# ConQueR: Contextualized Query Reduction using Search Logs

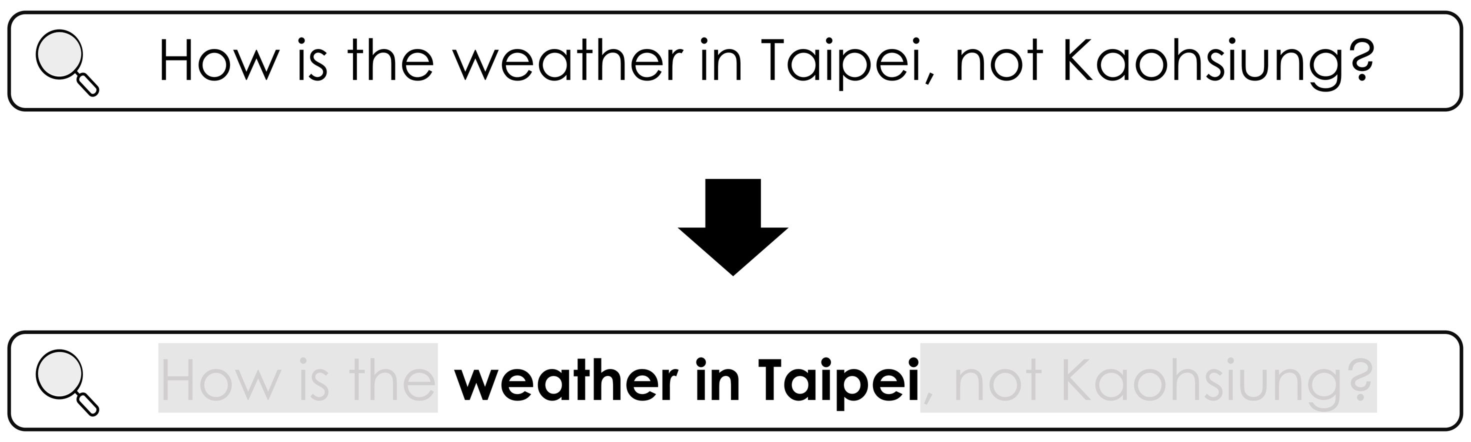

Hye Young Kim[1*], Minjin Choi[1*], Sunkyung Lee[1], Eunseong Choi[1], Young-In Song[2], Jongwuk Lee[1]

Sungkyunkwan University[1], Naver Corporation[2]

