From Transformers to the Table: How BERT and LLMs Cook Up Features Differently

NLP_Project

D'Angelo Jacopo (3161719)   Negri Giacomo (3155287)   Omizzolo Matteo (3316152)

Rienth Maximillian (3325056)   Thomopoulos Ioannis (3176145)

Abstract

This study compares LLM and BERT clustering for recipe classification using 25 K recipes. Mistral-7B-Instruct significantly outperformed unsupervised BERT clustering (accuracy: 0.70 vs 0.51; macro F1: 0.68 vs 0.49). The models exhibited distinct feature utilization patterns: BERT prioritized ingredients (average purity 3.8), while the LLM relied most heavily on self-generated terms (2.8). BERT depends on broad, uniform features but struggles with high-variance inputs, whereas the LLM uses sparse, high-leverage signals with fallback behaviour under uncertainty. These findings demonstrate complementary information extraction approaches for effective recipe classification.

1 Introduction

1.1 Background and Motivation

Large Language Models (LLMs) have rapidly become the method of choice for text-classification tasks in Natural Language Processing [17, 27, 22]. Their success stems from the Transformer architecture [30] and pre-training on vast corpora [13].

Transformer-based language models fall into three architectural families: (i) encoder-only pretrained language models (PLMs) such as BERT (Sec. 3.2) ; (ii) encoder-decoder hybrids (not explored here); and (iii) decoder-only LLMs such as Mistral [16] (Sec. 3.4).

To our knowledge, recent work involving PLMs and LLMs has primarily focused on predictive accuracy and clustering performance [26, 27]. However, no prior work has conducted a systematic comparison of an LLM-based classifier and a BERT-based classifier in terms of feature importance.

1.2 Objectives and Scope

Cooking recipes are selected to compare the two classification types because of their inherent ambiguity [26] and semi-structured nature. This creates a classification setting that balances both structure and semantics [10].

Considering this, the following research question is addressed: *"To what extent do LLMs and unsupervised BERT clustering differ in terms of feature utilization in the context of recipe classification?"*

The pipeline (Fig. 1) depicts the process through which we compare the two classifiers.

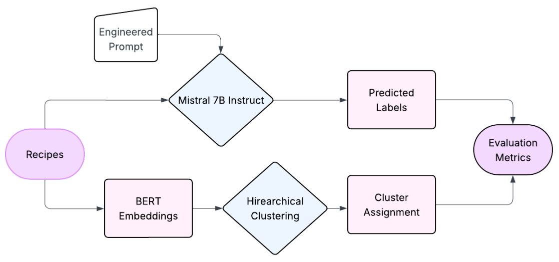


Figure 1: Pipeline comparing LLM classification to BERT clustering.

2 Data and Preprocessing

We reduced the original 2.2M recipe Kaggle dataset [20] to a focused sample by filtering cooking direction lengths to the interquartile range, removing titles with unusual punctuation, and excluding recipes with >50 ingredients (App. 2b) (586K remain). TF-IDF similarity based deduplication of titles through Union-Find algorithm reduced this to 298K recipes. Ingredient filtering retained only the top 90% most frequent terms (1 460 ingredients, 144K recipes). Finally, verb tokenization, POS-tagging, and frequency filtering on those verbs yielded 233 unique cooking direction tokens across 25 758 recipes. This systematic pipeline preserved essential classification information while ensuring computational manageability (App. A.1).

3 Methodology

3.1 Log-Odds as Feature Importance

We use the (smoothed) *log-odds ratio* as our feature-importance score because it combines both *effect size* and *directionality* in a single statistic. Concretely, let $n_{k,v}$ be the raw count of token v in all documents of cluster k, and let α be a small smoothing constant. The log-odds ratio for token v in cluster k is

$$\ell_{k,v} \;=\; \ln \frac{n_{k,v} + \alpha}{\sum_{j \neq k} (n_{j,v} + \alpha)}.$$

Positive values of $\ell_{k,v}$ indicate that v is over-represented in k, negative values that it is under-represented, and the magnitude reflects the strength of its discrimination.

Unlike raw term frequencies or TF–IDF, log-odds compares a token's frequency inside a cluster to its frequency in *all other* clusters, yielding a signed statistic: large positive values highlight tokens that characterise the cluster, while large *negative* values flag tokens that are distinctly under-represented [1]. Making log-odds more relevant for our analysis as it captures both the strength and the direction of each feature's contribution to a label's identity.

