|  |  |  |
| --- | --- | --- |
| **No** | **Issues** | **Date** |
| 1. | Read more related works and write the summarize | July plan |
| 2. | Applying multi queries shared computations   * Understanding shared computation of SeeDB * Implementing multi queries shared computation to our DiVE schemes * Compare the performance between shared computation in advanced and shared computation after sorted by diversity |  |
| 3. | Proposing another distance function of contextual similarity |  |
| 4. | Several previous works proposed algorithm to *trade-off* between relevance and diversity which claimed have better performance compare to Greedy and Swap.  Understanding those kind of approach and trying to implement to our work. |  |
| 5. | Diversifying chart types  Diversity in this work focus on only context of the A, F, M. We did not mention about the types of chart. However, different type of chart seems more interesting rather than only present all visualizations in bar chart. In fact, chart types also depends on the data itself. |  |
| 6. | Comparing among all subsets in the dataset instead of only compare each subset with the whole dataset.  Two types of query load in the experiments:   1. Compare between two subset (e.g., disease vs. no disease) – *targeted/ we know what we want to do* 2. Compare between one subset to whole dataset (get all subsets from dataset then compare to whole dataset)   Do we need to compare among all possible subsets in the dataset?  For instance, Flights dataset has attribute = ‘carrier’ and there are a lot of carrier (e.g., AA, US, XX, UU), each carrier can be one subset.  While we only compare between each subset to whole dataset, it only show the trend of each subset compared to whole dataset.  However, It seems interesting while we compare between each subset to others. But it increases the number of combination significantly.  It might rely mainly on user intention. For instance, user want to compare the performance of carrier AA vs XX in terms of arrival delay. Of course it depends on user want. We did not mention about this scenario in our experiments. This issue is also related to “User intention” issue below. |  |
| 7. | Handling several scenarios of user intention types:  There are three types of intention by task as follows (Toward Visualization Recommendation System’s paper):   1. Exploratory 2. Comparative 3. Targeted   We did not mention detail about these three types of user intention and how to handle it. |  |
| 8. | Using large number of views (e.g more than 2000 views). Observe if with this condition pruning based on prediction interval still feasible or not.  I really not sure pruning by prediction interval is feasible if our dataset has large number of views, for instance, more than 1000 views. By executing 50 samples of views with number of views more than 1000 does not make sense. While the number of views are very large (e.g., 1000 views), it seems approximation may be promising approach. |  |
| 9. | Using approximation (sampling) in case of very large dataset (has large number of attributes and subsets)  We did experiments mostly on heart dataset which has small size and small number of subset. Flights dataset has so many subsets but it seems not interesting. While we have large dataset such as Flights, even pruning has been used it still take a lot of time.  It seems the sampling approach or approximation is promising, instead of generating all possible views, we use sampling and show to the user the coverage of the results. |  |
| 10. | Doing experiment using non-flat data type (hierarchy OLAP)  This issue have been discussed since several months ago but not yet solved. |  |
| 11. | Auto-remove similar attributes by applying correlation  SeeDB has offline and online precomputation mechanism such as removing correlated attributes. While the dataset has so many attributes like superstore. It may has duplicate attributes such as “suburb” and “post code”. We should remove one of those attributes in advanced. We did not consider yet about this thing. |  |