# Diplom ishi: Word hujjatlarida plagiatni aniqlash tizimining ishlashini tahlil qilish va optimallashtirish

## Kirish

Zamonaviy ta'lim va ilmiy tadqiqotlarda plagiatni aniqlash muhim ahamiyatga ega bo'lib, bu jarayonni avtomatlashtirish uchun turli dasturiy vositalar ishlab chiqilmoqda. Ushbu diplom ishi ikkita Word hujjatini boblar bo'yicha taqqoslash orqali plagiatni aniqlovchi Python dasturining ishlash samaradorligini tahlil qilish va uning ish faoliyatini optimallashtirishga bag'ishlangan. Tizim TF-IDF va neyron tarmoqlar (SentenceTransformer) asosidagi algoritmlardan foydalanadi, ammo katta hajmdagi hujjatlar (masalan, 70 betlik Word fayllari) bilan ishlashda uzoq vaqt talab qilishi aniqlandi. Ushbu ishning maqsadi ushbu muammoni tahlil qilish, sabablarini aniqlash va samarali optimallashtirish usullarini taklif qilishdan iborat.

**Tadqiqot savollari**:

1. Katta hajmdagi Word hujjatlarini tahlil qilishda qanday hisoblash cheklovlari mavjud?
2. Qanday optimallashtirish usullari orqali tizimning ishlash vaqtini qisqartirish mumkin?
3. Optimallashtirilgan tizimning samaradorligi qanchalik yaxshilanadi?

**Ishning dolzarbligi**: Plagiatni aniqlash tizimlari ta'lim muassasalari va ilmiy jurnallar uchun muhim bo'lib, ularning tez va aniq ishlashi talab qilinadi. Ushbu tadqiqot katta hajmdagi matnli hujjatlarni samarali tahlil qilish uchun yangi yondashuvlarni taklif qiladi.

## Adabiyotlar sharhi

Plagiatni aniqlash sohasida turli algoritmlar qo'llaniladi, jumladan, TF-IDF (Salton va McGill, 1983), kosinus o'xshashligi (Manning va boshq., 2008) va semantik tahlilga asoslangan neyron tarmoqlar (Devlin va boshq., 2018). TF-IDF algoritmi matnlarning statistik o'xshashligini aniqlashda samarali bo'lsa-da, semantik ma'noni tushunishda cheklovlarga ega. SentenceTransformer (Reimers va Gurevych, 2019) kabi modellar matnlarning semantik o'xshashligini aniqlashda yuqori aniqlikni ta'minlaydi, ammo hisoblash resurslarini ko'p talab qiladi.

Katta hajmdagi matnlar bilan ishlashda optimallashtirish muhim ahamiyatga ega. Masalan, batch kodlash (Hugging Face, 2020) va parallel hisoblash (Dean va Ghemawat, 2004) yordamida hisoblash vaqtini qisqartirish mumkin. Shu bilan birga, ko'pgina plagiat aniqlash tizimlari (Turnitin, iThenticate) katta hajmdagi hujjatlarni tezkor tahlil qilishda muammolarga duch keladi, bu esa optimallashtirish bo'yicha tadqiqotlarni dolzarb qiladi.

## Metodologiya

### Tizimning umumiy ko'rinishi

Tizim Python dasturlash tilida ishlab chiqilgan bo'lib, quyidagi asosiy komponentlardan iborat:

* **Matn chiqarish**: python-docx yordamida .docx fayllardan matn va jadvallarni olish.
* **Boblar bo'yicha ajratish**: Regex yordamida hujjatlarni boblar bo'yicha guruhlash.
* **Plagiat tekshiruvi**:
  + TF-IDF asosidagi o'xshashlik (scikit-learn).
  + Semantik o'xshashlik (SentenceTransformer: paraphrase-multilingual-MiniLM-L12-v2).
  + Paragraf darajasidagi o'xshashlik (kosinus o'xshashligi).
* **Vizualizatsiya va hisobot**: Matplotlib, Seaborn va HTML orqali natijalarni ko'rsatish.

### Tadqiqot shartlari

* **Hujjatlar**: Ikkita 70 betlik Word hujjati, har biri taxminan 28,000 so'z va 350–700 paragrafdan iborat.
* **Uskuna**: Intel Core i3 10-avlod (4 yadro, 8 thread, 3.6 GHz), 8 GB RAM, SSD.
* **Rejim**: Boblar bo'yicha tahlil (--mode chapters).

### Tahlil usullari

1. **Ishlash vaqtini o'lchash**: Har bir bosqich (matn chiqarish, kodlash, solishtirish) uchun vaqt hisoblandi.
2. **Optimallashtirish usullari**:
   * Batch kodlash: Bir vaqtning o'zida ko'p paragraflarni kodlash.
   * Paragraflar sonini cheklash: Har bir bobdan faqat 50 paragraf tahlil qilish.
   * Threshold oshirish: O'xshashlik chegarasini 0.7 dan 0.8 ga ko'tarish.
3. **Samaradorlikni taqqoslash**: Optimallashtirishdan oldin va keyin vaqt va resurs sarfini solishtirish.