3.2 BERT Embeddings

Prior to clustering, a unified representation for each recipe is obtained through a weighted concatenation of title, ingredient, and cooking direction embeddings. For these features, the embeddings (of dimension 384 each) are generated through the "all-MiniLM-L6-v2" BERT transformer [25].

More specifically, for the ingredients and directions embedding $\mathbf{v}_r$ are computed by first fitting a TF–IDF vectorizer (unigrams, $\min_{\mathrm{df}} = 2$, $\max_{\mathrm{df}} = 0.85$) on the cleaned tokens to obtain weights $w_{i,r}$, then mapping each token i to its BERT embedding $\mathbf{e}_i$, and computing

$$\mathbf{v}_r \;=\; \frac{\sum_{i \in r} w_{i,r}\, \mathbf{e}_i}{\sum_{i \in r} w_{i,r}}$$

. While for the title, no weight is applied due to its uniqueness. This process ensures that unique tokens carry greater influence while common ones are downweighted [3].

3.3 Clustering on BERT Embeddings

Starting from the stacked feature vectors v_r $(dim = 384 * 3)$, we apply Uniform Manifold Approximation and Projection (UMAP) [19] to reduce dimensionality to 60 components using cosine similarity as a distance metric.

Agglomerative hierarchical clustering is performed on the 60 UMAP components using Ward's linkage [21]. The clusters are identified by cutting the dendrogram at *k*=7. This k was chosen over the optimal 4 cluster solution (as indicated by the elbow and silhouette scores 3, [11]), as it allowed for a more diverse set of labels for LLM comparison, with a minimal sacrifice in silhouette score.

The "predicted" cluster labels are assigned manually by looking at the most cluster-specific (by log-odds) ingredients, directions and titles (Tab. 5).

3.4 LLM: Mistral-7B-Instruct-v0.2

We employed Mistral-7B-Instruct-v0.2, a 7.3 billion parameter instruction-tuned Transformer [14, 28]. We selected this model based on recent empirical evidence demonstrating that Mistral-based models consistently outperform other open-source LLMs and Machine Learning methods in text classification [16]. This performance advantage makes it particularly suitable for complex classification tasks with textual inputs.

3.5 Prompt Engineering

Prompt engineering is a technique that involves writing instructions in order to guide the LLM toward producing the desired output [6]. Poor prompt engineering can mislead the model, resulting in suboptimal performance [15]. Therefore, a large emphasis is placed on the construction of our prompt. A zero-shot approach with structured reasoning is adopted, where the model is instructed to provide a brief Chain of Thought (COT) before outputting the final answer, improving accuracy and interpretability [5, 18, 31].

The list of recipe labels identified from the clustering of the BERT embeddings is provided to constrain the model's output and minimize off-topic answers [7, 26]. To ensure deterministic results, greedy decoding is employed, with responses limited to 60 tokens due to computational constraints. Lastly, in an attempt to gain some understanding of feature importance, we analyse the model's COT tokens. Self-explanation as a means of reflecting the underlying LLM thought process has been shown to be effective. However, due to its novel nature, its robustness is still in contention [23, 12, 8, 2, 9]. In our case, the input (title, ingredients and directions) should be simple enough to warrant the COT as a basis for feature importance extraction [24]. The combination of all this information led to the creation of the final prompt (App. C).

4 Results

4.1 Results of BERT classification

Table 1: Mean absolute log–odds per cluster (BERT)

Label	**Title**	**Ing.**	**Verb**	**Label Avg.**
Meat & Fish Mains	5.211	4.744	3.350	**4.435**
Salads / Slaws / Dressings	3.716	3.166	2.625	**3.169**
Pasta & Veg Bakes	2.774	2.508	1.867	**2.383**
Soups & One-Pots	2.620	2.557	1.492	**2.223**
Breakfast Bakes & Breads	3.409	4.472	2.221	**3.367**
Drinks & Fruit Treats	3.547	4.575	2.082	**3.401**
Desserts / Cookies / Cakes	4.424	4.420	2.756	**3.867**
Feature Avg.	**3.672**	**3.777**	**2.342**	**3.264**

Each "purity" is the mean log-odds over all tokens in each label. A purity of p implies an $\exp(p)$-fold difference in token odds inside versus outside the cluster (e.g. $p = 3 \Rightarrow \exp(3) \approx 20\times$).

