

KB4VA Specification Codebook

Analytical Task Specification

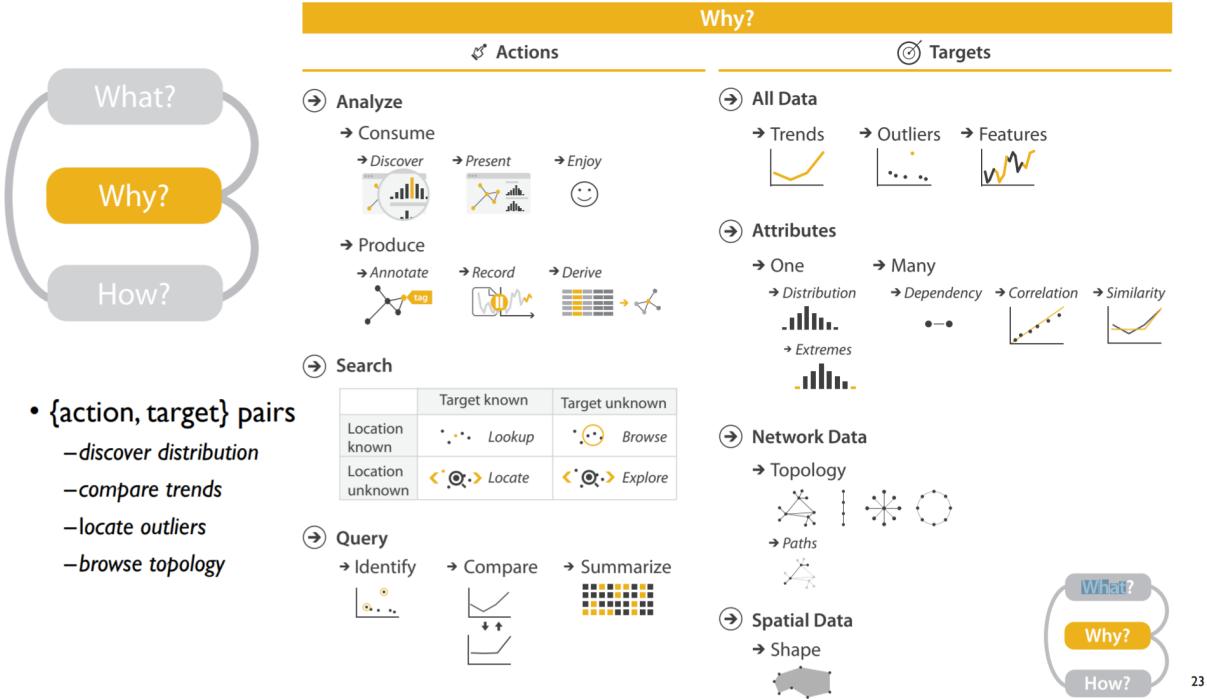


Figure is from Munzner T. *Visualization analysis and design*[M]. CRC press, 2014. Page 46 & Page 56

1. ACTION

The definitions of actions are quoted from Brehmer M, Munzner T. *A multi-level typology of abstract visualization tasks*[J]. *IEEE transactions on visualization and computer graphics*, 2013, 19(12): 2376-2385.

Consume

- **Present** refers to “the use of visualization for the succinct communication of information, for telling a story with data, guiding an audience through a series of cognitive operations”.
- **Discover** is “about the generation and verification of hypotheses, associated with modes of scientific inquiry”.
- **Enjoy** refers to “casual encounters with visualization”.

Produce

Definition: We use “produce in reference to tasks in which the intent is to generate new artifacts, including transformed or derived data, annotations, recorded visualization interactions, or screenshots of static visualizations”.

Search

➔ Search

	Target known	Target unknown
Location known	••• <i>Lookup</i>	••• <i>Browse</i>
Location unknown	⟨•••⟩ <i>Locate</i>	⟨•••⟩ <i>Explore</i>

- **Lookup:** target known, location known
- **Browse:** target unknown, location known
- **Locate:** target known, location unknown
- **Explore:** target unknown, location unknown

Query

- **Identify:** “returns characteristics or reference for a target”
- **Compare:** “returns characteristics or reference for two or multiple targets”
- **Summarize:** “returns characteristics or reference for a whole set of targets”

1.2 TARGET

Tabular Data

Values, extremum, ranges, distributions, anomalies, clusters, correlation, similarities, orders
The taxonomy is based on *Amar R, Eagan J, Stasko J. Low-level components of analytic activity in information visualization[C] IEEE Symposium on Information Visualization, 2005. INFOVIS 2005. IEEE, 2005: 111-117.*

Graph Data

Graphs, nodes, links/paths, topology/structures, group/clusters

The taxonomy is based on *Lee B, Plaisant C, Parr C S, et al. Task taxonomy for graph visualization[C] Proceedings of the AVI workshop on BEyond time and errors: novel evaluation methods for information visualization. 2006: 1-5.*

Data & Visualization Specification

We specify the visualization configuration in Visual Analytics designs by referring to declarative programming grammars, such as Vega, Vega-lite, Echarts, which are proven to be efficient to specify the visual encodings and compositions.

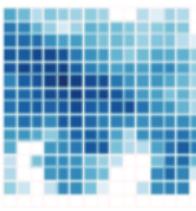
Specifically, Echarts supports generation of graph-related data:

- Sankey: Node + Link
- Tree/treemap:
 - data that is organized in a hierarchical structure, {name, children:[{name, children}], collapse:boolean}.
 - Layout: radial,
- Graph:
 - Node + Link

Our primary goal is to specify the visual mapping from data to visual channels/layouts.

1. Specify the visual mapping between “field”, “type”, and “encoding”

Original



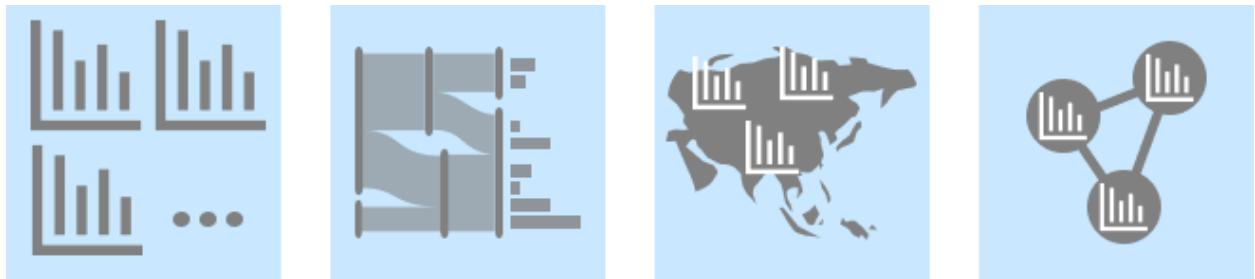
```
{  
  "mark": "rect",  
  "encoding": {  
    "y": {  
      "field": "flow_1",  
      "type": "nominal"  
    },  
    "x": {  
      "field": "flow_2",  
      "type": "nominal"  
    },  
    "color": {  
      "field": "flow_coverage",  
      "type": "quantitative"  
    }  
  }  
}
```

