizomaddin Xudayberganov[1]
Shaxboz Hasanov[2]

Abstrakt:

So‘zlar orasidagi masofa – so‘zlarning turli sifatlariga ko‘ra tavsiflanishi mumkin. Ularning aynan tarkibiy qismiga ko‘ra o‘xshashligini aniqlash uchun tabiiy tilni qayta ishlash yo‘nalishida turli algoritmlar taklif qilinadi. Bu algoritmlar o‘zining ishlatilish o‘rinlari hamda unumdorligi bilan bir-biridan farq qiladi. Asosan sohaning imlo tekshiruvi, nutqni aniqlash hamda plagiatni aniqlash kabi yo‘nalishlarida foydalanish uchun qo‘llaniladi. Ushbu maqolada birdan ortiq so‘zlarning o‘xshashlik masofasini aniqlashda foydalaniladigan algoritmlar va ularga xos xususiyatlar tahlil qilinadi.

Kalit so‘zlar: *Hamming masofasi, Levenshteyin masofasi, Kosinus o‘xshashligi, o‘xshashlik jadvali, kodlash nazariyasi*.

Kirish

Tabiiy tilni qayta ishlash hamda mashinali o‘rganish yo‘nalishlarida masofaviy o‘lchovlar muhim rol o‘ynaydi. Masofa o‘lchovi - bu muammoli sohadagi ikkita obyekt o‘rtasidagi nisbiy farqni umumlashtiruvchi qiymat hisoblanadi. Odatda, bu ikkita obyektni tavsiflovchi ma’lumotlar qatoridan foydalanib maxsus algoritmlar yordamida aniqlanuvchi qiymat sifatida qaraladi. Ma‘lumotlarning turlariga qarab turli xil masofa o‘lchovlari tanlanishi va ishlatilishi kerak. Shunday qilib, turli xil mashhur masofa o‘lchovlarini va natijada olingan qiymatlarni hisoblashni bilish muhimdir. So‘zlar orasidagi o‘xshashlik masofasi – ularning mazmunidan ko‘ra shaklan

[1]*Xudayberganov Nizomaddin Uktamboy o‘g‘li* – Alisher Navoiy nomidagi Toshkent davlat o‘zbek tili va adabiyoti universiteti Kompyuter lingvistikasi va raqamli texnologiyalar kafedrasi o‘qituvchisi.
E-pochta*:* nizomaddin@navoiy-uni.uz
ORCID*:* 0000-0002-6213-3015

[2]*Hasanov Shaxboz G‘olibjon o‘g‘li* – Toshkentdagi INHA universiteti talabasi.
E-pochta*:* shaxbozhsh@gmail.com
ORCID*:* 0000-0002-2368-2863
Iqtibos uchun*:* Xudayberganov N., Hasanov Sh. 2022. “Tabiiy tilni qayta ishlashda so‘zlar orasidagi masofani aniqlash algoritmlaridan foydalanish”. *O‘zbekiston: til va madaniyat. Amaliy filologiya*. 2 (5): 69-83.

bir-biriga qanchalik yaqinligini ko‘rsatuvchi qiymat. Maqolada quyidagi algoritmlar orqali ushbu qiymatni aniqlash hamda undan qanday foyda olish ko‘rib chiqiladi.

1. Hamming masofasi.
2. Levenshteyin masofasi.
3. Kosinus o‘xshashligi.

Mavjud nazariyalarni o‘rganish davomida shu narsa aniqlandiki, o‘zbek tilidagi so‘zlar raqamli ko‘rinishga olib kelinishi lozim. Ushbu qiymatlar uchun Python tilidan foydalanish maqsadga muvofiq.