As summarized in Table 1, ingredients (*feature avg. purity* $\approx$ 3.777) and titles ($\approx$ 3.672) both yield substantially higher cluster-specific signals than cooking directions ($\approx$ 2.342), positioning them as more important features.

In the UMAP projection (Fig. 4b), the clusters with the highest *label avg. feature purity* (Meat & Fish Mains, Desserts / Cookies / Cakes, and Drinks Fruit Treats) appear as tight, well-separated "islands," while lower-purity clusters (e.g. Pasta Veg Bakes) form less distinct clouds. This is further reflected in the dendrogram (Fig. 4a), where high-purity groups merge at larger linkage distances (indicating strong cohesion), while low-purity groups fuse earlier.

These overall scores confirm that, on average, ingredients are the strongest cluster-specific signal, followed by recipe titles, with directions contributing the least unique information.

4.2 Results of LLM classification

Keywords from the LLM's COT are mapped into five features analysed by their absolute frequency: ingredients (51.48%), titles (18.73%), directions (14.15%), self-generated terms [1] (10.53%), and in-prompt terms [2] (5.11%). Ingredients dominate the LLM's COT reasoning frequency across recipe labels (Fig. 5).

[1]Tokens not included in the prompt of the model.

[2]Tokens included in the prompt that do not fall under other feature categories.

However, the log-odds analysis in Tab. 2 reveals that frequency does not equal discriminative power. Self-generated terms exhibit the highest discriminative strength (*feature avg. purity* $\approx$ 2.845), followed by titles ($\approx$ 1.916), and ingredients ($\approx$ 0.722). This contrasts sharply with BERT clustering, where ingredients top feature importance, reflecting their different input utilization.

The important role of self-generated terms in classification reflects the LLM's ability to capture culinary features beyond the input. Examples include "dairy-based dessert" (log-odds $\approx$ 8.137), "hot beverage" ($\approx$ 10.356), and "slow cooked" ($\approx$ 7.579) (Tab. 6).

Table 2: Mean absolute log–odds per cluster (LLM)

Label	**Title**	**Ing.**	**Other**	**Label Avg.**
Breakfast Bakes, Breads & Pancakes	1.630	0.203	2.751	**1.528**
Desserts, Cookies & Cakes	2.017	0.913	3.341	**2.090**
Drinks, Punches & Fruit Treats	2.911	1.351	3.510	**2.591**
Salads, Slaws, Dips & Dressings	1.342	0.687	2.286	**1.438**
Pasta & Vegetable Casseroles/Bakes	1.848	0.537	2.398	**1.594**
Soups, Stews & One-Pot Meals	1.595	0.597	2.502	**1.565**
Meat & Fish Mains	2.066	0.766	3.126	**1.986**
Feature Avg.	**1.916**	**0.722**	**2.845**	**1.643**

5 Comparative Error Analysis

5.1 LLM vs. Clustering

The predictions from the BERT clustering and the LLM classification are both assessed against a human gold standard (on a manually labeled random sample of 1 000 recipes) to have an impartial metric of their strengths and weaknesses.

Prediction metrics: The LLM-based classifier outperforms the BERT-based classifier across all labels, raising overall accuracy from 0.51 to 0.70 and macro-average F1 from 0.49 to 0.68 (Tabs. 7 and 8).

5.2 Error Patterns and Category Confusion

A substantial proportion of errors arise from misclassifying label 1 (Breakfast Bakes, Breads & Pancakes) as label 2 (Desserts, Cookies & Cakes). BERT made 80 such directional errors (the opposite error occurs only 6 times). This is supported by the high Jaccard similarity [3] (0.85) between the two categories' vocabularies, since they are so similar, they become virtually indistinguishable to the

[3]Jaccard similarity measures vocabulary overlap between two categories[29]: $J(A, B) = |A \cap B|/|A \cup B|$. We compare different sets of misclassified recipes against their target category vocabularies.

BERT clustering (while the inverse has a Jaccard of 0.42) (Figs. 6a and 6b).

Out of the times that BERT (37 times) and LLM (38 times) both wrongly classify cluster 1 instead of 2, thirty-five are on the same exact recipe (Figs. 7a and 7b). TF–IDF analysis shows that these errors are driven by disproportionate weights of key terms, such as *flour, baking powder, oil, eggs* (higher weight), and *cream, chocolate, cheese* (lower weight) in misclassified recipes (Fig. 8).