*equal contribution


## Query Reduction

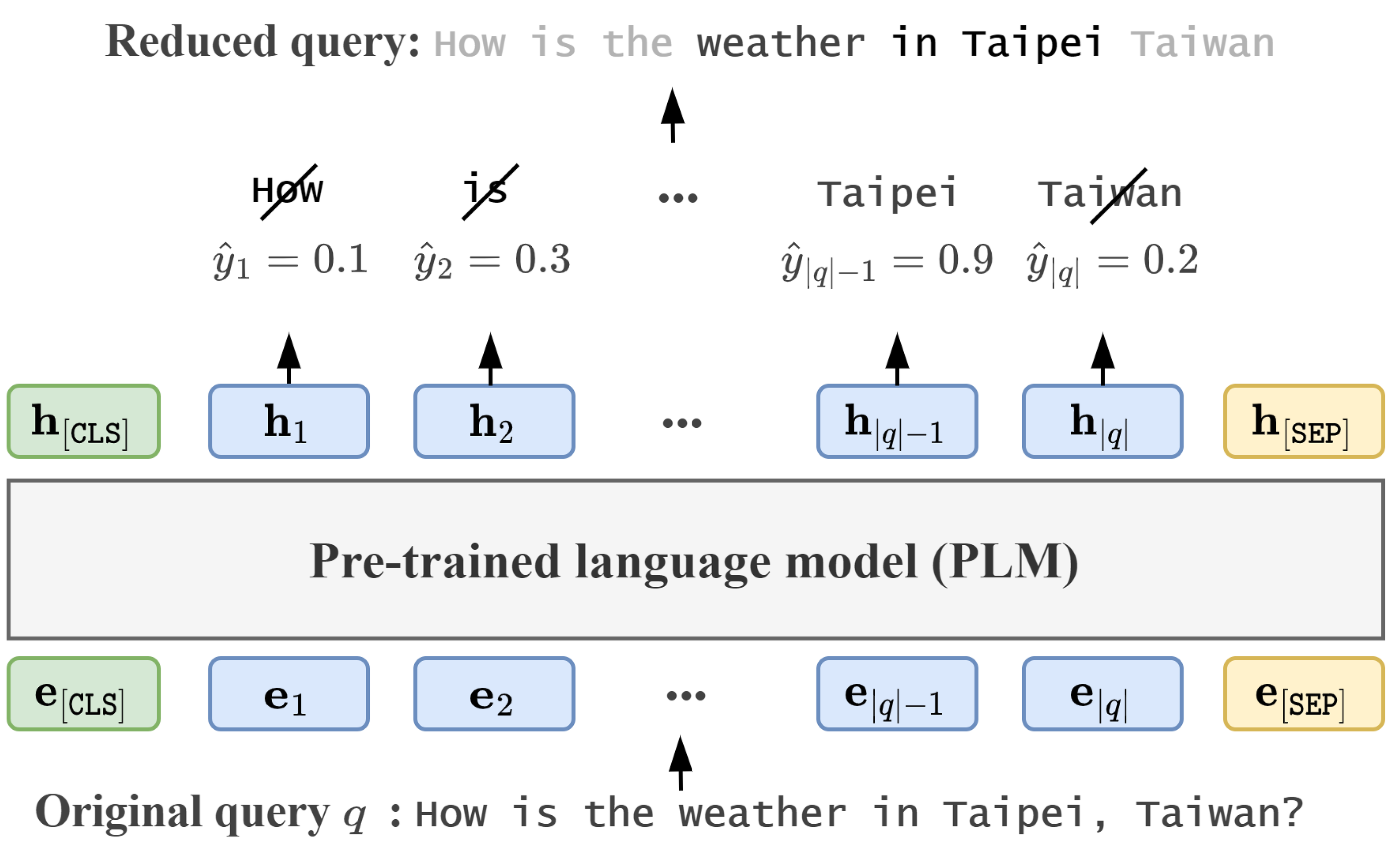
- **Query reduction** refers to **removing extraneous query terms** to obtain desired results.
- It is particularly beneficial for reducing long queries to reflect user intent better.