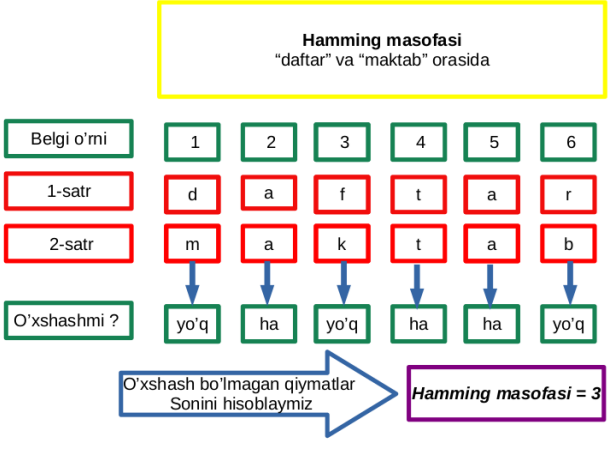
Asosiy qism

Berilgan satrlarni taxminan moslashtirish, shuningdek, katta hajmdagi matn orasidan kerakli ma’lumotlarni qidirish kabilarni amalga oshirish mumkin, bu esa o‘z navbatida imlo tekshiruvi, maxsus belgilarni aniqlash uchun tuzatish tizimlari, nutqni aniqlash, spamni filtrlash, to‘g‘ri yozuvni tekshirish kabi turli xil ilovalarda qo‘llaniladi. Bundan tashqari tabiiy tilni tushunishda aniqlikni oshirish, plagiatni aniqlash kabi yo‘nalishlarda samarali qo‘llash mumkin bo‘ladi. Matnlardagi plagiat akademik hamjamiyatni tashvishga solayotgan masalalardan biri hisoblanadi. Endi eng keng tarqalgan matn plagiati so‘zlarni kiritish, o‘chirish yoki almashtirishni o‘z ichiga olgan turli xil kichik o‘zgartirishlar kiritish orqali yuzaga keladi. Biroq, bunday oddiy o‘zgarishlar plagiatni aniqlash jarayonida ortiqcha taqqoslashni talab qiladi. Ushbu maqola orqali jarayonning samaradorligini oshiruvchi bir qancha nazariyalar taqdim qilinadi.

Hamming masofasi. Hamming masofasi - tabiiy tilni qayta ishlash yo‘nalishida teng uzunlikdagi ikki satr (so‘z yoki so‘zlar jamlanmasi) orasidagi mos belgilari turli xil bo‘lgan o‘rinlar sonidir. Boshqacha qilib aytganda, u bir qatorni boshqasiga o‘zgartirish uchun zarur bo‘lgan o‘zgartirishlarning minimal sonini yoki bitta qatorni boshqasiga o‘zgartirishi mumkin bo‘lgan xatolarning minimal sonini o‘lchaydi. Umumiyroq matnda Hamming masofasi ikki ketma-ketlik orasidagi tahrirlash masofasini o‘lchash uchun bir nechta qator ko‘rsatkichlaridan biridir. Hamming masofasi amerikalik matematik olim Richard Hamming sharafiga nomlangan. 1950-yilda Hamming kodlari, Xatolarni aniqlash va xatolarni tuzatish kodlari haqidagi fundamental maqolasida konseptsiyani kiritgan. Bitlarning Hamming masofasi tahlili axborot texnologiyalari, kodlash nazariyasi va kriptografiya kabi bir qancha fanlarda qo‘llaniladi.[Bill, 2020, 206]

Algoritmni tahlil qilish jarayonida berilgan satrlarning mos

belgilarini o‘zaro taqqoslash hamda Hamming masofasini aniqlovchi o‘zgaruvchi olishimiz va uni doimo o‘zgarishlar jarayonida oshirib borishimiz talab qilinadi. Misol qilib “daftar” hamda “maktab” so‘zlardan foydalanilganda ular orasidagi Hamming masofasini hisoblash ko‘rsatilgan.



1-rasm. Hamming masofasini aniqlashga oid misollar

Boshqacha qilib aytganda, “daftar” so‘zidan “maktab” so‘zini hosil qilishimiz uchun 3 ta belgini o‘zgartirish yetarli bo‘ladi. Satrlarda belgilar soni tengligi orqali natijaga erishishimiz uchun faqatgini belgilar almashuvini qo‘llashimiz yetarli bo‘ladi. Xuddi shu usulda quyidagi misollarda ham Hamming masofasi hisoblanadi.

1. “kun” va “tun” bir xil uzunlikdagi so‘zlar aynan 1 ta turli belgiga ega demak, ular orasidagi Hamming masofasi 1 ga teng bo‘ladi.

2. “uycha” va “uyat” so‘zlarida o‘zbek tilida ikkala so‘z ham 4 ta harfdan iborat, lekin belgilar soni turli (mos ravishda 5 ta va 4 ta) bo‘lganligi sababli Hamming masofasini aniqlashning iloji yo‘q.

3. 00010011 va 01010111 ikkilik sanoq sistemasiga tegishli bo‘lgan ushbu sonlar orasidagi 3-va 6-belgilar farq qilganligi sababli Hamming masofasi 2 ga teng.

O‘zgarmas n uzunlikdagi satr uchun Hammming masofasi n uzunlikdagi so‘zlar to‘plamining ko‘rsatkichidir (hamming fazosi deb ham ataladi). Chunki u musbat va simmetriya shartlarini bajaradi. Agarda ikki so‘zning Hamming masofasi 0 ga teng bo‘lsa, ular aynan bir xil bo‘ladi. Bundan tashqari ushbu qiymatlar uchburchak tengsizligini ham qanoatlantiradi. Haqiqatan ham, agar uchta a, b va c so‘zlari ko‘riladigan bo‘lsa, a satrning i-o‘rindagi qiymati

b satrning i-o‘rindagi qiymati va c satrning xuddi shu qiymatlari solishtiriladigan bo‘lsa nazariya to‘g‘riligini ko‘rish mumkin bo‘ladi. Demak, a va c orasidagi Hamming masofasi a va b hamda b va c orasidagi Hamming masofalarining yig‘indisidan katta emas. Ikkita ikkilik sanoq sistemasidagi a va b qiymatlar uchun Hamming masofasi a XOR b dagi birlar soniga teng. Ushbu qiymat esa kodlash nazariyasida xatolikni aniqlash va xatolarni tuzatish yo‘nalishida foydalaniladi [Derek, 2003, 255].