In negative log-odds cases, where the LLM mentions terms typically associated with other categories, one of two extremes emerge: 1. A very sophisticated correct classification 2. A blatantly wrong classification.

For instance, “ground beef” (-3.304 log-odds) appears in breakfast classifications, while “olive oil” (-3.844 log-odds) emerges in dessert categories. These cases represent either sophisticated contextual reasoning (e.g., breakfast burritos, Mediterranean desserts) or systematic classification errors where the model correctly identifies relevant terms but misassigns categories.

5.3 Feature Importance Comparison

In the LLM-based classifier, we observe a significant difference in the mean log-odds of all features between correctly (higher log-odds) and incorrectly (lower log-odds) classified recipes (FDR-adjusted[4] $p_{\text{mean}} < 0.05$, Fig. 10b). However, the difference in variance was not significant for any feature (all adjusted $p_{\text{var}} > 0.05$)(Fig. 10a, 14a). This is consistent with a “fallback” behavior: when the LLM is confident in its answers its features are highly specific to the chosen cluster, while under uncertainty it falls back to vocabulary common to all clusters (low log-odds). Moreover, the non-significant variance difference indicates that when the LLM is uncertain features specificity drops uniformly.

By contrast, the BERT-embedding clustering shows the inverse relationship, with no significant difference in mean log-odds yet a significantly higher variance in the misclassified set (FDR-adjusted $p_{\text{var}} < 0.05$)(Fig. 13a). In other words, the clusterer succeeds only when feature scores are stable and homogeneous, but struggles on recipes whose log-odds vary widely. This indicates that unsupervised embedding clusters rely on tight, consistent feature patterns.

Robustness to Outliers We repeated the log-odds comparisons after applying a robust outlier filter to control for potential biases and validate our previous results (1.5×IQR; Fig. 11, 12, 13, 14). Both models still exhibit the statistical significance previously identified.

6 Conclusions

In conclusion, the feature utilisation of LLMs and unsupervised clustering differs to a considerable extent, both in terms of feature importance and in classification landscape. For BERT ingredients are the most important feature, while for the LLM the self-generated terms are. Moreover, in terms of classification landscape BERT leans on broad, uniform features, while the LLM bets on sparse, high-leverage signals. Together, these contrasts highlight that effective recipe classification is not about which model is “better,” but about recognising that each extracts a different layer of information from the same text, breadth from embeddings and precision from generative reasoning. Future systems should be designed to exploit both perspectives.

Limitations and Future Work

Computational Constraints: the classification was restricted to a 25 K sampled recipes and a 60-token response cap, limiting COT depth and potentially under-utilising the model’s reasoning capacity.

Single-Model Choices: we only evaluate one LLM model (Mistral-7B-Instruct-v0.2), one BERT model and one clustering method. Thus, results may not generalise to the broader "LLM" and "Unsupervised BERT Clustering" categories. Future research could consider analysing a series of each model to gain broader insights.

Self-Explanation Assumption: the notion that LLM self-explanation can reflect the underlying thought process is still in contention in the literature. If new literature emerges invalidating this assumption, our interpretations and comparisons may be biased.

Differences across LLM temperatures: this report adopts a 0 temperature strategy, but analysing different levels of stochasticity may also produce interesting results. Future research could explore these avenues.

[4]False Discovery Rate correction using Benjamini-Hochberg procedure to control for multiple comparisons [4]

References

[1] Muhammad Abdul-Mageed et al. Log-odds ratio: Going beyond simple term frequencies to characterize textual categories, 2020. QCRI ACUA Blog.

[2] Sule Anjomshoae, Lili Jiang, and Kary Främling. Visual explanations for dnns with contextual importance. In *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021)*, pages 335–342. SCITEPRESS–Science and Technology Publications, 2021.

[3] Mohit Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations (ICLR)*, 2017.

[4] Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57(1):289–300, 1995.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[6] Sondos Mahmoud Bsharat, Aidar Myrzakhan, and Zhiqiang Shen. Principled instructions are all you need for questioning llama-1/2, gpt-3.5/4, 2023.

[7] Devichand Budagam, Ashutosh Kumar, Mahsa Khoshnoodi, Sankalp KJ, Vinija Jain, and Aman Chadha. Hierarchical prompting taxonomy: A universal evaluation framework for large language models aligned with human cognitive principles. *arXiv preprint arXiv:2406.12644*, 2024.