## Proposed Method: ConQueR

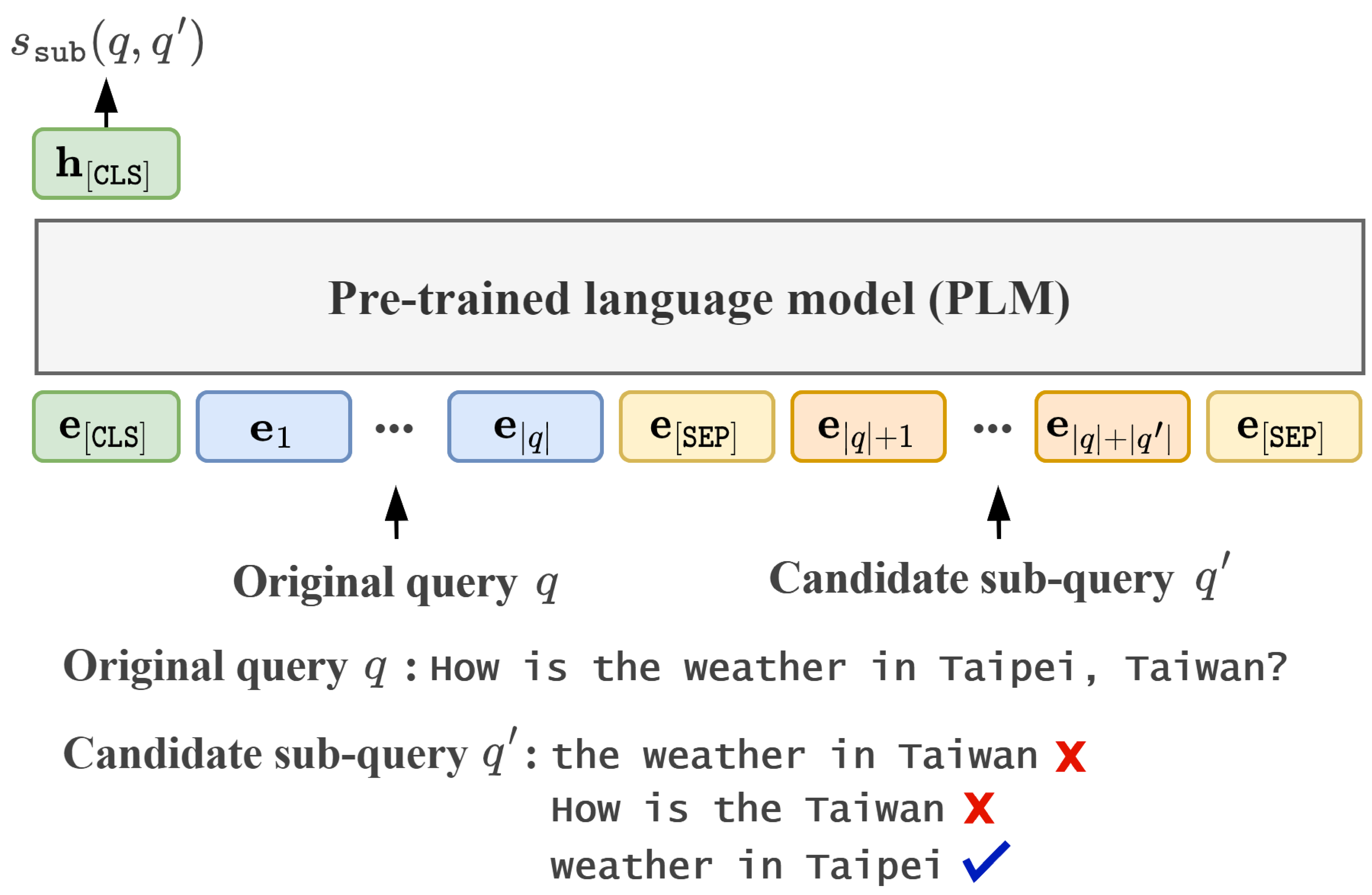
- **Contextualized Query Reduction (ConQueR)** exploits the contextualized representations of queries using PLMs with two different views; **core term extraction** and **sub-query selection**.

