Využitie LLM pre analýzu právnych dokumentov

Samuel Bagín

Úvod a motivácia

- **Problém**: Právne texty sú komplexné a sémanticky husté; samotné LLM trpia halucináciami a obmedzeným kontextom (daň, daň z príjmu, daň z nehnuteľnosti).
- **Riešenie**: Transformácia neštruktúrovaného textu do Znalostného grafu (KG).
- **Metóda**: Využitie GraphRAG (Graph Retrieval-Augmented Generation) pre presnejšie odpovede.
- **Cieľ**: Systém na automatickú extrakciu, prepojenie entít a sémantické vyhľadávanie v legislatíve.

Systémový prehľad

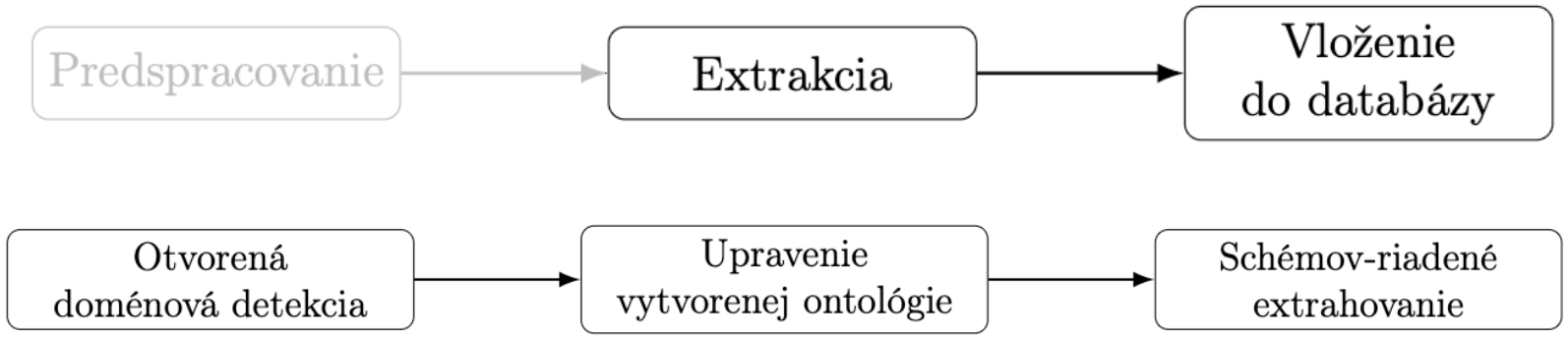
- **Backend**: Python, knižnica LangChain.
- **Modely:**
  - GPT-4o: Precízna extrakcia schémy a entít.
  - Claude 3.5 Sonnet: Generovanie Cypher dopytov (kódovanie).
  - Gemini 2 Flash: Klasifikácia a sumarizácia veľkého kontextu.
- **Databázy**:
  - Neo4j: Grafové úložisko vzťahov.
  - Neo4jVector: Vektorové embeddingy pre sémantickú podobnosť.
- **Embeddingy**: text-embedding-3-large (OpenAI).

