**Prezentarea Soluției: Sistem de Generare Automată a Întrebărilor Inteligente**

## SOLUTII EXISTENTE (<https://drive.google.com/file/d/19gMnMliEqMGBmyGt-q1OtLk2cP7u-rqj/edit>) - Nikahat Mulla · Prachi Gharpure

## Neural network-based approaches Context general

Când vrei ca un model să „învețe” să pună întrebări pe baza unui text (și eventual a unui răspuns dat), trebuie să faci o transformare de tip:

**Text (context) + Raspuns => Intrebare**

De exemplu:

**Input (context + răspuns):**

Context: „Marie Curie a fost prima femeie care a primit un premiu Nobel.”  
Răspuns: „Marie Curie”

**Output (întrebarea generată):**

„Cine a fost prima femeie care a primit un premiu Nobel?”

Acest tip de sarcină — „secvență de text la secvență de text” — se potrivește perfect cu arhitectura **encoder–decoder (seq2seq)**.

## Arhitectura encoder–decoder (seq2seq)

Modelul are **două părți principale**:

### Encoder

* Primește textul de intrare (de exemplu, un paragraf sau o propoziție + răspunsul).
* Reprezintă acest text ca o **succesiune de vectori numerici (embedding-uri)**.
* Apoi trece aceste reprezentări printr-o rețea neuronală recurentă (de exemplu, LSTM sau GRU).
* Scopul encoderului este să **comprima întreaga informație a textului** într-o **reprezentare internă** (un vector de context).

🔸 În articolul menționat, autorii folosesc **două encodere**:

1. Unul care procesează **informația la nivel de propoziție** (sentence-level encoder).
2. Unul care procesează **informația combinată propoziție + paragraf** (paragraph-level encoder).

Ambele sunt **LSTM bidirecționale cu mecanism de atenție (BiLSTM + Attention)**.

**Decoder**

* Primește reprezentarea internă produsă de encoder.
* Începe să **genereze textul de ieșire pas cu pas** (adică întrebarea).
* La fiecare pas, el decide **care cuvânt urmează**, folosind informația din encoder + cuvintele generate anterior.

## Rolul mecanismului de ****atenție (attention)****

Fără atenție, modelul ar trebui să rețină tot contextul într-un singur vector — lucru dificil pentru texte lungi.

➡️ Atenția permite decoderului să „se concentreze” pe **anumite părți ale textului de intrare** atunci când generează fiecare cuvânt al întrebării.

De exemplu:

* când generează cuvântul „Cine”, modelul se concentrează pe „Marie Curie”;
* când generează „a primit”, se concentrează pe „a fost prima femeie care a primit”.

Astfel, atenția îmbunătățește mult calitatea și relevanța întrebărilor generate.

**PROPUNEREA NOASTRA(Rule-based approaches)**

Ideea de bază

Scopul proiectului este crearea unui sistem care generează automat întrebări inteligente, nu doar le extrage din text. Sistemul utilizează reguli logice și o bază de cunoștințe bine structurată, fără a depinde de modele neuronale complexe.

# Componentele principale

## ****0. Extragerea parametrilor din prompt**** Se extrag folosind **regex**, pentru a identifica **subiectul/cursul** și **numărul de întrebări**.

## 1. Baza de cunoștințe (Knowledge Base)

Reprezintă un dicționar extins cu informații despre conceptele vizate: ce este fiecare problemă, din ce categorie face parte, ce strategii de rezolvare există și care este cea optimă.

Exemplu conceptual:  
  
• Problemă: n-queens  
• Categorie: search problem  
• Strategii posibile: backtracking, local search, constraint satisfaction  
• Strategie optimă: backtracking (cu MRV)  
• Cursuri in care apare notiunea: C1, C2

## 2. Reguli logice (Rule Base)

Regulile definesc legătura dintre categoria unei probleme și tipul de întrebare care poate fi generată. Ele stabilesc structura logică a întrebărilor și răspunsurilor posibile.

Exemple de reguli:  
  
• Dacă o problemă este de tip search problem → generează întrebări despre „strategii de căutare”.  
• Dacă o problemă este CSP → întreabă despre „asignare de variabile” sau „constraint propagation”.  
• Dacă o problemă este game theory → întreabă despre „există echilibru Nash?”.  
• Dacă o problemă are mai multe strategii → întreabă „care dintre următoarele este cea mai potrivită și de ce?”.

## 3. Șabloane de întrebări (Templates)

Regulile definesc tipul de întrebare, iar șabloanele definesc formularea concretă. Sistemul completează automat șabloanele cu date din baza de cunoștințe.

Exemple de șabloane:  
  
• „Pentru problema {problema}, care este cea mai potrivită strategie dintre {strategii}?”  
• „Pentru jocul dat, există echilibru Nash pur?”  
• „Care va fi valoarea din rădăcină dacă aplicăm strategia {strategie}?”

## 4. Generatorul de întrebări

Această componentă selectează o problemă, caută strategiile și regulile corespunzătoare, completează șablonul și generează întrebarea finală. Rezultatul este o propoziție coerentă, logică și personalizată.

## 5. Generatorul de răspunsuri model

Pe baza cunoștințelor, sistemul poate genera automat și răspunsul corect, permițând validarea automată a testelor.

Exemple:  
  
• „Pentru n-queens, cea mai potrivită strategie este backtracking, deoarece reduce spațiul de căutare prin MRV și forward-checking.”  
• „Pentru graph coloring, AC-3 combinat cu backtracking este eficient pentru propagarea constrângerilor.”

# Compararea răspunsurilor

Pentru verificarea automată a răspunsurilor utilizatorului, sistemul utilizează o măsură de similaritate semantică – cosine similarity – între răspunsul model și cel dat de utilizator. Astfel, sistemul evaluează apropierea logică și conceptuală, nu doar potrivirea textuală.