Application of LLMs to Qualitative Coding Tasks Using Quallmer

Public Opinion Analytics Lab (POAL) Methods Brief Series

Mollie Ruler and Ken Benoit

19th August 2026

Executive Summary

- This brief outlines the growing potential for the application of LLMs to qualitative coding tasks. It presents a pre-validated workflow, developed and refined through studies examining the measurement of issue positions and issue salience in party manifestos.
- These investigations find that LLMs perform effectively when applied to qualitative coding tasks, producing estimates that are comparable to human and expert coding in terms of reliability and predictive validity.
- To support implementation, the brief further introduces *quallmer*, an R package designed to make AI-assisted qualitative coding accessible, regardless of prior technical knowledge. A simplified step-by-step guide highlights the key functions.
- Whether adopting AI-powered or traditional human coding, core questions about how we measure political concepts persist. Regardless of technological and methodological development, the basic rule applies: rubbish in, rubbish out.

1 Introduction

Why (is such a method required)?

Interpreting and extracting meaning from unstructured data – for instance, text – is vital for the social sciences. It is not a question of whether to do it, but how. Love them or hate them, LLMs are simply the newest instruments in this long tradition, making it essential to establish clear and robust protocols to ensure their valid and replicable use. Although human coding remains the "gold standard" of qualitative research, it does not scale. Even when someone does have the money (millions) and time (years) required to conduct large-scale qualitative coding projects (i.e., The Manifesto Project, details here: https://manifesto-project.wzb.eu/), the same financial constraints making such datasets a rarity prevent independent tests of their reproducibility.

In response to this problem, text-as-data approaches (TaDa) were developed. In their simplest form, these supervised machine learning methods involve counting tokens (i.e., appearances of single words), but more advanced methods which account for vector representations of text units are also available. These innovations transformed empirical research in the social sciences, yet they also miss nuances such as the intensity of a speaker's stance. Moreover, TaDa methods rest on simplified and often flawed assumptions about language, most notably the idea that word order does not matter for meaning. Therefore, there exists a trade-off between scalability and statistical assumptions that strip away the meaning of language as it is spoken.

The solution proposed by Benoit et al. [2026] is to use LLMs. The method involves two key components: (1) ask the LLM to summarise the author's position, and (2) ask the LLM to answer a series of questions about the author's position. This approach differs from previous applications of LLMs to qualitative coding tasks as the authors advocate for texts to be treated "holistically", rather than going sentence by sentence and aggregating results. The expansion of content windows in new LLMs models allows the authors to request that LLMs read and summarise the whole text before scoring it. Machines cannot equal humans, but through treating the texts holistically, the LLMs achieve "functional equivalence" to the process followed by human coders.

Where (else can this take us)?

So far, Benoit et al. [2026] and Benoit and Laver [2026] have used this approach to assess both issue positions and issue saliency in party manifestos. Through selecting these topics the authors are able to leverage human coded data provided by The Manifesto Project and expert survey benchmarks to validate the LLM-based estimates. Their findings suggest that LLMs perform qualitative coding tasks effectively, producing outputs comparable to those produced by humans in terms of reliability and predictive validity.

Future work could (and should) push this much further across a wider set of research questions. As the validity of LLM use has been extensively documented by Ken Benoit and colleagues, researchers should feel confident using them to address questions where validation data are thinner or less comprehensive, drawing on an already refined and pre-validated workflow. But perhaps the more urgent task is a sustained conversation about how we measure political concepts across different kinds of texts, independent of the methodological approach adopted. As demonstrated by Benoit and Laver [2026], manifestos appear better suited to capturing issue positions than issue saliency. The basic rule of any analytical method still applies: rubbish in, rubbish out.

2 The method:

What (does the workflow look like)?

When advocating for the adoption of AI-powered text analysis, Benoit et al. [2026], construct a step-by-step workflow. This is presented in Figure 1. Due to the rapid development of LLMs and the everchanging technological landscape, it is possible that there is no need for steps 3 and 4 (summarise and scale, respectively) to be conducted separately. Whether LLMs now have the functionality to combine these steps without impacting the results and estimates produced,

remains to be tested. Thus, for now, those looking to implement such a method would be best to follow the workflow described below, subject to further methodological development and refinement.