Tabiiy tilni qayta ishlash yo‘nalishida qisqa uzunlikdagi so‘zlar orasidagi masofalarni aniqlash hamda ularni to‘g‘irlash orqali foydalanish maqsadga muvofiq bo‘ladi. Bundan tashqari kodlash nazariyasi sohasida ham ikkilik sanoq sistemasida berilgan xatoliklarni aniqlash va bartaraf etish uchun samarali yechim bo‘la oladi. Katta hajmdagi matn yoki hujjatlar uchun foydalanish uchun yuqori samaradorlikka ega emas.

Psevdokodi

```
function  hamming_masofasi(a,b)
        indicator ← 0
        len1 ← length(a)
        for n from 1 to len1 do
                if a[n] <> b[n] then
                        indicator ←  indicator +1
        return indicator
end
```

Algoritmning Python dasturlashga tatbiq qilinishi. Hamming masofasini hisoblovchi funksiya yaratiladi hamda unga 2 ta satr elementlari qabul qilinadi. Funksiya ishlash jarayonida berilgan qiymatlarning har bir belgisini o‘zaro tekshiradi hamda yakuniy qiymatni qaytaradi.

```
def hamming_masofasi (satr1, satr2):
        masofa_hisoblovchi = 0
        for n in range(len(satr1)):
                if satr1[n]!=satr2[n]:
                        masofa_hisoblovchi += 1
        return masofa_hisoblovchi
```

Levenshteyin masofasi. Levenshteyin masofasi (**Levenshteyin distance**) — tabiiy tilni qayta ishlash yo‘nalishida Levenshteyin masofasi ikki satr (bir yoki bir nechta so‘zlar) o‘rtasidagi masofani o‘lchash uchun qo‘llanadigan ko‘rsatkichidir. Boshqacha qilib aytganda, ikki so‘z orasidagi Levenshteyin masofasi bir so‘zni boshqasiga o‘zgartirish uchun zarur bo‘lgan tahrirlarning

(qo‘shish, o‘chirish yoki almashtirish) minimal soni hisoblanadi. Bu masofani 1965-yilda Vladimir Levenshteyin sharafiga nomlangan. Levenshteyin masofasini **tahrirlash masofasi** deb ham atash mumkin. Vladimir Iosifovich Levenshteyin - rus olimi bo‘lib, u axborot texnologiyalari, xatolarni to‘g‘rilash nazariyalari va kombinator dizayni bo‘yicha tadqiqotlar olib borgan. Fanga qo‘shgan boshqa hissalari orasida u 1965-yilda ishlab chiqqan Levenshteyin masofasi va Levenshteyin algoritmi mashhur [Levenshteyin, 1966].

Demak algoritm a satrni b satrga aylantirish uchun zarur bo‘lgan o‘zgarishlarning minimal sonini aniqlashda foydalaniladi. Ushbu jarayonni amalga oshirishda 3 xildagi amallar qo‘llaniladi: belgi qo‘shish, o‘chirish hamda almashtirish. Ushbu algoritmda satrlar uzunligi bir xil bo‘lishi talab qilinmaydi. Ushbu nazariyaning samaradorligini oshirish maqsadida pastdan yuqoriga dinamik dasturlashning namunasi bo‘lgan algoritm 1974-yilda Robert A. Vagner va Maykl J. Fisherning “Satrdan-satrga tuzatish muammosi” maqolasida muhokama qilingan.

Levenshteyin masofasidan tabiiy tilni qayta ishlash yo‘nalishida so‘zlarning o‘zaro mosligini aniqlashda foydalanish mumkin. Bunda asosiy maqsad kam sonli farqlar kutilgan vaziyatlarda, berilgan matnlardagi satrlar o‘xshashligini aniqlash bo‘ladi. Bu esa masalan, imlo tekshiruvi, optik belgilarni aniqlash uchun tuzatish tizimlari va tarjima xotirasi asosida tabiiy tilga tarjima qilishga yordam beradigan dasturiy ta’minotlar uchun asosiy qismlardan biri sifatida xizmat qila oladi. Optik belgilarni aniqlash yoki optik belgilarni o‘quvchi dasturlar - skanerlangan hujjat, qo‘lyozma fotosurati yoki subtitrdan bo‘ladimi, qo‘lda yozilgan yoki bosilgan matn tasvirlarini mashina tiliga elektron yoki mexanik aylantirish jarayonlaridan tashkil topadi. Bundan tashqari ushbu o‘xshashlikdan plagiatni tekshirish kabi maqsadlarda foydalanish muvaffaqiyatli natija beradi. Bundan tashqari taxminiy satrlarni moslashtirish jarayonida ham keng qo‘llaniladi. Taxminiy satrlarni moslashtirish (ko‘pincha so‘zlashuv tilida noaniq satrlarni qidirish deb ataladi) aniq bo‘lmagan ko‘rinishdagi namunalarga mos keladigan satrlarni topish usulidir. Taxminiy satrlarni moslashtirish muammosi odatda ikkita kichik muammoga bo‘linadi: berilgan satr ichida taxminiy namuna mosliklarini topish va namunaga taxminan mos keladigan satrlarni topish.