- About “field”: using the name in the paper, and check if the two fields are the same column (double encoding).
 - About “aggregate”: bin, mean, count, sum, min, max, median
2. Specify the composition type of the visualization

Aggregate Types

- Count
- Sum
- Min/max
- Average (mean)
- Median
- Variance
- Stdev
- q1, q3, ci0, ci1

Composition Types



Facet

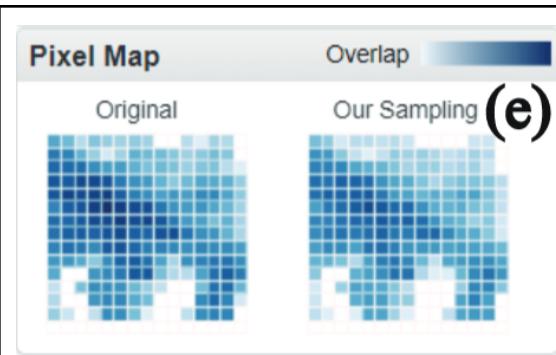
Specification:

...

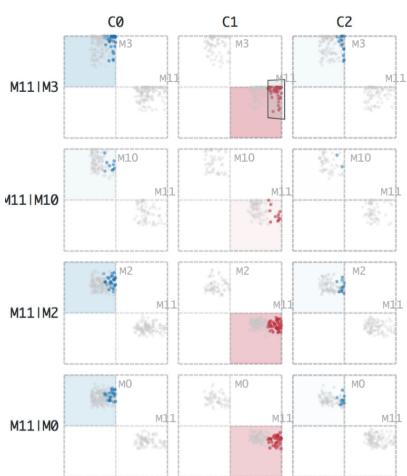
```
facet: {
  layout: "grid/circular/mirrored" (default grid)
  row:{}
  column:{}
},
spec: {}
...
```

Example:

1. Grid layout

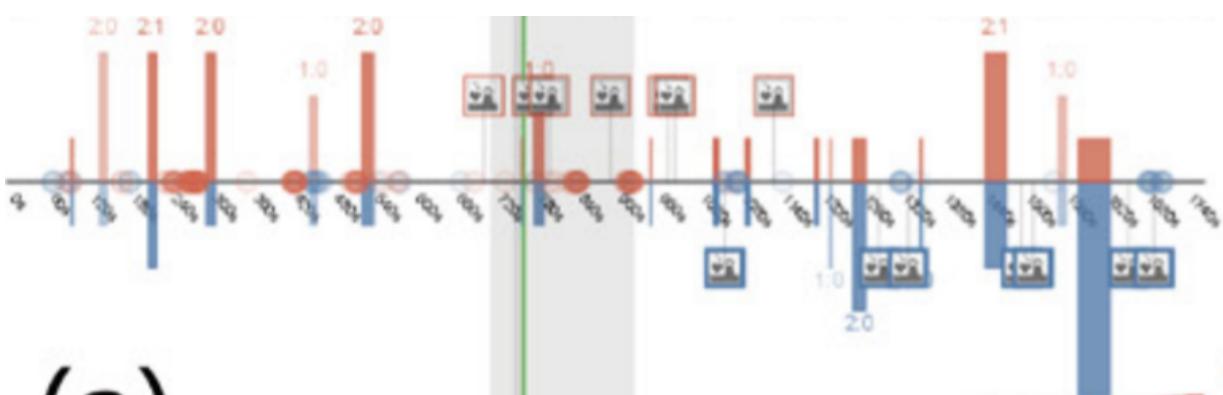


```
{  
  "facet": {  
    "column": {  
      "field": "method",  
      "type": "nominal"  
    }  
  },  
  "spec": {  
    "mark": "rect",  
    "encoding": {  
      "y": {  
        "field": "flow_1",  
        "type": "nominal"  
      },  
      "x": {  
        "field": "flow_2",  
        "type": "nominal"  
      },  
      "color": {  
        "field": "flow_coverage",  
        "type": "quantitative"  
      }  
    }  
  }  
}
```



```
{
  "facet": {
    "column": {
      "field": "class (Ck)",
      "type": "nominal"
    },
    "row": {
      "field": "model_pair (Mi, Mj)",
      "type": "nominal"
    }
  },
  "spec": {
    "mark": "point",
    "encoding": {
      "x": {
        "field": "Mi_prediction_score",
        "type": "quantitative"
      },
      "y": {
        "aggregate": "Mj_prediction_score",
        "type": "quantitative"
      },
      "color": {
        "field": "ground_truth_is_Ck_or_not",
        "type": "nominal"
      }
    }
  }
}
```

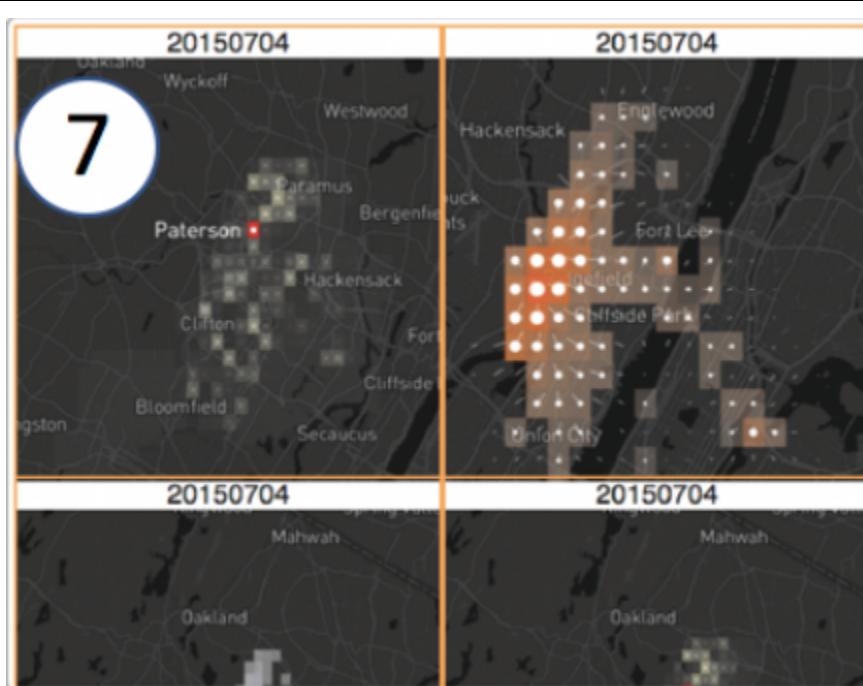
2. Mirrored layout



```
{  
  "layer": [  
    {  
      "facet": {  
        "layout": "mirrored",  
        "row": {  
          "field": "team",  
          "type": "nominal"  
        },  
        "color": {  
          "field": "team",  
          "type": "nominal"  
        }  
      },  
      "spec": {  
        "mark": "bar",  
        "encoding": {  
          "x": {  
            "field": "time",  
            "type": "temporal"  
          },  
          "y": {  
            "field": "combat_result",  
            "type": "quantitative"  
          },  
          "width": {  
            "field": "duration",  
            "type": "quantitative"  
          }  
        }  
      },  
      {  
        "mark": "point",  
        "remark": "attacking towers",  
        "encoding": {  
          "x": {  
            "field": "time",  
            "type": "temporal"  
          }  
        }  
      }  
    }  
  ]  
}
```

```
        "type": "temporal"
    },
    "color": {
        "field": "team",
        "type": "nominal"
    }
}
{
    "mark": "others",
    "remark": "occupying light towers",
    "encoding": {
        "x": {
            "field": "time",
            "type": "temporal"
        },
        "stroke": {
            "field": "team",
            "type": "nominal"
        }
    }
}
]
```