[8] Daniel C. Elton. Self-explaining ai as an alternative to interpretable ai. *Patterns*, 2(9):100275, 2021.

[9] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2301.00375*, 2023.

[10] Mansi Goel, Pallab Chakraborty, Vijay Ponnaganti, Minnet Khan, Sritanaya Tatipamala, Aakanksha Saini, and Ganesh Bagler. Ratatouille: A tool for novel recipe generation, 2024. Equal contribution: †. Corresponding author: Ganesh Bagler.

[11] Adria Binte Habib. Elbow method vs silhouette co-efficient in determining the number of clusters, February 2021. ResearchGate.

[12] Shiyuan Huang, Siddarth Mamidanna, Shreedhar Jangam, Yilun Zhou, and Leilani H. Gilpin. Can large language models explain themselves? a study of llm-generated self-explanations. *arXiv preprint arXiv:2305.16666*, 2023.

[13] Yunpeng Huang, Jingwei Xu, Junyu Lai, Zixu Jiang, Taolue Chen, Zenan Li, Yuan Yao, Xiaoxing Ma, Lijuan Yang, Hao Chen, Shupeng Li, and Penghao Zhao. Advancing transformer architecture in long-context large language models: A comprehensive survey. *ACM Computing Surveys*, 2024. To appear.

[14] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[15] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[16] Arina Kostina, Marios D. Dikaiakos, Dimosthenis Stefanidis, and George Pallis. Large language models for text classification: Case study and comprehensive review. *arXiv preprint arXiv:2403.17641*, 2024.

[17] Menglin Liu and Ge Shi. Poliprompt: A high-performance cost-effective llm-based text classification framework for political science, 2024. mliliu@ucdavis.edu, geshi@ucdavis.edu.

[18] Yiheng Liu, Hao He, Tianle Han, Xu Zhang, Mengyuan Liu, Jiaming Tian, Yutong Zhang, Jiaqi Wang, Xiaohui Gao, Tianyang Zhong, Yi Pan, Shaochen Xu, Zihao Wu, Zhengliang Liu, Xin Zhang, Shu Zhang, Xintao Hu, Tuo Zhang, Ning Qiang, Tianming Liu, and Bao Ge. Understanding llms: A comprehensive overview from training to inference. *Neurocomputing*, 556:129190, 2024.

[19] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2018. arXiv preprint arXiv:1802.03426.

[20] Paul Mooney. Recipenlg (cooking recipes dataset). https://www.kaggle.com/datasets/paultimothymooney/recipenlg, 2021. Acknowledgements for the dataset https://www.aclweb.org/anthology/2020.inlg-1.4.pdf, https://github.com/Glorf/recipenlg, https://recipenlg.cs.put.poznan.pl/.

[21] Fionn Murtagh and Pierre Legendre. Ward's hierarchical clustering method: Clustering criterion and agglomerative algorithm. *Journal of Classification*, 31(3):274–295, 2014.

[22] Vasileios Ntinopoulos, Hector Rodriguez Cetina Biefer, and Igor Tudorache. Large language models for data extraction from unstructured and semi-structured electronic health records: A multiple model performance evaluation. *BMJ Health and Care Informatics*, 32(1), 2025. License: CC BY-NC 4.0.

[23] Daniel Omeiza, Sule Anjomshoae, Helena Webb, Marina Jirotka, and Lars Kunze. From spoken thoughts to automated driving commentary: Predicting and explaining intelligent vehicles' actions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 1412–1421. IEEE, 2021.

[24] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

[25] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019. arXiv preprint arXiv:1908.10084.

[26] Nazmus Sakib, G. M. Shahariar, Md. Mohsinul Kabir, Md. Kamrul Hasan, and Hasan Mahmud. Towards automated recipe genre classification using semi-supervised learning. *PLOS ONE*, 20(1):e0317697, 2025.

[27] Gautam Kishore Shahi and Oliver Hummel. On the effectiveness of large language models in automating categorization of scientific texts, 2024. `{g.shahi,o.hummel}@hs-mannheim.de`, ORCID: Shahi, Hummel.

[28] Mistral AI Team. Mistral-7b-instruct-v0.2. Hugging Face Model Hub, 2024.

[29] Gonzalo Travieso, Alexandre Benatti, and Luciano da F. Costa. An analytical approach to the jaccard similarity index. *arXiv preprint arXiv:2303.17319*, 2023.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30. Curran Associates, Inc., 2017.