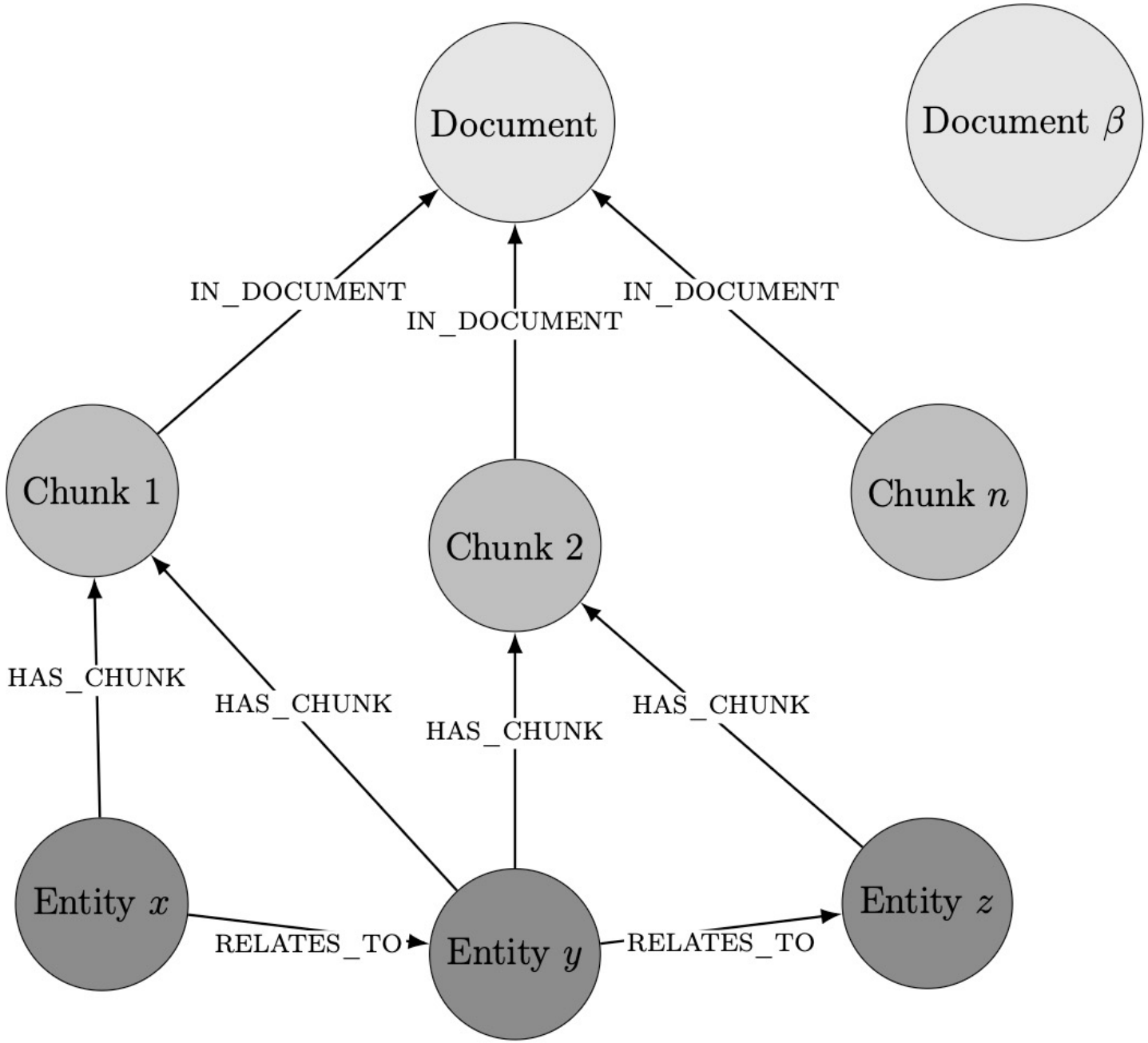
Spracovanie dokumentu I.

- **Klasifikácia** (Gemini): Automatické určenie kategórie (zmluva, zákon) a právneho odvetvia s percentuálnou istotou.
- **Segmentácia** (Chunking):
  - Využitie RecursiveCharacterTextSplitter.
  - Veľkosť: 1024 – 1200 tokenov s prekrývaním (overlap) 128 – 200 tokenov.
- **Štruktúra**: Prepojenie dokumentov s chunkmi cez vzťah `IN_DOCUMENT`.

Spracovanie dokumentu II.

- **Metóda**: Inšpirácia štúdiou Apple (Ontology-Guided Open-Domain Knowledge Extraction with LLMs).
- Tri kroky extrakcie:
  1. Otvorená detekcia (GPT-4o): Paralelné hľadanie všetkých entít/vzťahov (bohatosť dát).
  2. Úprava ontológie (Gemini): Mapovanie na preddefinované právne entity (konzistencia, eliminácia duplicít).
  3. Schémov-riadená extrakcia: Extrakcia dát podľa očistenej schémy.
- **Výsledok**: Presnosť extrakcie až 98,8 %.