1) **The core term extraction method (ConQueR$_{core}$)** determines the importance of each term in the query. It effectively predicts important terms and drops all nonessential terms.




Core term extraction

2) **The sub-query selection method (ConQueR$_{sub}$)** takes the original query and a candidate sub-query as input to the transformer encoder. It determines whether the given sub-query is suitably reducing the original query.




Sub-query selection

- Finally, we **aggregate the two methods**, *i.e.*, ConQueR$_{core}$ and ConQueR$_{sub}$, because they complement each other by tackling query reduction at different levels.

## Dataset

- We collect the **search logs** from **a commercial web search engine** (https://www.naver.com), where **two queries are successive in a session**, and the latter is always a terminological subset of the former query.
- The **total number of query pairs is 239,976**, while the number of unique original queries is 104,002. We split them into three subsets, *i.e.*, a training set (80,202, 80%), a validation set (10,400, 10%), and a test set (10,400, 10%).

## Effectiveness on Search Logs

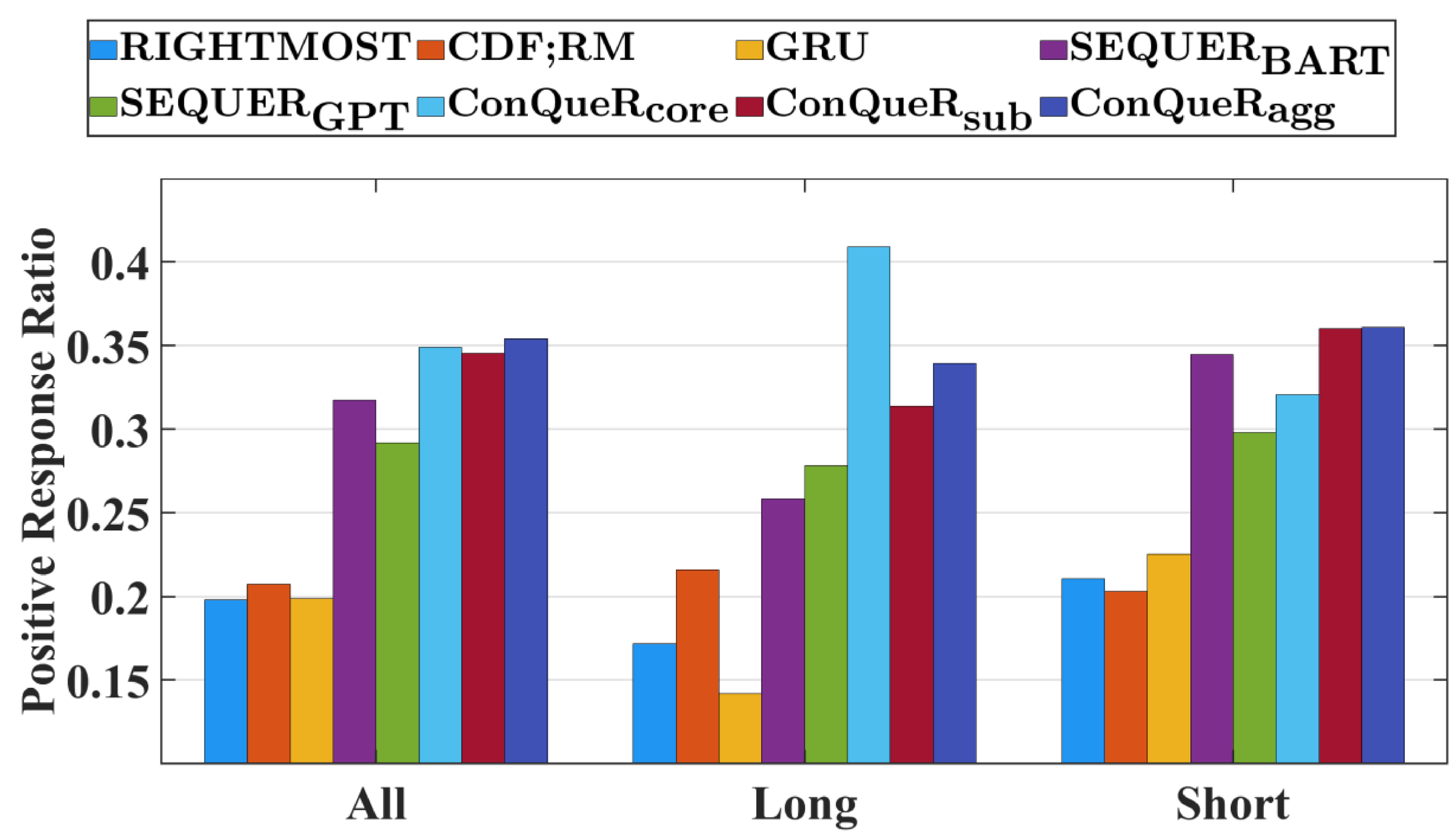
- Our proposed methods consistently outperform the baselines. ConQueR$_{agg}$ surpasses the best competing model by an 8.45% gain in EM and a 5.54% gain in Acc.
- ConQueR$_{agg}$ shows the best performance, **indicating that the two methods are well aggregated**.

| | Models | EM | Acc | F1 |
|---|---|---|---|---|
| Rule-based methods | LEFTMOST (LM) | 0.170* | 0.338* | 0.403* |
| | RIGHTMOST (RM) | 0.693* | 0.794* | 0.814* |
| | Deletion Frequency (DF); RM | 0.596* | 0.738* | 0.775* |
| | Conditional DF; RM | 0.697* | 0.814* | 0.836* |
| Neural methods | GRU | 0.752* | 0.882* | 0.893* |
| | SEQUER | 0.833* | 0.884* | 0.894* |
| Ours | ConQueR$_{core}$ | 0.892 | 0.928 | 0.932 |
| | ConQueR$_{sub}$ | 0.905 | 0.929 | 0.935 |
| | ConQueR$_{agg}$ | **0.911** | **0.934** | **0.940** |

Significant differences ($p$ < 0.01) between baselines and ConQueR$_{agg}$ are denoted with *.

## User Study

- About 35% of the users think that ConQueR$_{agg}$ correctly reduces the original queries.
- **ConQueR$_{core}$ and ConQueR$_{sub}$ perform better for each subset of *Long* and *Short***. This is because the longer the query, the more it needs to be reduced, and ConQueR$_{core}$ is more suited for multi-term deletions.




$$\textit{Positive Response Ratio} = \frac{\sum_u select_u}{\#users} \quad s.t. \quad select_u \in \{0,1\}$$

website: https://github.com/HyeYoung1218/ConQueR