Levenshteyin masofasi bir nechta oddiy yuqori va quyi chegaralarga ega. Bularga quyidagilar kiradi:

1. Berilgan ikki satrlar uzunliklari orasidagi farq.

2. Uzunligi kattaroq bo‘lgan satrning uzunlik qiymatini aniqlovchi birlik.

3. Agar satrlar bir xil uzunlikka ega bo‘lsa ular orasidagi farq 0 ga teng.

4. Agar satrlar bir xil o‘lchamga ega bo‘lsa, Hamming masofasi Levenshteyin masofasining yuqori chegarasi hisoblanadi.

5. Ikki satr orasidagi Levenshteyin masofasi ularning uchinchi satr bilan o‘zaro Levenshteyin masofalarining yig‘indisidan katta emas (uchburchak tengsizligi o‘rinli).

Levenshteyin masofasini ikkita uzunroq satrlar orasida ham hisoblash mumkin, ammo uni hisoblash uchun sarflanadigan vaqt, bu ikki satrlar uzunligiga taxminan proporsional bo‘lib, buni amaliy jarayonlarda foydalanish nisbatan kamroq unumdorlikka sabab bo‘ladi. Shunday qilib, taqqoslash tezligini yaxshilash maqsadida odatda qisqa satrlardan foydalaniladi.

Quyidagi misollar orqali nazariy bilimlarni amaliy ko‘rinishga olib kelish mumkin.

1. “olma” so‘zidan “olmaxon” so‘zini hosil qilish uchun so‘zning davomiga “x”, “o” va “n” harflarini qo‘shish orqali natijaga erishiladi. Bunda umumiy 3 o‘zgarish, ya’ni qo‘shish amalga oshirildi. Demak Levenshteyin masofasi 3 ga teng bo‘ladi.

2. Yuqoridagi misolni teskari holati qaraladigan bo‘lsa, “olmaxon” so‘zidan “olma” so‘zini hosil qilish uchun 3 o‘zgarishni, ya’ni so‘z oxiridagi 3 harfni o‘chirishni amalga oshirish talab qilinadi. Bu vaziyatda ham Levenshteyin masofasi 3 ga teng bo‘ladi.

3. “men” so‘zi xatolik tufayli “man” kabi yozilgan vaziyatda Levenshteyin masofasi nechta o‘zgartirish orqali to‘g‘ri so‘zga erishish mumkinligini ko‘rsatadi. Bu vaziyatda faqat “a” harfini “e” harfiga o‘zgartirish yetarli. Demak Levenshteyin masofasi 1ga teng.

Endi umumiy vaziyatda Levenshteyin masofasini yanada chuqurroq tushunish maqsadida quyidagi o‘zbek tiliga oid bo‘lgan “matematika” hamda “maktab” so‘zlari ko‘rib chiqiladi. Berilgan “matematika” so‘zidan “maktab” so‘zini hosil qilish uchun o‘zaro mos tushuvchi harflarni belgilab olish hamda boshqa belgilar uchun o‘zgarishlarni amalga oshirish lozim bo‘ladi.

1. M, A harflaridan so‘ng K harfini qo‘shish (1 ta o‘zgarish)

2. 1- T harfidan so‘ng E, M harflarini o‘chirish (2 ta o‘zgarish)

3. 2- T harfini B harfiga almashtirish (1 ta o‘zgarish)

4. 2- T harfidan so‘ng I, K va A harflarini o‘chirish (3 ta o‘zgarish)

			BELGI QO'SHISH		BELGI O'CHIRISH	BELGI O'CHIRISH		BELGI ALMASHTIRISH	BELGI O'CHIRISH	BELGI O'CHIRISH	BELGI O'CHIRISH
a=	M	A		T	E	M	A	T	I	K	A
b=	M	A	K	T			A	B			

2-rasm. Levenshteyin masofasini aniqlashga oid misollar

O‘zgarishlar soni: 1ta belgi qo‘shish, 1ta belgi almashtirish va 5ta belgini o‘chirish. Demak, Levenshteyin masofasi = 7ga teng bo‘ladi.

Levenshteyin jadvali - bu 2 o‘lchamli massiv bo‘lib, lev[i][j] - Levenshteyin masofasining qiymatiga teng bo‘ladi. Bunda: i = ‘a’ satr uzunligi] va j = [‘b’ satr uzunligi] [Grashchenko, 2022]

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

3-rasm. Levenshteyin jadvalini hosil qilish qoidasi

Berilgan formula orqali Levenshteyin jadvali hosil qilinadi.

	i→	M	A	T	E	M	A	T	I	K	A
jl	0	1	2	3	4	5	6	7	8	9	10
M	1	0	1	2	3	4	5	6	7	8	9
A	2	1	0	1	2	3	4	5	6	7	8
K	3	2	1	1	2	3	4	5	6	6	7
T	4	3	2	1	2	3	4	4	5	6	7
A	5	4	3	2	2	3	3	4	5	6	6
B	6	5	4	3	3	3	4	4	5	6	7

4-rasm. Levenshteyin jadvali

Psevdokodi

```
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

Python dasturlash tiliga tatbig‘i. Avvalambor, berilgan satrlar uzunligi orqali maxsus Levenshteyin jadvalimizni hosil qilamiz. Keyingi qadamlarni kerakli bo‘lgan qiymatlarni hisoblash hamda ularni maxsus jadvalga saqlash orqali olib boriladi.