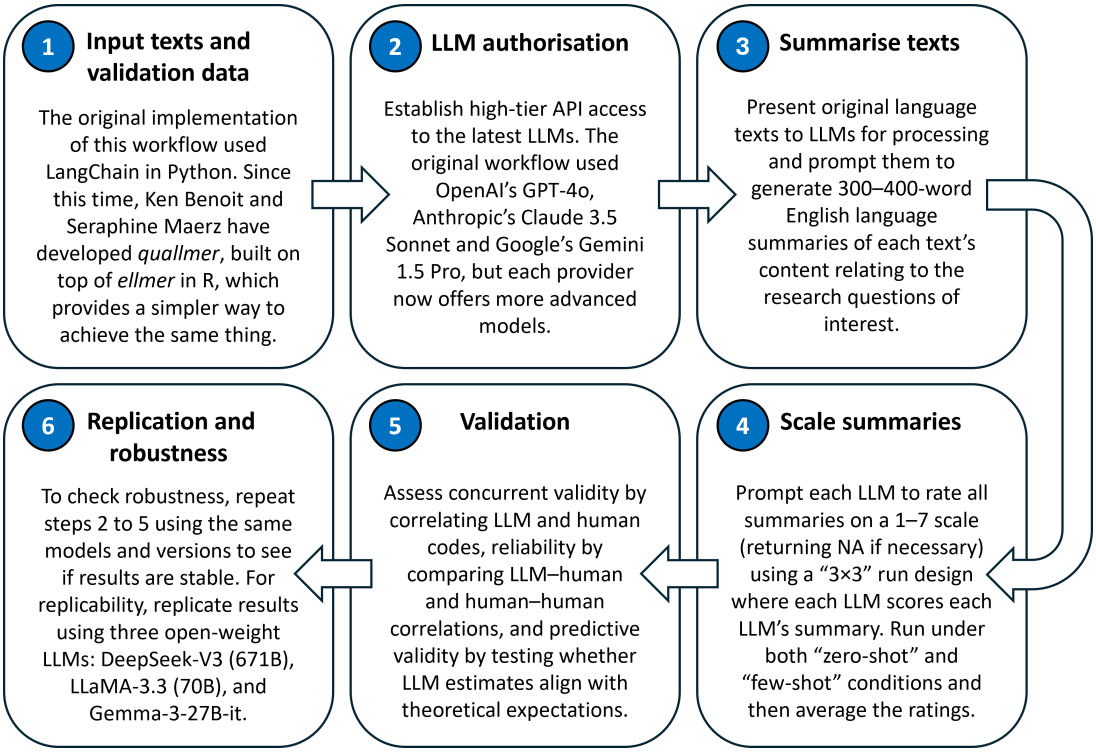

Figure 1: AI-Powered Text Analysis Workflow (Benoit et al. [2026])

How (is this carried out in practical terms)?

As described in the workflow above, Benoit et al. [2026] used Python (specifically, LangChain) for implementation. The later analysis of issue saliency by Benoit and Laver [2026] used an early version of the *ellmer* package in R. Fortunately, *quallmer* has since been built on top of the *ellmer* package by Maerz and Benoit [2026] and is designed to make AI-assisted qualitative coding accessible without requiring deep expertise in R, programming or machine learning.

In addition to built-in codebooks, it allows researchers to create custom codebooks tailored to their specific research questions. Most importantly, it encourages users to adopt best practices by providing one-step functions for evaluating inter-rater reliability, assessing accuracy against gold standards, and comparing results across different models. All of this can be documented using a function for creating audit trails, which are essential for ensuring trustworthiness in qualitative research.

Set up simply requires downloading package from GitHub (`https://quallmer.github.io/quallmer/`) as well as obtaining API access keys for models of interest. The full workflow and quick start guide can be found on the *quallmer* website (provided above).

Replication as a design principle

A result produced by a single model, run once, is a demonstration, not a finding. What distinguishes *quallmer* from simply calling an LLM API directly is that it makes *genuine* replication - repeating the coding exercise with a different tool, not merely re-running the same one - a routine part of the workflow rather than a bespoke undertaking. Because `qlm_code()` treats the model as a single, swappable argument, `qlm_replicate()` can rerun an entire codebook against an alternative model, whether from the same provider (e.g., swapping GPT-4o for GPT-5) or a competing one (e.g., swapping a proprietary OpenAI or Anthropic model for an open-weights model served locally), and `qlm_compare()` then quantifies the agreement between the two sets of results.

This lets researchers ask directly whether an LLM-produced estimate is robust to the choice of instrument, in exactly the way that a result's dependence on a single human coder would invite scrutiny of its stability. Where classical inter-coder reliability asks "would another coder produce the same result?", cross-model, cross-provider replication asks the analogous question of the machine: "would a different model, built differently, trained differently, and served by a different company, produce the same result?" *quallmer* provides tools that make such comparisons trivial to produce, automatically computing reliability and validation statistics appropriate to the level of coded data (categorical, ordinal, interval) as defined by the codebook. As we move increasingly to using LLMs as replacements for human qualitative content analysts, such comparisons are essential to establishing the credibility of the LLM instruments as methodologically valid tools.

Indeed, *quallmer* was built for precisely this purpose: an easy to use but powerful environment for establishing the reliability and validity of LLMs as qualititive analytic tools, as well as refining the codebooks and their applications as part of the research pipeline. When Benoit et al. [2026] first proposed the holistic LLM coding method, *quallmer* did not yet exist, so that analysis was built directly in Python using LangChain rather than through the package. By the time Benoit and Laver had turned to issue salience in Benoit and Laver [2026], published in *West European Politics*, they were already working in the R workflow that *quallmer* now formalises. Since then, *quallmer* has become the tool they reach for exclusively, in current and planned work applying LLMs as qualitative content analysts, and it is the workflow recommended to anyone looking to adopt or extend this approach.

3 Implementation guidelines

Recommended Practice

1. Define codebook. Built-in codebooks are available, or researchers may create their own. - `qlm_codebook()`.
2. Apply the codebook to the text with your LLM of choice. Works with any LLM supported by *ellmer* - `qlm_code()`.
3. Replicate findings with a different model to assess sensitivity to model choices – `qlm_replicate()`.
4. Compare results from the original and replicated output (inter-rater reliability and agreement metrics), as well as with human coding of some form (classification metrics) – `qlm_compare()` and `qlm_validate()`.
5. Create an audit trail to save full decision history (models, parameters, timestamps, and parent-child relationships) alongside results – `qlm_trail()`.

References

Kenneth Benoit and Michael Laver. Measuring issue salience for political parties using LLMs. *West European Politics*, pages 1–22, April 2026. doi: 10.1080/01402382.2026.2637134.

Kenneth Benoit, Scott De Marchi, Conor Laver, Michael Laver, and Jinshuai Ma. Using large language models to analyze political texts through natural language understanding. *American Journal of Political Science*, page ajps.70050, March 2026. doi: 10.1111/ajps.70050.

Seraphine F. Maerz and Kenneth Benoit. *quallmer: Qualitative Analysis with Large Language Models*. GitHub, 2026. URL https://quallmer.github.io/quallmer/. R package version 0.4.0.9000.