

**DATA CLEANING:**

First, we focused on data cleaning, removing inconsistencies, special characters, and trimming whitespace. We ensured that characters were UTF-8 encoded and escaped any special characters like quotes or backslashes that could interfere with JSON formatting.

We then performed data validation by testing JSON serialization to catch any errors and validated against a JSON schema if applicable. For complex data types, we ensured that nested structures within the entity column were correctly formatted for JSON or flattened where necessary.

These steps ensured that the entity column was fully compatible with JSON, allowing seamless serialization and integration with systems that rely on structured JSON data, all while preserving data integrity.

**Approach and Model Structure:**

We divided the whole problem into two subparts. The first part deals with some named entity extraction as per the given data and the second part deals with some important attributes’ extraction related to the involved patient.

**First Problem: Attribute Extraction**

**Regex-Based Extraction**

We modified the given data as per our requirement. Then we approached the attribute extraction with the aid of regex, since age and gender follow a particular specified format for their representation. We were successfully able to extract gender, but model failed poorly on other attributes. Then we shifted onto the transformer model T5(Text-to-Text transformer).

**T5 Transformer with Sliding Window**

Since the T5 transformer has a limitation of max permissible tokens (512) so we incorporated sliding window technique where in the max window size was 512 , with strides of 256. So, this way we achieved our goal of dividing the data into smaller processible chunks to predict the attributes. But this model sort of overfitted the data, so we moved onto the final approach.

**Summarization with Regex Extraction**

This time we used a summarization technique to avoid the max Length problem. We used Meta’s LLM (Facebook-Bert-large-CNN), which generated the summary of that text. Then we used regex to extract the attributes. This approach performed better than the above-mentioned approaches. Our model also outperformed the pretrained RAG (Retrieval-Augmented Generation).

**Second Problem: Named Entity Recognition and Classification (NERC)**

**T5 transformer:** Initially we trained the T5 transformer model on our data to predict the named entities without the prediction of their confidence scores. But then again the problem of input size came which was resolved in the further approaches.

**Token classification with BERT-token classifier:** We created a mapping for all the entities with their entity type e.g. Neuropathic Pain as Problem, X-ray as Procedure and so on for better Performance. Then we trained the model with Discharge Summary as input and Predicted s.no, Entity, entity type, span start, span end.

**Summarization+T5 model**: We used T5 Conditional Generator model for summarizing the Discharge Summary (so we have not to think about 512 token limitations).we predicted the summary in such a way that it contains only entities. Then we mapped these entities with the entity type that we have already stored in a map.in this we predicted s.no, Entity, entity type, span start, span end with much better accuracy.