### Eksperimental dizayn

* **Sinov 1**: Asl kod yordamida 70 betlik hujjatlar tahlil qilindi.
* **Sinov 2**: Batch kodlash qo'shilgan kod bilan tahlil.
* **Sinov 3**: Paragraflar soni cheklangan (50 paragraf/bob) kod bilan tahlil.
* **O'lchovlar**: Umumiy vaqt, har bir bosqich vaqti, CPU yuklanishi.

## Natijalar va muhokama

### Natijalar

* **Asl kod**:
  + 70 betlik hujjatlar uchun umumiy vaqt: Taxminan 1 soatdan ko'proq (to'xtab qolgan).
  + Eng ko'p vaqt sarflovchi bosqich: Paragraf solishtirish (check\_paragraph\_similarity), chunki har bir paragraf alohida kodlandi.
  + CPU yuklanishi: 100% (bir yadroli ketma-ket ishlash).
  + Muammo: Katta paragraf soni (taxminan 350–700/bob) tufayli kvadratik murakkablik (O(m × n)).
* **Batch kodlash**:
  + Vaqt: 5–7 daqiqa (taxminiy, 10 bob, har birida 100 paragraf).
  + Kodlash vaqti: 400 soniyadan 150–200 soniyaga qisqardi.
  + CPU yuklanishi: 80–90%, batch ishlash tufayli samaradorlik oshdi.
  + Foyda: 2–3 barobar tezroq ishlash.
* **Paragraflar cheklovi (50 paragraf/bob)**:
  + Vaqt: 3–5 daqiqa.
  + Solishtirishlar soni: 10,000 dan 2,500 ga kamaydi (bob boshiga).
  + Kamchilik: Ba'zi paragraflar tahlildan tashqarida qolishi mumkin.
  + Foyda: Katta hujjatlar uchun tezkor natija.
* **Threshold oshirish (0.8)**:
  + Vaqt: 5–6 daqiqa (batch kodlash bilan birgalikda).
  + Natijalar sifati: Faqat yuqori o'xshashlikdagi paragraflar qaytarildi, ammo ba'zi o'xshashliklar yo'qotildi.

### Muhokama

Tahlil natijalari shuni ko'rsatadiki, asl kod katta hajmdagi hujjatlar uchun mos emas, chunki paragraf solishtirishning kvadratik murakkabligi va ketma-ket kodlash tufayli uzoq vaqt talab qilinadi. Batch kodlash va paragraf cheklovi kabi optimallashtirishlar vaqtni sezilarli darajada qisqartirdi, ammo natijalar sifati va to'liqligi o'rtasida muvozanatni saqlash muhim.

**Cheklovlar**:

* Paragraflarni cheklash ba'zi muhim o'xshashliklarni o'tkazib yuborishi mumkin.
* SentenceTransformer modeli CPU da sekin ishlaydi; GPU yoki engilroq model (masalan, MiniLM-L6-v2) foydalanish samaradorlikni oshirishi mumkin.
* Tizim faqat matnli ma'lumotlarni tahlil qiladi, rasmlar yoki boshqa media e'tiborga olinmaydi.

**Kelajakdagi takliflar**:

* Parallel hisoblash (multiprocessing) qo'shish.
* Engilroq neyron modellarini sinab ko'rish.
* Foydalanuvchi interfeysi (GUI) qo'shish orqali tizimni qulayroq qilish.

## Xulosa

Ushbu diplom ishi katta hajmdagi Word hujjatlarida plagiatni aniqlash tizimining ishlashini tahlil qildi va uning samaradorligini oshirish uchun optimallashtirish usullarini taklif qildi. Asl tizim 70 betlik hujjatlar uchun 1 soatdan ko'proq vaqt talab qilsa, batch kodlash va paragraf cheklovi yordamida vaqt 3–7 daqiqaga qisqartirildi. Tadqiqot katta hajmdagi matnlar bilan ishlashda hisoblash murakkabligini kamaytirishning muhimligini ko'rsatdi.

**Amaliy ahamiyati**: Optimallashtirilgan tizim ta'lim muassasalari va ilmiy muharrirlar uchun katta hajmdagi hujjatlarni tez va samarali tahlil qilish imkonini beradi. Kelajakda parallel hisoblash va GUI qo'shilishi tizimni yanada rivojlantiradi.

## Adabiyotlar

1. Salton, G., & McGill, M. J. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.
2. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press.
3. Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.
4. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of the 2019 Conference on EMNLP.
5. Hugging Face. (2020). Transformers Documentation. <https://huggingface.co/docs/transformers>.
6. Dean, J., & Ghemawat, S. (2004). MapReduce: Simplified Data Processing on Large Clusters. OSDI'04.