**Fact Sheet for EHRSQL-2024 Shared Task:**

1. **Team leader name : Sourav Bhowmik Joy**

* Username on Codalab: souravjoy
* Team leader affiliation: Shahjalal University Of Science & Technology, Sylhet
* Team leader email: sourav35joy80@gmail.com
* Name of other team members (and affiliation):
  + Rohan Redwan
  + Argha Pratim Saha
  + Minhaj Ahmed
  + Utsho Das
  + Partha Sarothi Bhowmik

Other Members Affiliation - Shahjalal University Of Science & Technology, Sylhet

* Team website URL (if any): Performing under the representational organization named AIBOT on Codalab

**II. Contribution**

* **Title of the contribution**

Automated SQL Query Generation from Natural Language Questions based on Electronic Health Records using Neural Networks and Database Schemas

* **Detailed method description**

**Preprocessing:** The approach we have chosen for this text-to-sql generation begins by importing necessary libraries and initializing the model and tokenizer from the Hugging Face transformers library. The model used is defog/sqlcoder-7b-2.

**Model Loading:** Depending on the available memory, it is decided whether to load the model in float16 or 8-bit precision.

**Prompt Setup:** A prompt string is defined, including placeholders for questions and the database schema.

**Question Iteration:** Next step includes iterating through a list of questions.

**Query Generation:** For each question, a SQL query is generated using the chosen model. The generate\_query() function takes a question as input, tokenizes it, and feeds it to the model for query generation. The generated SQL query is then processed to remove unnecessary characters and formatted using sqlparse.

**Result Handling:** If an appropriate response is not found, 'null' is assigned to the SQL query. Otherwise, the generated query is appended to a list.

* **Shared task results**
  + **rs0:** 14.14
  + **rs5:** -349.61
  + **rs10:** -713.37
  + **rsN:** -84885.86
* **Final Remaks**

**Pros**

* + **Efficiency:** By using pre-trained language models, the model can quickly generate SQL queries, improving productivity in database-related tasks.
  + **Memory Efficiency:** The model supports loading in float16 or 8-bit precision based on available memory, optimizing memory usage and reducing the risk of crashes in resource-constrained environments.

**Cons**

* + **Dependency on Pre-trained Models:** The model's performance heavily relies on the quality and domain coverage of the pre-trained language model it is based on. If the pre-trained model lacks domain-specific knowledge or has biases, it may affect the quality of generated queries.

**III. Additional method details**

* Did you use any pre-trained model?

Yes, we used the "defog/sqlcoder-7b-2" model.

* Did you use external data?

No, external data was not used.

* Did you perform any data augmentation?

No, data augmentation was not performed.

* At the test phase, did you use the provided validation set as part of your training set?

No, the provided validation set was not used as part of the training set.

* Did you use any regularization strategies/terms?

No

* Did you use handcrafted features?

No, handcrafted features were not used.

* Did you use any domain adaptation strategy?

No, domain adaptation strategy was not used

**IV. Code Repository**

Link for the github repository of the notebook implementing the model is shared below-

[nlpConference/FinalModel at master · joy-2019331037/nlpConference (github.com)](https://github.com/joy-2019331037/nlpConference/tree/master/FinalModel)