```
#Levenshteyn algoritmi
# ‘a’ va ‘b’ qiymatlarni aniqlash:
a = “matematika”
b = “maktab”

# lev o‘zgaruchi massivini rows = len(a) + 1 va
columns = len(b) + 1 orqali  kiritish
lev = [[0 for i in range(len(b) + 1)] for j in
range(len(a) + 1)]

# Birinchi qatordagi(row) sonlarni aniqlash:
for i in range(len(a) + 1):
lev[i][0] = i
# Birinchi ustundagi(column) sonlarni aniqlash:
for j in range(len(b) + 1):
lev[0][j] = j

for i in range(1, len(a) + 1):
        for j in range(1, len(b) + 1):
                if a[i - 1] == b[j - 1]:
                lev[i][j] = lev[i - 1][j - 1]
                else:
                        insertion = 1 + lev[i][j - 1]
#belgini qo‘shish
                        deletion = 1 + lev[i - 1][j]
#belgini o‘chirish
                        replacement = 1 + lev[i - 1][j - 1]
#belgini almashtirish
```

```
                                    # Eng foydali o‘zgarishni
tanlash:
                                    lev[i][j] = min(insertion,
deletion, replacement)

            #Levenshtein Masofasini ekranga chiqar
            print(“Levenshtein Masofasi: “, lev[len(a)][len(b)])
```

Kosinus o‘xshashligi – bu ikki obyekt bir-biriga qanchalik o‘xshashligini aniqlash usuli. Bunda obyektlarning maxsus xususiyatlariga va ushbu xususiyatlarning qancha miqdoriga ega ekanligi qaraladi. Ikki obyekt o‘xshash, agar ularning xususiyatlari va mos miqdorlari o‘xshash bo‘lsa. Kosinus o‘xshashligining qiymatini hisoblashda chiziqli algebradan foydalaniladi. Demak, obyektlarning maxsus qiymatlari ularga mos son ko‘rinishiga olib kelinadi hamda obyekt esa ushbu sonlar umumiylashgan maxsus vektor sifatida qaraladi va kosinus o‘xshashligi vektorlar orasidagi burchakning kosinusi orqali aniqlanadi (xususiyatlar soniga qarab bu vektorlar n o‘lchamli fazoda bo‘lishi mumkin). Bundan kelib chiqadiki, kosinus o‘xshashligi qiymati doimo [-1,1] oraliqqa tegishli bo‘ladi. Kosinus o‘xshashligi 0ga teng bo‘lganda (burchak 90) hech qanday o‘xshashlik topilmaydi, kosinus o‘xshashligi 1ga yaqinlashgan sari vektorlar orasidagi burchak kichrayadi va o‘xshashlik ortadi, aksincha burchak ortgan sari kosinus o‘xshashligi qiymati kamayadi, natijada ular o‘zaro farqli ekanligi aniqlanadi. Kosinus o‘xshashligi murakkablik darajasi yuqori bo‘lmaganligi va muammoga tatbiq qilinish jarayoni oson bo‘lganligi bois turli sohalarda qo‘llanilib keladi [https://en.wikipedia.org].