3. List layout



```

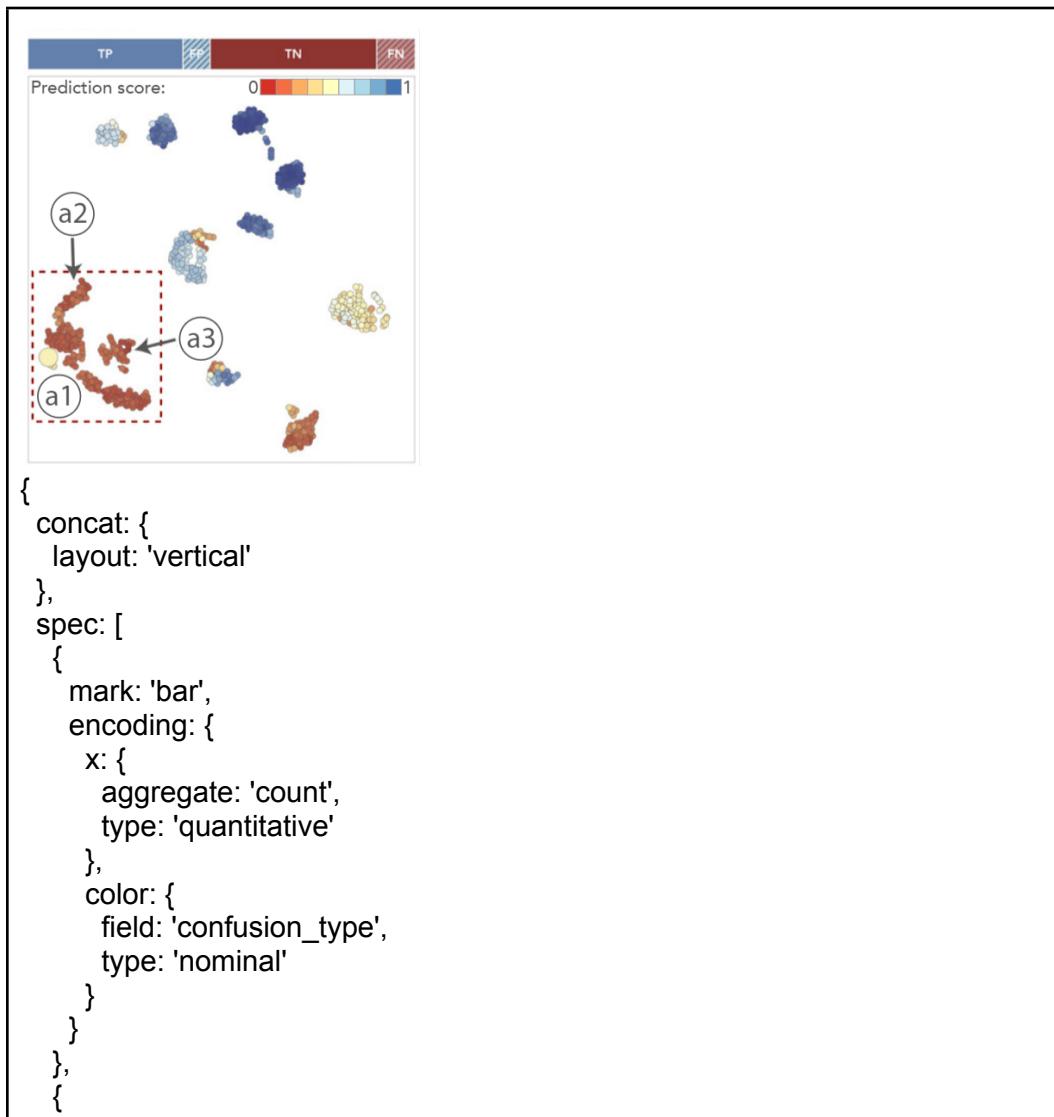
"facet": {
  "field": "date",
  "columns": 2,
  "type": "temporal"
},
"spec": {
}
}

```

Concat

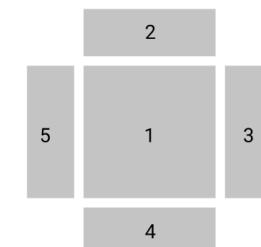
Vega-lite: vconcat, hconcat. However, there might be multiple concatenation directions, such as crossing. Therefore, we have a concat + layout specification

1. Vertical/horizontal



```
mark: 'point',
encoding: {
  x: {
    field: 'dr_1',
    type: 'quantitative'
  },
  y: {
    field: 'dr_2',
    type: 'quantitative'
  },
  color: {
    field: 'prediction_score',
    type: 'quantitative'
  }
}
]
```

2. Crossing



```
{
  "concat": {
    "layout": "crossing"
  },
  "spec": [
    {
      "mark": "geoshape",
      "position": 1,
      "encoding": {}
    },
    {
      "mark": "line",
      "position": 2,
      "encoding": {}
    },
    {
      "mark": "line",
      "position": 3,
      "encoding": {}
    }
}
```

```
]  
}
```

Layer

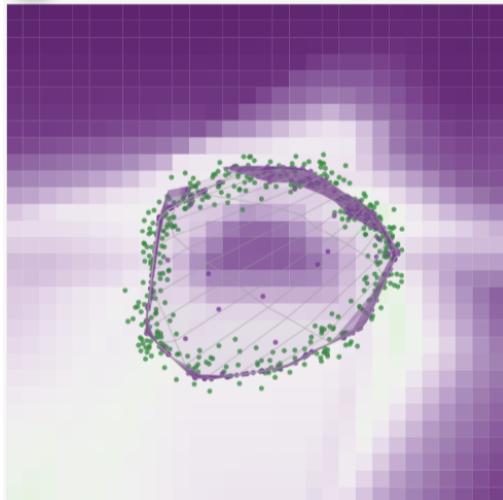
Specification:

...

layer: []

...