[31] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 24824–24837, 2022.

A Data and Preprocessing

A.1 Step-by-step guide

A Kaggle dataset of 2 231 141 [20] recipes was used, containing fields: `title`, `ingredients` (string and list), `directions`, and `source` with `link`. Due to its size, full text preprocessing was infeasible. To reduce dimensionality, recipes were filtered by `directions` length, keeping only those within the interquartile range, justified by the skewed length distribution and presence of outliers (App. A.2).

Titles were lowercased, stripped and duplicates removed, reducing the dataset to about 660 000 records. Recipes with over 50 ingredients were excluded to avoid excessive complexity, and those with duplicate ingredients were discarded (App. A.2). Finally, recipes with titles containing `?,!,*,...,#` were removed because assumed to be not from reliable sources, leaving 586 357 entries.

Subsequently, TF-IDF vectors of titles and nearest neighbor search with cosine similarity identified recipes with over 80% similarity as duplicates or near-duplicates. A Union-Find (disjoint set) structure efficiently clustered and filtered these recipes (App. A.3) The dataset was reduced to 298 331 rows.

Since the text pre-processing goal was to shrink the dataset to an analyzable subset while retaining relevant information driving recipe classification. Every step targets noise, sparsity, or ambiguity.

Ingredient strings were lowercased, stripped of punctuation and stopwords, and erased if token length was smaller than 3. Since fuzzy-matching was unreliable, a frequency cut was adopted: retaining recipes with ingredients in the top 90% of the global frequency distribution. Rows dropped from 298 331 to 144 759 and unique ingredients from 57 579 to 1 460, removing rare elements like *corn hominy"*, *zwack"*, or *"sour rye bread"*. Fewer ingredients means denser co-occurrence counts and a smaller BERT embedding matrix, which affect classification performance.

Because directions relies on syntax to make sense, a word-level tokenization was applied. Verbs were then POS-tagged and lemmatized to map cases like *mix*, *mixed*, and *mixing* to the root. Applying the 90% frequency cut reduced unique verbs from 6 908 to 233 and rows to 25 758, making analysis feasible.

Checks ensured verbs that are also ingredients once lemmatized were excluded, while retaining the record, reducing ambiguity. Mistagged verbs like *summer*, *containers*, or *regular* were removed. Only nine ingredients were lost due to this, now totaling 1 451.

In summary, noisy text was compressed into a dense, semantically coherent dataset while retaining key classification information.

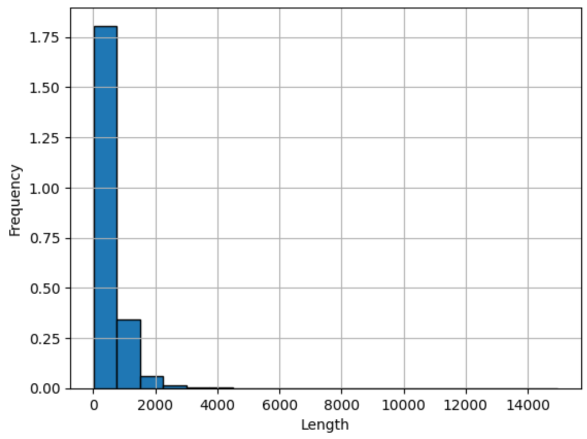
A.2 `description` and `ingredients`

Table 3: Descriptive statistics of `NER_len`

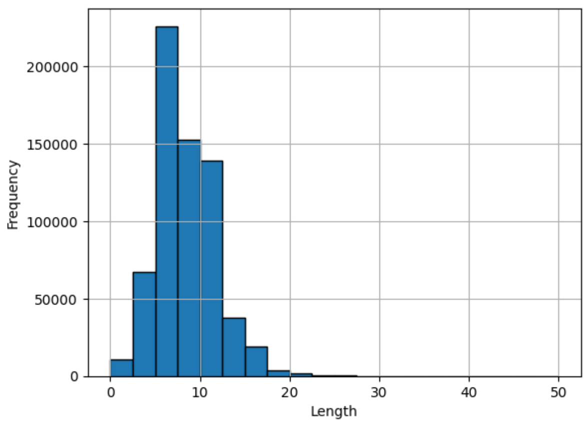
Statistic	Value
count	658 717.000000
mean	8.144030
std	3.338155
min	0.000000
25%	6.000000
50%	8.000000
75%	10.000000
max	402.000000

Table 4: Descriptive statistics of `directions_len`

Statistic	Value
count	2.231142×10^6
mean	5.051099×10^2
std	4.524093×10^2
min	5.000000×10^0
25%	2.210000×10^2
50%	3.710000×10^2
75%	6.410000×10^2
max	1.497900×10^4



(a) Distribution of length of `description` (in Million).



