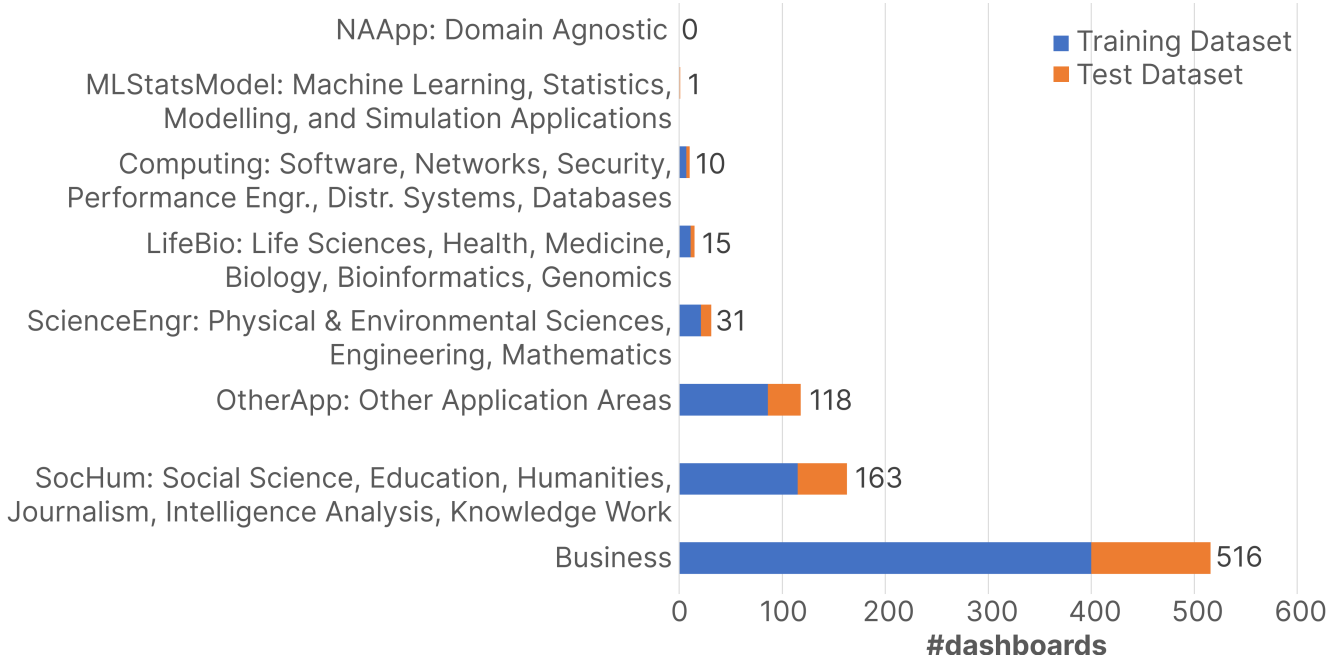
Supplementary Material of DMiner: Dashboard Design Mining and Recommendation

S1: Application

We group the different dashboards by labeling their applications to help identify the dashboard intention. VIS community has proposed seven keywords to categorize the application areas. When labeling, we found many dashboards focusing on business analysis, like sales and profit analysis. Thus, we add one more area denoted as *Business*. Then, we labeled each dashboard to one of these eight areas. To ensure the quality of labels, each dashboard is first labeled by two co-authors individually.
Conflicts are then discussed among authors until reaching a consensus.
The figure shown below presents the application distribution of the dashboards. We can observe that our collected dataset covers almost all application areas and focuses much on *Business* and *SocHum*.

This imbalanced distribution makes a comparative study between applications challenging.
We encourage the community to contribute more dashboards covering more areas to support future intention research.



S2: Data

- The hyperlinks and application domain of the dashboard workbooks are in the `info/dashboard_link_application.csv`
- The pairwise-view information of all dashboards is in the `info/dashboard_df`, with the field description in `info/field_description.pdf`.

S3: Mark Type

The mark types of DMiner consider both the mark types of Tableau and Vega-Lite, since Vega-Lite is a widely-used grammar of visualizations and they both have some problems. Specifically, Tableau treats *Box Plot* and *Error Bar* as *Circle*. Vega-Lite regards the scatterplot marked with filled squares as *Square* but empty squares as *Point*. In this case, DMiner considers both. The details are shown in the table below:

Mark Type in DMiner	**Mark Type in Vega-lite**	**Mark Type in Tableau**
Arc	Arc	Pie
Area	Area	Area
Bar	Bar	Bar
Box Plot	Box Plot	Circle
Circle	Circle	Circle
Density Map	Density Map	Density Map
Error Band	Error Band	-
Error Bar	Error Bar	Circle
Line	Line	Line
Map	geoshape	Map
Point	Point +Square	Shape
Rectangle	Rect	Square
Text	Text	Text

The specific Examples for each mark type are in the `info/mark_type_example.pdf`.

S4: Feature List

Before the detailed features, we introduce the definition of the **Source** view and **Target** view. Since the view pair (A, B) is directional, i.e., some features of (A, B) are different from that of (B, A). Thus, in a view pair (A, B), we call A a **Source** view, which means that it is a view to raise comparison or raise coordination; and we call B ad **Target** view, representing a view to receive comparison and coordination.