1. Plain layout

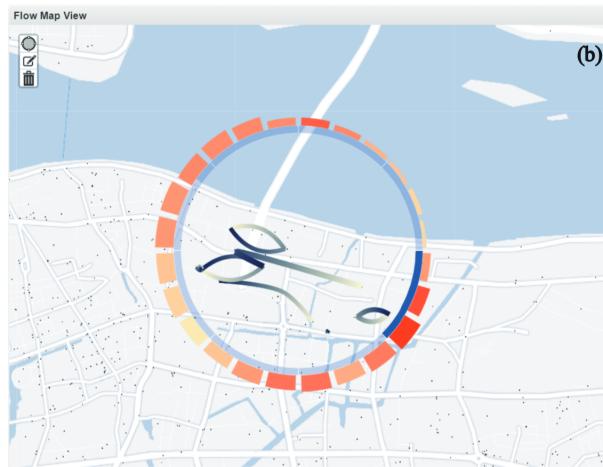


```
{
  "layer": [
    {
      {
        "mark": "point",
        "encoding": {
          "x": {
            "field": "dimension_1",
            "type": "quantitative"
          },
          "y": {
            "field": "dimension_2",
            "type": "quantitative"
          }
        }
      }
    ]
  }
}
```

2. Circular layout with content inside

a. ring/donut

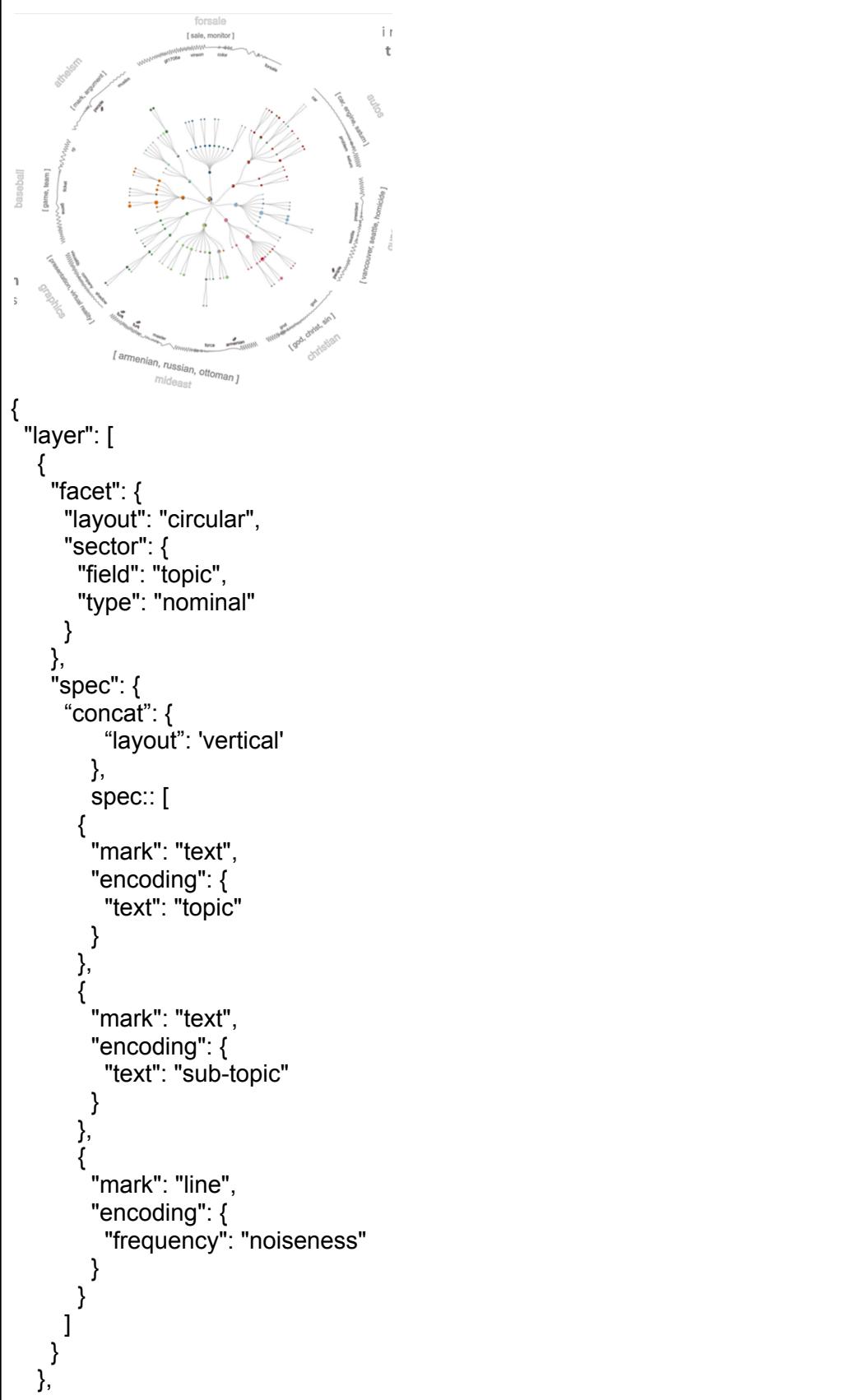
mark: "bar", layout: "circular"



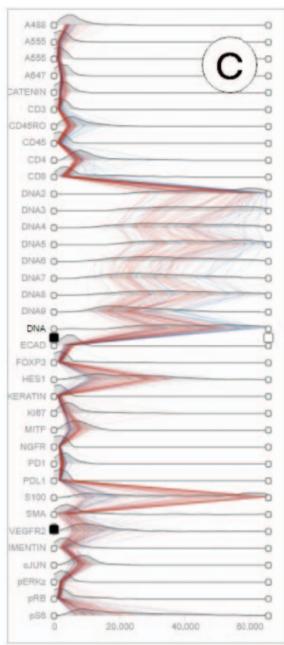
```
{
  "layer": [
    {
      "mark": "geoshape"
    },
    {
      "mark": "point",
      "encoding": {
        "x": {
          "field": "geo_x",
          "type": "quantitative"
        },
        "y": {
          "field": "geo_y",
          "type": "quantitative"
        }
      }
    },
    {
      "mark": "bar",
      "layout": "circular",
      "encoding": {
        "x": {
          "field": "hour",
          "type": "nominal"
        },
        "y": {
          "field": "flow_magnitude",
          "type": "quantitative",
          "aggregate": "sum"
        }
      }
    }
  ]
}
```

```
        "color": {
          "field": "flow_magnitude",
          "type": "quantitative",
          "aggregate": "mean"
        },
        "ring": {
          "field": "focus_interval",
          "type": "quantitative"
        }
      },
      {
        "mark": "graph",
        "encoding": {
          "node_x": {
            "field": "OD_x",
            "type": "quantitative"
          },
          "node_y": {
            "field": "OD_y",
            "type": "quantitative"
          },
          "link": {
            "field": "OD_path",
            "type": "relation"
          },
          "link_color": {
            "field": "time",
            "type": "temporal"
          }
        }
      }
    ]
}
```

