LLM-assisted coding of qualitative data

Example text: `data/essay.md`

Zero-shot coding

Zero-shot prompts are just instructions for the Large Language Model (LLM) to follow. They do not include examples of the desired output and rely entirely on the model's pre-trained knowledge.

Prompt:

> Perform qualitative coding of the following essay:
>
> `<paste essay text here>`

Refined prompt with proper context

The model can only work with the context we provide in our prompt. LLMs also do not have any internal state, they only process the input sequence of words. Therefore, previous interactions do not have any impact on subsequent conversations, unless there is an application layer around the LLM that saves and injects previous interactions into your prompt.

Because of this, we need to provide as much explicit context as possible:

> I am a qualitative researcher at the Center for Program Design & Evaluation at Dartmouth College. I am working on a collection of essays by Dartmouth students reflecting on their experience with the Guarini Institute Exchange Program. I am most interesetd in capturing the students positive and negative experiences, and extract actionable information on how to improve the program.
>
> With that in mind, perform qualitative coding of the following essay:
>
> `<paste essay text here>`

Request specific output format

LLM outputs should rarely be the final result of any analysis. Instead, a human analyst should take the LLM's response and further refine it, potentially in a feedback loop with the LLM. This paradigm is called *human in the loop*.

To facilitate this, we often want the LLM to respond in a particular format, which makes it easier to export the preliminary result into other analysis tools.

For example: We could assume that we are working with a tool that can export and import a coded document in XML format. In that case, a prompt like this could be helpful:

> I am a qualitative researcher at the Center for Program Design & Evaluation at Dartmouth College. I am working on a collection of essays by Dartmouth students reflecting on their experience with the Guarini Institute Exchange Program. I am most interesetd in capturing the students positive and negative experiences, and extract actionable information on how to improve the program.

> Respond with the marked up essay using XML tags to highlight the coded sections.
>
> With that in mind, perform qualitative coding of the following essay:
>
> `<paste essay text here>`

Few-shot prompting

Few-shot prompting adds examples of the desired output to the model's instructions. These examples show patterns that help guide the model to align its new outputs with the patterns already existing in the examples. This is also called "in-context learning".

In our example here, we might be interested in coding multiple essays with a certain amount of consistency. To achieve this, we can provide one fully or partially (human-coded) essay as an example, before then prompting the model to code the new example.

> I am a qualitative researcher at the Center for Program Design & Evaluation at Dartmouth College. I am working on a collection of essays by Dartmouth students reflecting on their experience with the Guarini Institute Exchange Program. I am most interesetd in capturing the students positive and negative experiences, and extract actionable information on how to improve the program.
>
> Respond with the marked up essay using XML tags to highlight the coded sections.
>
> I have already coded the following essay, which you can use as an example for the codes and format I am looking for:
>
> `<paste coded essay here>`
>
> With that in mind, perform qualitative coding of the following essay:
>
> `<paste essay text here>`

Bonus content

If you are using Dartmouth Chat, you can get a nice visualization of a coded document by prompting the model to render it in HTML:

> Render the essays in HTML with the coded sections highlighted, including the label. Put all HTML code in proper code blocks.

This will generate HTML code that Dartmouth Chat renders in the sidebar.

Note: This will most likely create a very long response, which may eventually be cut off due to the default setting for the maximum response length. Once the generation stops, you can continue the response generation by clicking the button that looks like play button labeled *Continue Response*. Once the entire response is generated, the rendered HTML will appear on the side.