Algorithm 1: PageRank-stratified and QA-pair extraction

Input: Triples dump $\mathcal{D}$, subject type set T_S, object type set T_O, patterns $\mathcal{P} = \{(id_i, r_i, freq_i, MaxSW_i, AvgSW_i, MinSW_i)\}_{i=1}^{N}$, PageRank parameters *EDGE_CAP*, *DAMPING*, *MAX_ITER*, optional sample cap K.

Output: Two dataframes: QA_SW and QA_OS each stratified by cardinality bin and PageRank quartile.

1. Harvest typed entities:
   $S \leftarrow \{\, s \mid (s, \texttt{rdf:type}, t) \in \mathcal{D},\ t \in T_S \,\}$
   $O \leftarrow \{\, o \mid (o, \texttt{rdf:type}, t) \in \mathcal{D},\ t \in T_O \,\}$
2. Compute incoming-only PageRank:
   Build adjacency list $inAdj$ from first `EDGE_CAP` triples of $\mathcal{D}$
   Let $N = |inAdj|$ and initialize $pr_0(e) = 1/N$
for $t \leftarrow 1$ **to** *MAX_ITER* **do**
  foreach *node* v *in* $inAdj$ **do**
    $pr_t(v) \leftarrow (1 - \texttt{DAMPING})/N + \texttt{DAMPING} \sum_{u \in inAdj[v]} \frac{pr_{t-1}(u)}{\text{outdeg}(u)}$
  end
end
Set $PR(e) = pr_{\texttt{MAX\_ITER}}(e)$ for all e

3. Build subject-object mappings:
foreach $p = (id, r, sup, maxSO, avgSO) \in \mathcal{P}$ **do**
  Define $M_{SO}(p) : S \to 2^O$ by $M_{SO}(p)(s) = \{\, o \mid (s, r, o) \in \mathcal{D} \}$
  Define $M_{OS}(p) : O \to 2^S$ by $M_{OS}(p)(o) = \{\, s \mid (s, r, o) \in \mathcal{D} \}$
end

4. Stratify and sample by cardinality and PR quartile:
foreach *direction* $d \in \{SO, OS\}$ **do**
  foreach *pattern* $p = (id, r, f, MaxSW, AvgSW, MinSW) \in \mathcal{P}$ **do**
    Compute threshold $T \leftarrow \lceil (AvgSW + maxSW)/2 \rceil$
    Let $E = (M_d(p))$
    `// 4.1 Cardinality bins`
    Partition E into $E_{\text{single}} = \{e \mid |M_d(p)(e)| = 1\}$,
      $E_{\text{few}} = \{e \mid 2 \le |M_d(p)(e)| \le T\}$, $E_{\text{many}} = \{e \mid |M_d(p)(e)| > T\}$
    `// 4.2 PageRank quartiles`
    foreach *bin* $b \in \{single, few, many\}$ **do**
      Sort E_b by increasing $PR(e)$
      Split into four contiguous equal-size quartiles $Q_{b,1}, Q_{b,2}, Q_{b,3}, Q_{b,4}$
      `// 4.3 Sample  collect pairs`
      for $\ell \leftarrow 1$ **to** 4 **do**
        Let $S_{b,\ell} = Q_{b,\ell}$ (or a uniform random sample of size K if $|Q_{b,\ell}| > K$)
        Let $P_{b,\ell} = \{(e, o) \mid e \in S_{b,\ell},\ o \in M_d(p)(e)\}$
        Emit slice record $(id, b, \ell, |S_{b,\ell}|, |P_{b,\ell}|, P_{b,\ell}, d)$
      end
    end
  end
end

5. Build and write QA pairs:
foreach *emitted slice record* (id, b, ℓ, S, P, d) **do**
  Initialize empty list $\mathcal{Q}_d$
  foreach *entity* $e \in S$ **do**
    Let $O_e \leftarrow M_d(p)(e)$
    if $|O_e| = 1$ **then** choose template $T_{\text{sing}}(r)$
    else choose template $T_{\text{plur}}(r)$
    Form question $q \leftarrow T(\,\text{label}(e)\,)$
    Form answer $a \leftarrow \text{concat}\big(\text{label}(o) \mid o \in O_e\big)$
    Append (id, b, ℓ, e, q, a) to $\mathcal{Q}_d$
  end
  if $d = SO$ **then** Append $\mathcal{Q}_d$ to $\mathcal{Q}_{SO}$
  else Append $\mathcal{Q}_d$ to $\mathcal{Q}_{OS}$
end
Write $\mathcal{Q}_{SW}$ and $\mathcal{Q}_{OW}$

Algorithm 2: LLM QA Evaluation and Elasticity Estimation for Pattern-Aligned Questions

Input: QA dataset (e.g., QA_SW, QA_OW) containing questions, answers, and pattern metadata; Pretrained LLM checkpoint models; Prompt templates for zero- and few-shot settings; Similarity thresholds θ_m for metrics $m \in \{\text{exact}, \text{edit}, \text{jaccard}, \text{rougeL}, \text{BLEU}, \text{BERT}\}$

Output: Annotated dataframe with predicted answers, similarity scores, and verdicts per metric; Aggregated accuracy by pattern, cardinality class, and PageRank quartile; and Elasticity coefficients for each metric.

1. Model Setup:
Load LLM and tokenizer with 8-bit quantization and CPU offloading, Initialize text generation pipeline with max length, padding, and temperature settings.

2. Prompt Construction:
foreach *QA pair* (q, a) *in input file* **do**
| Format prompt using either zero-shot or few-shot template.
end

3. Batch Inference:
foreach *batch of prompts* **do**
| Run generation via model pipeline to produce predicted answers.
end

4. Similarity Scoring and Verdict Assignment:
foreach *prediction* p *and ground truth* g **do**
| Compute similarity scores:
| Exact match: $\texttt{exact\_match}(p, g)$
| Edit similarity: $\texttt{edit\_sim}(p, g)$
| Jaccard similarity: $\texttt{jaccard\_sim}(p, g)$
| ROUGE-L F1: $\texttt{rouge\_l}(p, g)$
| BLEU: $\texttt{bleu\_sim}(p, g)$
| BERT cosine: $\texttt{bert\_sim}(p, g)$
| For each metric m, assign verdict as:

$$\texttt{verdict}_m \leftarrow \begin{cases} \text{"unsure"} & \text{if } p = \text{"unsure"} \\ \text{"correct"} & \text{if score} \geq \theta_m \\ \text{"hallucination"} & \text{otherwise} \end{cases}$$

end

5. Aggregate Accuracy by Pattern and Quartile, Group predictions by `(pattern_id, predicate, freq, card_class[, pr_quartile])` and compute metric-wise accuracy:
$\texttt{acc}_m = \texttt{mean}(\texttt{verdict}_m = \text{"correct"}) \times 100$

6. Compute Elasticity Coefficients:
foreach *metric* m *and cardinality group* g **do**
| Fit linear regression:

$$\texttt{acc}_m = \alpha + \beta \cdot \log_{10}(\texttt{frequency}) + \varepsilon$$

| Store β as the elasticity of accuracy with respect to frequency for group g and metric m.
end

7. Write Results