- b. segments
facet:{layout: circular}



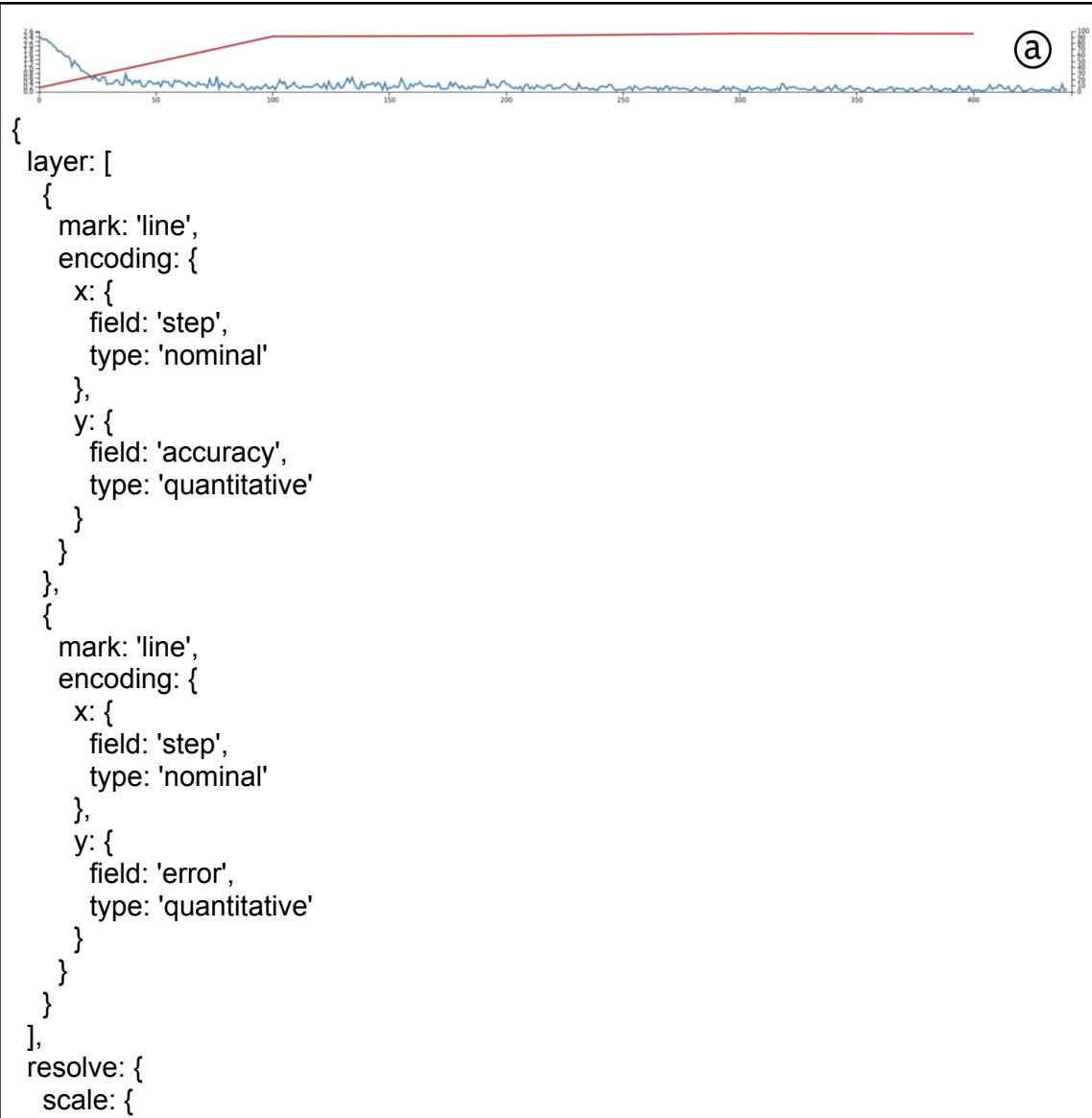
```
{  
  "nested": {  
    "parent": {  
      "mark": "tree",  
      "encoding": {  
        "node": {  
          "field": "topic",  
          "type": "nominal"  
        },  
        "link": {  
          "field": "tree_hierarchy",  
          "type": "relation"  
        }  
      },  
      "child": {  
        "child_type": "configured",  
        "canvas": "node",  
        "configuration": {  
          "mark": "arc",  
          "encoding": {  
            "theta": {  
              "aggregate": "count",  
              "type": "quantitative"  
            },  
            "color": {  
              "field": "document-authors",  
              "type": "nominal"  
            },  
            "size": {  
              "field": "text_amount",  
              "type": "quantitative"  
            }  
          }  
        }  
      }  
    }  
  }  
}
```



```
{
  nested: {
    parent: {
      mark: 'line',
      encoding: {
        x: {
          field: 'feature_value',
          type: 'quantitative'
        },
        y: {
          field: 'feature',
          type: 'nominal'
        },
        color: {
          field: 'subset',
          type: 'nominal'
        }
      }
    },
    child: {
      child_type: 'configured',
      canvas: 'axis',
      configuration: {
        mark: 'area',
        encoding: {
          x: {
            field: 'feature_value',
            bin: true,
            type: 'quantitative',
            remark: 'square root scaling'
          }
        }
      }
    }
  }
}
```

```
 },
y: {
  aggregate: 'density',
  type: 'quantitative'
}
}
}
```

3. Dual Axes



```

        y: 'independent'
    }
}
}
```

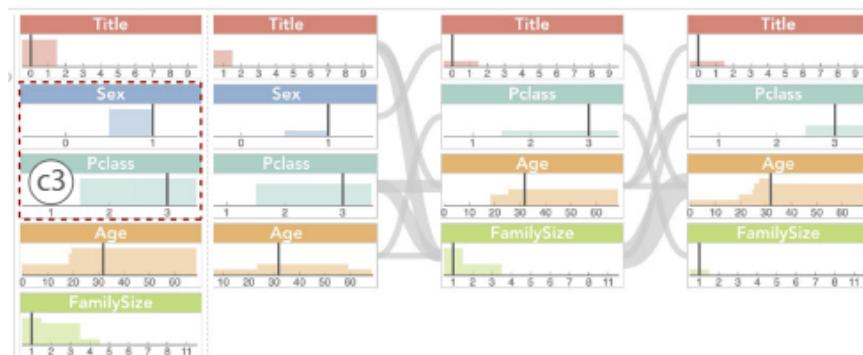
Nested

Specification:

...