Matnlarning (hujjatning) o‘xshashligini aniqlashda yuqoridagi berilgan usullarga nisbatan ancha samaraliroq usul hisoblanadi. Bundan tashqari matnlar o‘xshashligini tekshirish plagiatni aniqlash hamda oldini olishda ham keng foydalaniladi. Bu esa mualliflik huquqini himoya qilish uchun ham samarali usul sifatida ko‘rilishi mumkin. Maktab, litsey, kollej va universitet talabalarining vazifalar va imtihonlar jarayonida topshirgan ishlaridan ularning bu vazifalarga yondashuvini aniqlash mumkin bo‘ladi. Kosinus o‘xshashligi ma’lumotlarni qazib olish, axborot qidirish va matnni moslashtirish jarayonlarida ham qo‘llaniladi. Vektorlar o‘zgaruvchining xususiyatlariga tayinlangandan so‘ng, o‘lchov obyektlar orasidagi o‘xshashlikni tushunish uchun qimmatli vositaga aylanadi. Ma’lumotni qazib olish - bu ma’lumotlarni tahlil qilish orqali biznes muammolarini hal qilishga yordam beradigan

namuna va munosabatlarni aniqlash uchun katta ma'lumotlar to'plamini saralash jarayonidir. Ma'lumotni qazib olish texnikasi va vositalari korxonalarga kelajakdagi tendensiyalarni bashorat qilish va ko'proq ma'lumotga ega biznes qarorlar qabul qilish imkonini beradi. Axborot qidirish - bu hujjatdagi ma'lumotlarni qidirish, hujjatlarning o'zini qidirish, shuningdek, ma'lumotlarni tavsiflovchi metama'lumotlarni, matnlar, tasvirlar yoki tovushlar ma'lumotlar bazalaridan qidirish haqidagi fandir.

Xususiyatlari. Kosinus o'xshashligining eng diqqatga sazovor xususiyati shundaki, u individual vektor o'lchamlarini mutlaq emas, balki nisbiy taqqoslashni aks ettiradi. Har qanday konstanta a va vektor V uchun V hamda aV vektorlari maksimal darajada o'xshashdir. Shunday qilib, o'lchov chastotasi mutlaq qiymatlardan muhimroq bo'lgan ma'lumotlar uchun mos keladi. Biroq, Jensen-Shannon, SED va uchburchak divergensiya kabi axborot nazariyasiga asoslangan so'nggi o'lchovlar hech bo'lmaganda ba'zi kontekstlarda yaxshilangan vaziyatda barqaror semantikaga ega ekanligi ko'rsatiladi.

Kosinus o'xshashligi Evklid masofasi bilan bog'liq ekanini ham ta'kidlab o'tish lozim. Vektorlar orasidagi Evklid masofasi, ular ichidagi kvadrat qiymatlarning birlik yig'indisining normallashtirilgan holati sifatida aniqlanadi. [Connor, 2016]

Kosinus o'xshashligi quyidagi maxsus matematik formula orqali aniqlanadi.

$$\text{cosine similarity} = S_C(A,B) := \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

5-rasm. Kosinus o'xshashligini aniqlovchi matematik formula

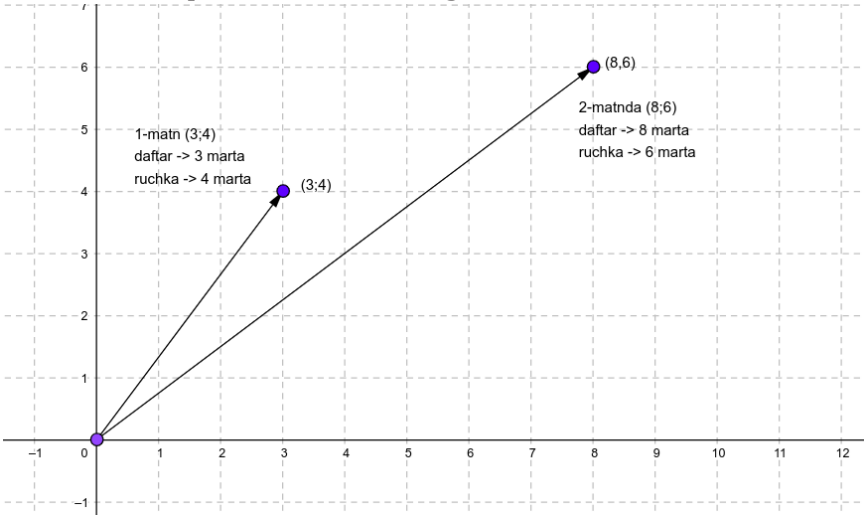
Quyida matnlar faqat 2 so'zdan tashkil topgan 2 xil matn va ularda qatnashgan so'zlarning soniga nisbatan 2 o'lchamli dekart koordinatalar sistemasida vektorlar sifatida tasvirlangan (berilgan misol tasavvur qilish va tasvirlash oson bo'lishi uchun tanlangan).

1- matn: {ruchka, daftar, daftar, ruchka, daftar, ruchka, ruchka}

2- matn: {daftar, ruchka, daftar, daftar, ruchka, daftar, daftar, ruchka, ruchka, ruchka, ruchka, daftar, daftar, daftar} dan tashkil topgan bo'lsin.