Single-view Features

- Data and Encoding Features

Feature Name	Meaning	Type
mark	the mark type	String
qkColNum	the number of used quantitative fields	Number
okColNum	the number of used fields ordinal fields	Number
nkColNum	the number of used fields nominal fields	Number
xColNum	the number of the fields used on X-axis of the source view	Number
yColNum	the number of the fields used on Y-axis of the source view	Number
colNum	the number of the used fields, and regarding *sum(col1)* is the same as *count(col1)*	Number
colNum_agg	the number of the used fields, and regarding *sum(col1)* is different from *count(col1)*	Number
aggColNum	the number of the fields using aggregation	Number
aggColNumPer	the percentage of the fields using aggregation to all used fields	Number
dataCoverage	the percentage of the used fields to all fields in the data table	Number

- Layout Arrangement Features

feature name	Meaning	type
grid*m_n*	boolean, whether the source view has an area on (m,n) grid larger than the threshold (like more than 90% of that grid)	Boolean
height	the **normalized** height of the view, the values are in [1, 2, 3, 4]	Number
width	the normalized width of the view	Number
x	the normalized x position of the view	Number
y	the normalized y position of the view	Number
size	the normalized size of the view	Number

Pairwise-view Features

- Data and Encoding Relationship Features

Feature Name	Meaning	Type
equalMark	whether two views are in the same mark	Boolean
withSameX	boolean, whether two views use the shared fields on X-axis	Boolean
withSameY	boolean, whether two views use the shared fields on Y-axis	Boolean
withSameSize	boolean, whether two views use the shared fields on size encoding	Boolean
withSameShape	boolean, whether two views use the shared fields on shape encoding	Boolean
withSameColor	boolean, whether two views use the shared fields on color encoding	Boolean
withSameEncoding	boolean, whether two views use the shared field on size, shape, or color encoding	Boolean
equalX	whether two views used exactly the same fields on X-axis	Boolean
equalY	whether two views used exactly the same fields on Y-axis	Boolean
equalXCol	whether two views use the same number of fields on X-axis	Boolean
equalYCol	whether two views use the same number of fields on Y-axis	Boolean
SwithMoreXColNum	whether source view uses more fields on X-axis	Boolean
SwithMoreYColNum	whether source view uses more fields on Y-axis	Boolean
SwithMoreAggCols	whether source views use more fields using aggregation	Boolean
equalAggCols	whether two views have the same number of fields using aggregation	Boolean
SwithMoreAggColPer	whether source views use more percentage of fields using aggregation	Boolean
equalAggColPer	whether two views use the same percentage of fields using aggregation	Boolean
SwithMoreCols	whether source view uses more fields	Boolean
equalColNum	whether two views use the same number of fields	Boolean
isOverlap	whether two views use the shared fields when regarding *sum(col1)* and *count(col1)* as the same	Boolean

overlapNum	how many shared fields when regarding *sum(col1)* and *count(col1)* as the same	Number
dataOverlapS2T	the percentage of the number of the shared fields occupy the number of the used fields of the target views, i.e., the number of the shared fields/ number of the used fields of the target views (regarding *sum(col1)* and *count(col1)* as the same)	Number
dataOverlapT2S	the percentage of the number of the shared fields occupy the number of the used fields of the source views, i.e., the number of the shared fields/ number of the used fields of the source views (regarding *sum(col1)* and *count(col1)* as the same)	Number
dataR	define the relationship into 5 types, i.e., No, Equal, SasSubset (source view as subset), TasSubset (target view as subset), Intersection (regarding *sum(col1)* and *count(col1)* as the same)	String
SwithMoreCols_agg	whether source view uses more fields when considering *sum(col1)* and *count(col1)* as different	Boolean
equalColNum_agg	whether two views use the same number of fields when considering *sum(col1)* and *count(col1)* as different	Boolean
isOverlap_agg	whether two views use the shared fields when considering *sum(col1)* and *count(col1)* as different	Boolean
overlapNum_agg	how many shared fields when considering *sum(col1)* and *count(col1)* as different	Number
dataOverlapS2T_agg	the percentage of the number of the shared fields occupy the number of the used fields of the target views, i.e., the number of the shared fields/ number of the used fields of the target views (regarding *sum(col1)* and *count(col1)* as different)	Number
dataOverlapT2S_agg	the percentage of the number of the shared fields occupy the number of the used fields of the source views, i.e., the number of the shared fields/ number of the used fields of the source views (regarding *sum(col1)* and *count(col1)* as different)	Number
dataR_agg	define the relationship into 5 types, i.e., No, Equal, SasSubset (source view as subset), TasSubset (target view as subset), Intersection (regarding *sum(col1)* is different from *count(col1)*)	String

- Layout Arrangement Relationship

feature name	Meaning	type
shortest_view_dis	the minimum distance of two views	Number
isNeighbor	whether the minimum distance of two views are in the threshold	Boolean
angle	the angle of the center of two views	Number
direction	we divided the 360 degree to 8 directions and each with 45 degrees, e.g., [-22.5, 22.5) as right, [22.5,67.5] as top right	String
SwithLargerHeight	Boolean, whether the height of source view is larger than that of target view	Boolean
SwithLargerWidth	Boolean, whether the width of source view is larger than that of target view	Boolean
SwithLargerSize	Boolean, whether the size of source view is larger than that of target view	Boolean
equalHeight	whether two views have the same height	Boolean
equalWidth	whether two views have the same width	Boolean
equalSize	whether two views have the same size	Boolean

- Coordination Features

feature name	Meaning	type
coordination	the coordination type between two views, including *filter, brush, and no*	String