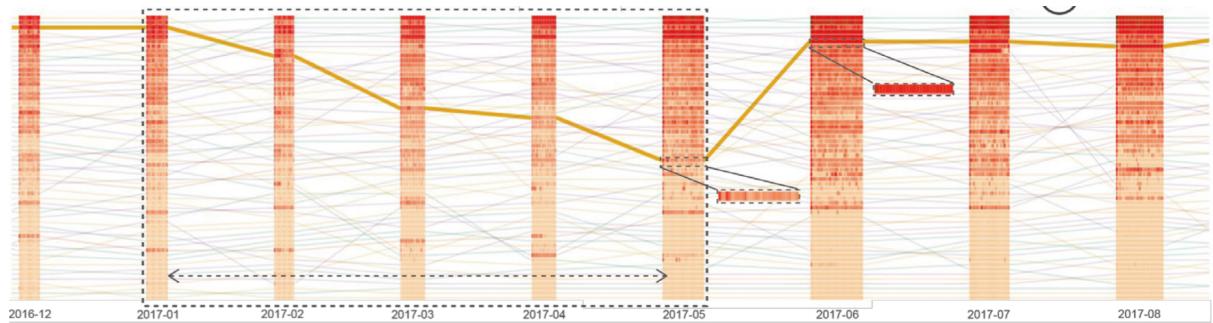
```

nested: {
    parent:{},
    child:{}
        "canvas": "inner_circular_area/node",
        "child_type": "configured/exemplified",
    }
}
...
```



```
{
  "nested": {
    "parent": {
      "mark": "sankey",
      "encoding": {
        "node": {
          "field": "attribute_type",
          "type": "nominal"
        },
        "link": {
          "field": "decision_path",
          "type": "relation"
        }
      }
    },
    "child": {
      "canvas": "node",
      "y": "independent"
    }
  }
}
```

```
"configuration": {  
  "mark": "bar",  
  "encoding": {  
    "x": {  
      "field": "attribute_value",  
      "type": "quantitative"  
    },  
    "y": {  
      "aggregate": "count",  
      "type": "quantitative"  
    }  
  }  
}
```



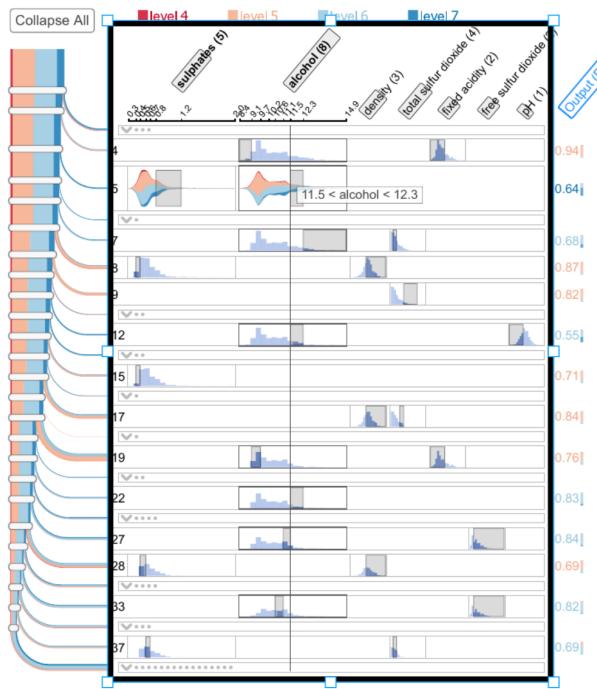
```
{  
  nested: {  
    parent: {  
      mark: 'line',  
      encoding: {  
        x: {  
          field: 'year_month',  
          type: 'temporal'  
        },  
        y: {  
          field: 'exchange',  
          type: 'nominal'  
        },  
        color: {  
          field: 'continent',  
          type: 'nominal'  
        }  
      }  
    },  
    child: {  
      canvas: 'axis',  
      configuration: {  
        mark: 'rect',  
        encoding: {  
          x: {  
            field: 'year_month',  
            type: 'temporal'  
          },  
          y: {  
            field: 'exchange',  
            type: 'nominal'  
          }  
        }  
      }  
    }  
  }  
}
```

```
encoding: {
  x: {
    field: 'year_month',
    type: 'temporal'
  },
  xOffset: {
    field: 'day',
    type: 'temporal'
  },
  y: {
    field: 'exchange',
    type: 'nominal'
  },
  width: {
    field: 'client',
    aggeragate: 'count',
    type: 'quantitative'
  },
  color: {
    field: 'transaction_volume',
    type: 'quantitative'
  }
}
```

Conditions (OR)

We introduce condition specifications to handle the situations that different visualizations are rendered by conditions.