So'zlarni matnda uchragan o'rinlariga nisbatan mos sonlar bilan almashtiramiz va maxsus vektorlarni hosil qilib olamiz.

1- matnda 3 ta daftar va 4 ta ruchka uchragani bois, daftar 3 va ruchka so‘zi esa 4 soni bilan almashtiriladi hamda (3;4) nuqtasi hosil qilinadi. Xuddi shunday usulda 2-matndan mos ravishda (8;6) nuqtasi hosil qilinadi. So‘ngra koordinatalar boshiga nisbatan vektorlar hosil qilinib, ular orasidagi burchak kosinusi hisoblanadi.



Kosinus o‘xshashligi = $\frac{3*8+4*6}{\sqrt{3^2+4^2}*\sqrt{8^2+6^2}} = \frac{24}{25} = 0.96$ ga teng.

Bundan tashqari ko‘proq so‘zlardan tashkil topgan matnlar orqali so‘zlarning takrorlanish darajalarini quyidagicha jadvalga mos ravishda joylashtirib chiqiladi.

Hosil bo‘lgan so‘zlardan tashkil topgan setlar hosil qilinadi hamda ular orqali umumiy kalit so‘zlarga ega to‘plam yasaladi. Qiymatlarni maxsus vektorlarga aylantirish orqali yuqorida berilgan formuladan foydalanib kosinus o‘xshashligi hisoblab chiqiladi [https://www.itl.nist.gov].

Hujjat	jamoa	murabbiy	jarima	futbol	basketbol	zarba	g’alaba
Hujjat 1	1	1	0	2	1	0	2
Hujjat 2	2	3	0	3	2	1	3
Hujjat 3	1	2	1	2	3	1	0
Hujjat 4	0	2	3	3	1	1	2

6-rasm. Kosinus o‘xshashligi jadvali

“Yumshoq” kosinus o‘lchovi - ikki vektor orasidagi berilgan obyektlarning xususiyatlar juftligi o‘rtasidagi o‘xshashlikni hisobga oladi. An’anaviy kosinus o‘xshashligi vektor fazo modelining (VFM) xususiyatlarini mustaqil yoki butunlay boshqacha hisoblaydi, yumshoq kosinus o‘lchovi esa VFMdagi xususiyatlarning

o‘xshashligini hisobga olishni taklif qiladi, bu an’anaviy kosinus o‘xshashligiga nisbatan qo‘shimcha ma’lumotlar talab qiladi [Connor, 2016].

Masalan, tabiiy tilni qayta ishlash (NLP) sohasida so‘zlarning xususiyatlari o‘rtasidagi o‘xshashlik juda intuitivdir. Shuning uchun ulardan qo‘shimcha ma’lumotlar talab qilinishi aniqlik darajasini oshirishga yordam beradi. Misol uchun, “misol” va “boshqotirma” so‘zlari tarkiban turli xil so‘zlar va shuning uchun VFMning turli nuqtalarida ko‘rsatiladi. Biroq ular ma’no jihatidan bir-biriga bog‘liqdir.

Yumshoq kosinusni hisoblash uchun s o‘zgaruvchi tanlanadi va matritsasi xususiyatlari orasidagi o‘xshashlikni ko‘rsatish uchun ishlatiladi. N o‘lchamli fazoda a va b vektorlari orasida yumshoq kosinus o‘xshashligi quyidagicha hisoblanadi:

$$\text{soft\_cosine}_1(a, b) = \frac{\sum_{i,j}^{N} s_{ij} a_i b_j}{\sqrt{\sum_{i,j}^{N} s_{ij} a_i a_j} \sqrt{\sum_{i,j}^{N} s_{ij} b_i b_j}},$$

7-rasm. Formula

$s_{(i,j)} = o'xshashlik(xususiyat_i, xususiyat_j)$ agar xususiyatlar o‘rtasida o‘xshashlik bo‘lmasa 0ga, aks holda 1ga teng. Berilgan tenglama an’anaviy kosinus o‘xshashlik formulasiga ekvivalentdir [Sidorov, Velasquez, Stamatatos, 2013, 1].

Psevdokodi

