

Prompts in Narrative Player

We provide the prompts employed for the Large Language Model (LLM) within the Narrative Analysis module of the Narrative Player. These prompts specifically correlate to the sections where the LLM operates, as detailed in Sections 3.1, 3.3, 3.4, and 3.5 of the paper.

1 Prompt I for Narration Slicing and Classification

The data is "`{{data}}`".

The narration is "`{{narration}}`".

You are a narration analyst. Please complete 2-step task:

1. Determine all the sentences that contain the data fact.
2. Divide each sentence containing the data fact into several clauses, and output them in the format of a List.

Note: we identify a clause as a unit of a sentence that forms recognizable syntactic constituents: a subject and predicate, with or without adjuncts, or solely a predicate with or without adjuncts.

For example, the sentence A contains four clauses: [A1, A2, A3, A4].

Please only output all the clauses in one List. Don't separate them for each sentence. The format should be [Clause1, Clause2,...]

2 Prompt II for Preliminary Extraction of Data Fact Candidates

You are a narration analyst who specifies in converting clauses of long text entered by the user individually into multiple possible structured representations.

The representation is in the format as

{

```

    'Type': String,
    'Parameter': String/ Float/ Null,
    'Measures': List,
    'Context': {
        String: List,
        String: List,
        ...
    },
    'Breakdowns': List,
    'Focus': List | Null
}

```

Here is the definition of each element in the structured representation.

1. 'Type' represents the type of fact contained in this sentence, including the following categories:

- 'trend': Analyse how the data changes over time series
- 'distribution': distribution of the data over the set
- 'comparison': Give emphasis to comparison of different entities
- 'correlate': relationships between the columns
- 'deviation': Compare data with certain values like zero, mean, or a certain value
- 'anomalies': anomalies within the dataset
- 'extremum': extreme values of data attribute
- 'proportion': component elements of a single entity
- 'sort': Rank data according to some ordinal metric

2. 'Parameter' is a 'Type'-related attribute. It is usually Null, except for the following categories:

-if 'Type' is 'deviation', then the 'Parameter' should be 'Mean' (String) or a specific Number (Float)

-if 'Type' is 'trend', the 'Parameter' should be 'Increase' or 'Decrease'

-if 'Type' is 'extremum', the 'Parameter' should be 'Minimal' or 'Maximal'

3. 'Measures' is a List, including a field or fields in the data table that is treated as a dependent variable for this fact. If the 'Type' is tasks such as correlate and comparison, 'Measure' contains at least two items.

4. 'Context' is a Dict representing the whole data subspace, where the key should be a String, i.e., a field in the data table, and the value is a List, where items are specific items in the field to filter the space into subspace.

5. 'Breakdowns' is a List, where item(s) is a field in the data, dividing the subspace into child groups in this subspace. 6. 'Focus' is a List, where item(s) is (1) specific items in Breakdowns, if Length of Breakdowns' List >1; (2) specific items in the field corresponding to the only element in breakdowns. Focus represents the group in this subspace to be highlighted. If there's no subjects to emphasize, the Focus should be Null

Here is one example:

The data is "{{data}}".

The narration is "{{narration}}".

The clause to be converted is "{{clause}}".

The converted two fact candidates are [fact1, fact2].

Now you are required to do the transformation from Clauses to Structured representation candidates.

The data is "{{data}}".

The narration is "{{narration}}".

The ClauseList is "[clause1, clause2, ...]"

List 3 possible fact candidates in json format. The format of Output should be:

```
[
  A: [
    {
      A1
    }, {
      A2
    }, {
      A3
    }
  ], ....
]
```

Where A is the Clause, while A1, A2, and A3 are 3 fact candidates.

Notes: Don't output any other words, just output the json data

The 3 candidates should show the diversity and richness of the 'Context' and 'Focus'. Avoid 3 candidates who have the same 'Context' and 'Focus' fields.

The 'Focus' and the value in 'Context' shouldn't be the same. Because the 'Focus' and the value in 'Context' are the same, the Focus is meaningless. In this case, the Context should be an empty Dict or the Focus should be an empty List.

If there is no relevant information in the data for a certain Clause, output 'None' for that Clause

There should be no more than 2 'Type' in 3 candidates for each Clause in ClauseList. Usually, the 'Type' of 3 candidates for one Clause are the same.

3 Prompt III for Data Fact Validation and Clause Characterization

The background about fact representation will be explained at the beginning as the part in Prompt II

The narration is: ... {} ...

where {} represents a clause that you need to fill.

Here are 3 fact candidates that you can use as a basis for clause generation: [F1, F2, F3].

Write clauses [C1, C2, C3] individually based on these fact candidates to fill in the {} of the context.

Pay attention to maintaining the coherence of the context and avoid redundant information. The clauses shouldn't contain any punctuation marks that indicate division, such as commas, semicolons, etc., and only have one subject-predicate system.

4 Prompt IV for Data Fact Inference and Completion within Context

The background about fact representation will be explained at the beginning as the part in Prompt II

The data is "{{data}}".

The narration is "{{narration}}".

The ClauseList is "[clause1, clause2, ...]"

We have generate possible fact candidates in json format in the ClauseList: [clause1: None, clause2: [Fact1, Fact2, ...], ...]

There are some clauses initially with "None" fact candidates, as they are metaphorical expressions, vague statements, or personifications. You are required to infer the data behind these vague Clauses following the 4 steps below:

1. Check all nouns and adjectives in the clause, and extract the relevant zero, one or more sets of data properties or values accordingly

2. If any relevant data column names cannot be deduced in step 1, copy the candidates of the previous clause or the latter clause as appropriate, and also keep the context consistent.

3. If multiple groups of related data properties or values are inferred in step 1, check the data properties and values (especially the Measures) determined in the proceeding or following clauses as references, and filter the related data properties or values accordingly by the union of the intersection of each reference.

4. Use the data properties and values and context information obtained in step 1/2/3 to infer the 3 fact candidates of each clause.

You need to output 3 fact candidates of each clause with initial "None" fact candidates.