(B)



```
{  
  facet: {  
    row: {  
      field: 'feature_name',  
      type: 'categorical',  
      sort: 'importance_score'  
    },  
    column: {  
      field: 'rule',  
      type: 'ordinal'  
    }  
  },  
  spec: {  
    mark: 'bar',  
    encoding: {  
      x: {  
        field: 'feature_value',  
        type: 'quantitative',  
        bin: true  
      },  
      y: {  
        aggregate: 'count',  
        type: 'quantitative'  
      }  
    }  
  },  
  alternative: {  
    facet: {  
      row: {  
        field: 'feature_name',  
        type: 'categorical',  
        sort: 'importance_score'  
      },  
      column: {  
        field: 'rule',  
        type: 'ordinal'  
      }  
    },  
    spec: {  
      mark: 'rule',  
      encoding: {  
        x: {  
          field: 'rule',  
          type: 'ordinal'  
        },  
        y: {  
          field: 'feature_name',  
          type: 'categorical'  
        }  
      }  
    }  
  }  
}
```

```
field: 'feature_name',
type: 'categorical',
sort: 'importance_score'
},
column: {
  field: 'rule',
  type: 'ordinal'
}
},
spec: {
  condition_1: {
    test: 'feature type is continuous',
    value: {
      mark: 'area',
      encoding: {
        x: {
          field: 'feature_value',
          type: 'quantitative',
          bin: true
        },
        y: {
          aggregate: 'count',
          type: 'quantitative'
        }
      }
    }
  },
  condition_2: {
    test: 'feature type is discrete',
    value: {
      mark: 'bar',
      encoding: {
        x: {
          field: 'feature_value',
          type: 'quantitative',
          bin: true
        },
        y: {
          aggregate: 'count',
          type: 'quantitative'
        }
      }
    }
  },
  condition_3: {
    test: 'a clause is using feature j',
    value: {
      mark: 'rect',
      encoding: {
        x: {
```

```
        field: 'start interval in the clause',
        type: 'quantitative'
    },
    x2: {
        field: 'end interval in the clause',
        type: 'quantitative'
    }
}
}
}
}
}
```



```
nested: {
  parent: {
    mark: 'graph',
    encoding: {
      node_left: {
        field: 'player',
        type: 'nominal'
      },
      node_right: {
        field: 'passing_pattern',
        type: 'nominal'
      },
      link: {
        field: 'player_involved_in_passing_pattern',
        type: 'relation'
      }
    }
}
```

```

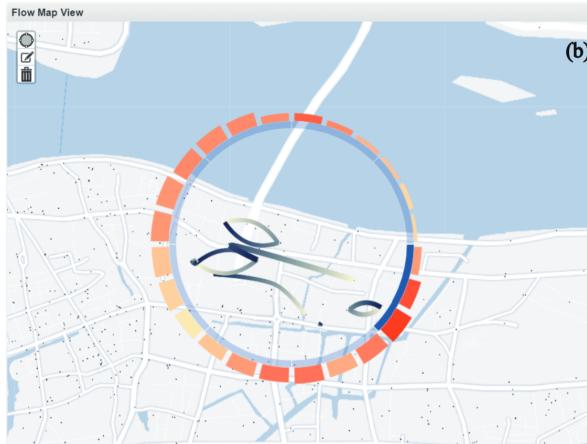
},
},
child: {
  canvas: 'node',
  configuration: {
    condition_1: {
      test: 'node_left',
      value: {
        mark: 'bar',
        encoding: {
          x: {
            field: 'total_pass',
            type: 'quantitative'
          },
          y: {
            field: 'player',
            type: 'nominal'
          },
          color: {
            field: 'pass_in_passing_pattern',
            type: 'nominal',
            remark: 'When hovering on a pattern, a dark bar (Fig. 4 (B)) is presented to show
the number of a player's passes in that passing pattern'
          }
        }
      }
    },
    condition_2: {
      test: 'node_right',
      value: {
        concat: {
          layout: 'vertical'
        },
        spec: [
          {
            mark: 'bar',
            encoding: {
              x: {
                field: 'pass_in_passing_pattern',
                aggregate: 'count',
                type: 'quantitative'
              }
            }
          },
          {
            mark: 'surface',
            encoding: {
              x: {
                field: 'soccer_pitch_dim_1',
                type: 'quantitative'
              }
            }
          }
        ]
      }
    }
  }
}

```

```
        },
        y: {
          field: 'soccer_pitch_dim_1',
          type: 'quantitative'
        },
        surface: {
          field: 'start_or_end_position',
          aggregate: 'count',
          type: 'quantitative'
        }
      }
    }
  ]
}
}
```

Visualization Types

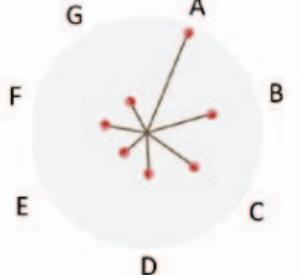
Type	Remarks	Examples
Bar chart	x, y, color, width, xOffset	<p>Edge Betweenness</p> <p>(f)</p> <pre>{ mark: 'bar', encoding: { x: { field: 'EB', type: 'quantitative', bin: 'true' }, y: { field: 'frequency', type: 'quantitative' } } }</pre>

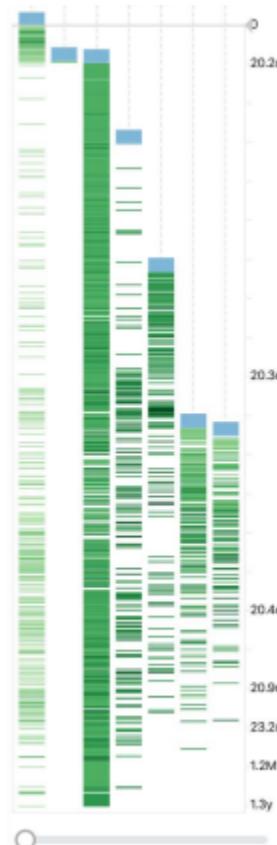
		<pre> }, xOffset: { field: 'method_type', type: 'nominal' }, color: { field: 'method_type', type: 'nominal' } } } } } </pre>
Bar chart (circular)		 <pre> Flow Map View (b) { "mark": "bar", "layout": "circular", "encoding": { "x": { "field": "hour", "type": "nominal" }, "y": { "field": "flow_magnitude", "type": "quantitative", "aggregate": "sum" }, "color": { "field": "flow_magnitude", "type": "quantitative", "aggregate": "mean" }, "ring": { "field": "focus_interval", "type": "quantitative" } } } </pre>

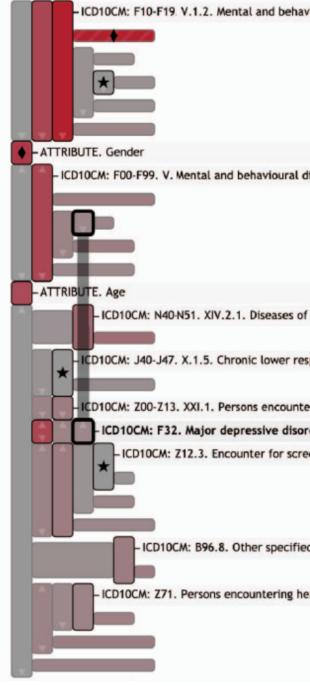
Heatmap		<pre>{ "mark": "rect", "encoding": { "y": { "field": "flow_1", "type": "nominal" }, "x": { "field": "flow_2", "type": "nominal" }, "color": { "field": "flow_coverage", "type": "quantitative" } } }</pre>
Surface graph		<pre>{ "mark": "surface", "encoding": { "x": { "field": "dimension_1", "type": "quantitative" }, "y": { "field": "dimension_2", "type": "quantitative" }, "surface": { "field": "probability_density", "type": "quantitative" } } }</pre>
Contour graph	<p>A contour line of a function of two variables is a curve along which the function has a constant value, so that the curve joins points of equal value.</p> <p>https://en.wikipedia.org/wiki/Contour_line</p>	<pre>{ "mark": "contour", "encoding": { "x": { "field": "dimension_1", "type": "quantitative" }, "y": { "field": "dimension_2", "type": "quantitative" }, } },</pre>