```
function CosineDist(a,b):
X_list <- TOKENIZE(LOWERCASE(a));
Y_list <- TOKENIZE(LOWERCASE(b));
X_set <- REMOVE_STOPWORDS(X_list);
 Y_set <- REMOVE_STOPWORDS(Y_list) ;
l1 = CREATE_VECTOR(X_set);
l2 = CREATE_VECTOR(Y_set);
return COSINE(l1,l2)
end
```

Kosinus o‘xshashligining python dasturlash tiliga tatbig‘i

```
l1 =[];l2 =[]
# berilgan gaplar to‘plamlari
X_set = {‘men’, ‘har’, ‘kuni’, ‘maktabga’, ‘boraman’}
Y_set = {‘maktabga’, ‘bugun’, ‘men’, ‘barvaqt’, ‘bordim’}
# ikkala satrning kalit so‘zlarini o‘z ichiga olgan
```

to‘plam hosil qilamiz

```
rvector = X_set.union(Y_set)
for w in rvector:
    if w in X_set: l1.append(1) # maxsus vektor yaratamiz
    else: l1.append(0)
    if w in Y_set: l2.append(1)
    else: l2.append(0)
    c = 0

# kosinus formulasi
for i in range(len(rvector)):
    c+= l1[i]*l2[i]
    cosine = c / float((sum(l1)*sum(l2))**0.5)
print("kosinus o‘xshashligi: ", cosine)
```

Natijada: kosinus o‘xshashligi 0.4

Xulosa

So‘zlar orasidagi masofa qiymatini tavsiflovchi bir qancha qiymatlar mavjud. Ulardan Hamming, Levenshteyin masofalari va Kosinus o‘xshashligini yuqorida ko‘rib chiqqan holda shuni xulosa qilish mumkinki, har bir nazariyada kelib chiqadigan natija hamda ularning samaradorligi orqali o‘zining ishlatilish o‘rnini aniqlash mumkin. Aynan bir xil belgilar bilan boshlanuvchi so‘zlarni tekshirgan vaziyatda, Hamming masofasi boshqalarga nisbatan tabiiy tilni qayta ishlash jarayonida soddaroq hamda samaraliroq bo‘ladi. Biroq, satrlar hajmi ortishi hamda farq qiluvchi nuqtalar o‘zgarishi bilan Levenshteyin masofasi samaradorlik darajasi yuqoriga ko‘tariladi. Bundan tashqari, yuqorida berilgan nazariyalar orasida, katta hajmdagi matn yoki hujjatlar bilan ishlagan vaziyatda Kosinus o‘xshashligi eng munosib tanlov bo‘ladi. Demak, Hamming masofasini boshlang‘ich asos deb qaraladigan bo‘lsa, Levenshteyin masofasini uning so‘zlarga nisbatan mukammalroq vaziyati, shu bilan birgalikda, Kosinus o‘xshashligi esa so‘zlar jamlanmasi bo‘lgan matnlar yoki hujjatlar bilan ishlash uchun eng munosib nazariya sifatida qaraladi.

Adabiyotlar:

Waggener Bill. Pulse Code Modulation Techniques. Springer. p. 206. ISBN. Retrieved 13 June, 2020.

Robinson, Derek J. S. (2003). An Introduction to Abstract Algebra. Walter de Gruyter. pp. 255–257.ISBN.

Levenshteyin, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady.

Levenshteyin Distance Computation by Sergey Grashchenko November 16, 2022. https://www.baeldung.com/cs/Levenshteyin-distance-computation

Cosine similarity https://en.wikipedia.org/wiki/Cosine_similarity

Connor, Richard (2016). A Tale of Four Metrics. Similarity Search and Applications. Tokyo: Springer.

Cosine distance, cosine similarity, angular cosine distance, angular cosine similarity. https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/cosdist.htm.

Understanding cosine similarity and its application. Richmond Alake Sep 15,2020. Connor, Richard (2016). A Tale of Four Metrics. Similarity Search and Applications. Tokyo: Springer.

Sidorov, Grigori; Velasquez, Francisco; Stamatatos, Efstathios; Gelbukh, Alexander; Chanona-Hernández, Liliana (2013). Advances in Computational Intelligence. Lecture Notes in Computer Science. Vol.7630. LNAI 7630. pp.1–11.

The use of word distance algorithms in natural language processing

Nizomaddin Xudayberganov[1]
Shaxboz Khasanov[2]

Abstract:

Distance between words - can be characterized according to different qualities of words. Different algorithms are proposed in the direction of natural language processing to determine their similarity according to their component. These algorithms differ from each other in their places of use and performance. It is mainly used in areas such as spell check, speech recognition and plagiarism detection. This article analyzes the algorithms used to determine the similarity distance of more than one word and their characteristics.

[1]*Xudayberganov Nizomaddin Uktamboy o'g'li* – Teacher of the Department of Computer Linguistics and Digital Technologies of Tashkent State University of Uzbek Language and Literature named after Alisher Navoi.
E-mail: nizomaddin@navoiy-uni.uz
ORCID: 0000-0002-6213-3015

[2]*Hasanov Shaxboz G'olibjon o'g'li* – student, The INHA University in Tashkent.
E-mail: shaxbozhsh@gmail.com
ORCID: 0000-0002-2368-2863
For reference: Xudayberganov N., Khasanov Sh. 2022. "The use of word distance algorithms in natural language processing". *Uzbekistan: language and culture. Applied philology*. 2 (5): 69-83.

Key words: *Hamming distance, Levenshteyin distance, Cosine similarity, similarity table, coding theory*.

References:

Waggener Bill. Pulse Code Modulation Techniques. Springer. p. 206. ISBN. Retrieved 13 June, 2020.

Robinson, Derek J. S. (2003). An Introduction to Abstract Algebra. Walter de Gruyter. pp. 255–257.ISBN.

Levenshteyin, Vladimir I. (February 1966). "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady.

Levenshteyin Distance Computation by Sergey Grashchenko November 16, 2022. https://www.baeldung.com/cs/Levenshteyin-distance-computation

Cosine similarity https://en.wikipedia.org/wiki/Cosine_similarity

Connor, Richard (2016). A Tale of Four Metrics. Similarity Search and Applications. Tokyo: Springer.

Cosine distance, cosine similarity, angular cosine distance, angular cosine similarity. https://www.itl.nist.gov/div898/software/dataplot/refman2/auxillar/cosdist.htm.

Understanding cosine similarity and its application. Richmond Alake Sep 15,2020. Connor, Richard (2016). A Tale of Four Metrics. Similarity Search and Applications. Tokyo: Springer.

Sidorov, Grigori; Velasquez, Francisco; Stamatatos, Efstathios; Gelbukh, Alexander; Chanona-Hernández, Liliana (2013). Advances in Computational Intelligence. Lecture Notes in Computer Science. Vol.7630. LNAI 7630. pp.1–11.