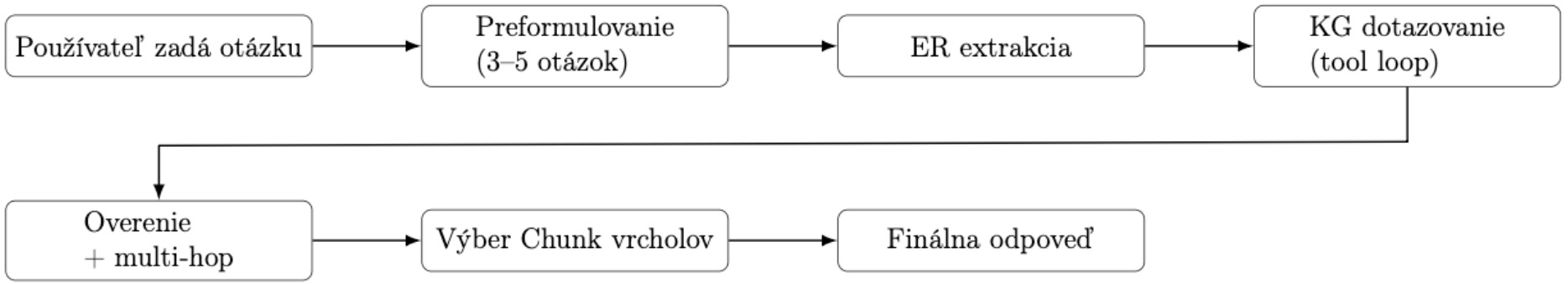



Document
Document β
IN_DOCUMENT
IN_DOCUMENT
IN_DOCUMENT
Chunk 1
Chunk 2
Chunk n
HAS_CHUNK
HAS_CHUNK
HAS_CHUNK
HAS_CHUNK
Entity x
Entity y
Entity z
RELATES_TO
RELATES_TO

```
},
{
  "n": {
    "identity": 1,
    "labels": [
      "__Entity__",
      "Organizational_structure"
    ],
    "properties": {
      "name": "Organizačná štruktúra podniku",
      "id": "1"
    },
    "elementId": "4:9510f823-c89e-450a-8ece-303fc7080585:1"
  },
  "r": {
    "identity": 1153329423420751873,
    "start": 1,
    "end": 2,
    "type": "INCLUDES",
    "properties": {},
    "elementId": "5:9510f823-c89e-450a-8ece-303fc7080585:1153329423420751873",
    "startNodeElementId": "4:9510f823-c89e-450a-8ece-303fc7080585:1",
    "endNodeElementId": "4:9510f823-c89e-450a-8ece-303fc7080585:2"
  },
  "m": {
    "identity": 2,
    "labels": [
      "__Entity__",
      "Employee"
    ],
    "properties": {
      "name": "zamestnanci",
      "id": "2"
    },
    "elementId": "4:9510f823-c89e-450a-8ece-303fc7080585:2"
  }
},
{
```


Hybridný GraphRAG

- **Preformulovanie**: LLM vytvorí 3–5 variácií otázky pre širší záber.
- Iteratívne dopytovanie:
  - Extrakcia entít z otázky.
  - Vyhľadanie podobných uzlov vo vektorovej databáze.
  - Cypher Agent: LLM generuje a spúšťa databázové dopyty iteratívne (tool loop).
- **Multi-hop Reasoning**: Hľadanie súvislostí "za roh" (napr. Osoba -> Zmluva -> Zákon).



Overenie a Generovanie

- **Kontrolný mechanizmus**: LLM posúdi, či sú dáta z grafu dostatočné.
- Kontextová syntéza. Odpoveď sa skladá z:
  - Znalostných podgrafov (štruktúrované dáta).
  - Textových chunkov (sémantický kontext).
  - Pôvodnej a preformulovanej otázky.
- **Gemini Flash**: Vďaka veľkému oknu spracuje všetky vstupy do finálnej právnej analýzy.

Validácia na literárnom diele

- **Prečo**: Jednoduchšie overenie faktov a komplexných vzťahov.
- **Efektivita**: Asynchrónne spracovanie znížilo čas z 12 min na 2 min (6x zrýchlenie).
- **Presnosť**: Napriek nekonzistentnosti v otvorenej doméne (5 rôznych vzťahov pre "lásku") systém vďaka grafu vždy našiel správnu odpoveď.
- **Aktuálny stav**: Prechod na reálne dáta zo Slov-Lex.

Príklad vytvorenej Cypher Query na otázku *"Which Characters who belong to the Capulet house are in love with a Character who belongs to the Montague house?"*:

```
MATCH (cap)-[b:BELONGS_TO]-(h:House)\nWHERE toLower(h.id) CONTAINS toLower('capulet') AND
(cap:Character OR cap:Person)\nMATCH (cap)-[r]-(m)\nWHERE type(r) IN ['LOVES','IN_LOVE_WITH','LOVE',
'EXPRESSES_FEELING_FOR','TENDER_LOVE','POTENTIAL_LOVE','WOO','LOVER'] AND (m:Character OR m:Person)\nMATCH
(m)-[bm:BELONGS_TO]-(h2:House)\nWHERE toLower(h2.id) CONTAINS toLower('montague')\nRETURN DISTINCT cap.id AS
capulet_character, labels(cap) AS capulet_labels, type(r) AS rel_type, m.id AS montague_character, labels(m) AS
montague_labels, properties(cap) AS cap_props, properties(m) AS mont_props\nLIMIT 25
```


```
"nodes_found": [
        "Juliet",
        "Romeo",
        "Capulet",
        "Montague"
      ],
      "relationships_found": [
        "Juliet -[LOVES]-> Romeo",
        "Juliet -[EXPRESSES_FEELING_FOR]-> Romeo",
        "Juliet -[LOVER]-> Romeo",
        "Juliet -[LOVE]-> Romeo",
        "Juliet -[IN_LOVE_WITH]-> Romeo",
        "Juliet -[BELONGS_TO]-> Capulet",
        "Romeo -[BELONGS_TO]-> Montague"
      ]
    },
```

Plán na letný semester

- Optimalizácia promptov.
- **Webová aplikácia**: Používateľské rozhranie pre vkladanie otázok a vizualizáciu nájdených grafov alebo **Jupyter Notebook**.
- Dokončiť implementáciu vyhľadávania a multi-hop reasoningu.

Ďakujem za pozornosť.