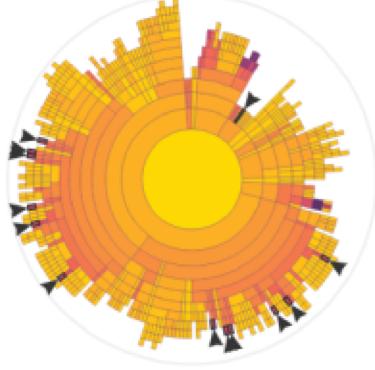
		<pre> "contour": { "field": "value", "type": "quantitative" } } </pre>
Scatter plot		<pre> { "mark": "point", "encoding": { "x": { "field": "dimension_1", "type": "quantitative" }, "y": { "field": "dimension_2", "type": "quantitative" } } } </pre>
Scatter (circular)	In some cases, scatterplots are organized by radius and angle in a circular layout.	<pre> { mark: 'point', layout: 'circular', encoding: { radius: { field: 'jacard index', type: 'quantitative' }, theta: { field: 'category', type: 'nominal' } } } </pre>
Line chart		<pre> { "mark": "line", "encoding": { "x": { "field": "epoch", "type": "quantitative" }, "y": { "field": "loss", "type": "quantitative" }, "color": { "field": "loss_type", "type": "nominal" } } } </pre>

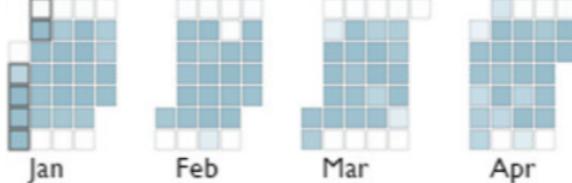
		<pre> "type": "nominal" } } } } </pre>
PCP		<pre> { "mark": "line", "encoding": { "x": { "field": "feature type", "type": "nominal" }, "y": { "aggregate": "feature value", "type": "quantitative" } } } </pre>
Matrix		<pre> { "mark": "rect", "encoding": { "y": { "field": "flow_1", "type": "nominal" }, "x": { "field": "flow_2", "type": "nominal" }, "color": { "field": "flow_coverage", "type": "quantitative" } } } </pre>
Radar chart		<pre> { "mark": "radar", "encoding": { "theta": { "field": "metric_type", "type": "nominal" }, "radius": { "field": "metric_value", "type": "quantitative" }, "color": { "field": "metric_color", "type": "nominal" } } } </pre>

		<pre> "field": "method", "type": "nominal" } } } } </pre>
Area chart		<pre> { "mark": "area", "encoding": { "x": { "field": "epoch", "type": "quantitative" }, "y": { "aggregate": "sum", "field": "count" }, "color": { "field": "action/reward_type", "type": "nominal" } } } </pre>
Box plot		<pre> { "mark": "boxplot", "encoding": { "y": { "field": "word", "type": "nominal" }, "x": { "field": "activation_distribution", "type": "quantitative" } } } </pre>
Dandelion graph		<p>Dandelion Glyph</p>  <pre> { "name": "dandelion", "x": 500, "y": 500, "r": 100 } </pre>

		<pre> "mark": "dandelion", "layout": "circular", "encoding": { "theta": { "field": "attribute", "type": "nominal" }, "radius": { "field": "group", "type": "nominal" } } } </pre>
Strip graph		 <pre> { mark: 'tick', encoding: { y: { field: 'delay time of reposting', type: 'temporal' }, x: { field: 'player', type: 'nominal' } } } </pre>

		<pre> }, color: { field: 'post type', type: 'nominal' } } } } } </pre>
Word cloud		<pre> { "mark": "word_cloud", "encoding": { "word": { "field": "word", "type": "nominal" }, "size": { "field": "TF-IDF value", "type": "quantitative" } } } </pre>
Icicle graph	Special cases of node-link diagrams	 <pre> { mark: 'icicle', encoding: { node: { field: 'shifts', type: 'node', color: { field: 'type', type: 'nominal' } } } } </pre>

		<pre> field: 'weighted distance', type: 'quantitative' } }, link: { field: 'hierarchy', type: 'relation' } } } </pre>
sunburst		 <pre> { mark: 'sunburst', encoding: { node: { field: 'node', type: 'others', encoding: { color: { field: 'metric', type: 'quantitative', remark: 'Darker colors indicate stronger metrics' }, width: { field: 'count', type: 'quantitative', remark: 'The width of segments communicates the number of leaf nodes, with each leaf having equal weight, and all charts are sorted identically to support comparisons.' } } }, link: { field: 'global pattern hierarchy', type: 'relation' } } } </pre>

		<pre> } } } </pre>
Unit visualization		 <pre> { mark: 'unit', encoding: { unit: { field: 'model', type: 'node' }, color: { field: 'model type', type: 'quantitative' } } } </pre>
Calendar		 <pre> { mark: 'calendar', encoding: { x: { field: 'month', type: 'temporal' }, unit: { field: 'day', type: 'node' }, color: { field: 'number of products', type: 'quantitative' } } </pre>

		<pre> } } } </pre>
Violin plot	<p>A violin plot is a method of plotting numeric data. It is similar to a box plot, with the addition of a rotated kernel density plot on each side.</p> <p>https://en.wikipedia.org/wiki/Violin_plot</p>	<pre> { mark: 'boxplot', encoding: { yoffset: { field: 'observed/predicted', type: 'nominal' }, color: { field: 'observed/predicted', type: 'nominal' }, x: { field: 'speed', type: 'quantitative' } } } </pre>
Venn		<pre> { mark: 'venn', encoding: { set: { field: 'element-to-group', type: 'relation' } } } </pre>
Tree		<pre> { mark: 'tree', encoding: { node: { field: 'shifts', type: 'node', encoding: { color: { field: 'weighted distance', </pre>

		<pre> type: 'quantitative' } } }, link: { field: 'hierarchy', type: 'relation' } } } </pre>
Treemap		<pre> { mark: 'treemap', encoding: { node: { field: 'shifts', type: 'node', encoding: { color: { field: 'weighted distance', type: 'quantitative' } } }, link: { field: 'hierarchy', type: 'relation' } } } </pre>
graph		<pre> { "mark": "graph", "encoding": { "node": { "field": "plan", "type": "node" }, "link": { "field": "difference between plans", "type": "relation" } } } </pre>
sankey		<pre> { "mark": "sankey", "encoding": { "node": { "field": "attribute_type", </pre>

```
        "type": "node"
    },
    "link": {
        "field": "decision_path",
        "type": "relation"
    }
}
```