(b) Distribution of number of `ingredients`.

Figure 2: Distributions of `description` length and number of `ingredients`. Frequencies measured in millions.

A.3 Similar Recipes filtering and Union-Find

In oder to retain one representative for every close cluster of very similar titles, after vectorization through TF-IDF (sklearn TfidfVectorizer) was implemented and the cosine distance (sklearn cosine_similarity) was computed, a selection through Union-Find algorithm was implemented, setting a similarity threshold to 20%. Given the size of the dataframe of about 600 K records a matrix was unfeasible, therefore the 10 nearest neighborhoods where found (sklearn NearestNeighbors) and the results where then collapse in a graph build through the Union-Find algorithm, hence making use of connected components and disjoint-set data structure.

The Union–Find (or *disjoint-set*) data structure keeps track of which elements belong to the same cluster. Each element starts in its own singleton set. Whenever we discover that two elements are similar, which in the current case mean that two titles have cosine distance < 0.2, we combine (union) their sets by concatenating one to the other leaf. This operation is repeated multiple times so that a forest of shallow trees whose roots act as representatives is created. Two operations are implemented in near constant time due to the use of dictionaries:

- `find(x)`: returns the representative of `x`.
- `union(x, y)`: merges the sets of `x` and `y`.

In our case the algorithm already take in inputs the indexes of neighborhoods for each recipes and the associated cosine similarity.

-
  ```
  def find(x):
      while parent[x] != x:
          parent[x] = parent[parent[x]]
          x = parent[x]
      return x
  ```

  It traverse the tree upwards until the root of the tree is found, which is the representative component of the disjoint-set, and it returns it.

-
  ```
  def union(x, y):
      rootX = find(x)
      rootY = find(y)
      if rootX != rootY:
          parent[rootY] = rootX
  ```

It calls `find(x)` to obtain the representative component of `x` and `y`. If the two roots differ, the root of `y` is reassigned to `x`, and by consequence, merging the two sets, since every element previously rooted at `y` now inherits `x` as representative.

- Neighbour loop: for each recipe, iterate over its neighbours; whenever a neighbour meets the similarity threshold, merge the two corresponding disjoint-sets.
- Cluster assembly: for every recipe, retrieve its representative and place the recipe into the representative's cluster; if the representative is yet to be present in the clusters dictionary, start a new cluster.
- Selection: only the representative of each cluster is kept.

B Unsupervised BERT-classification

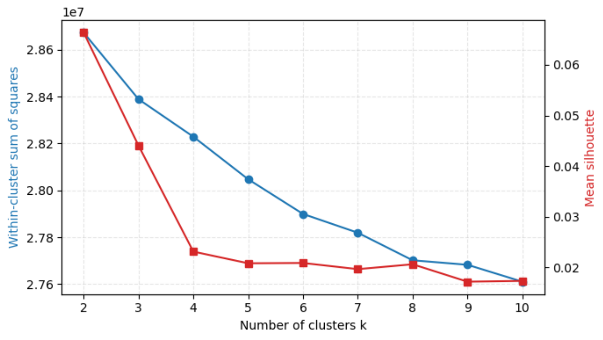
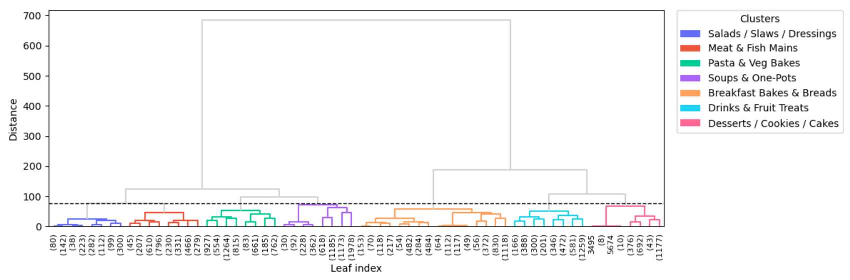
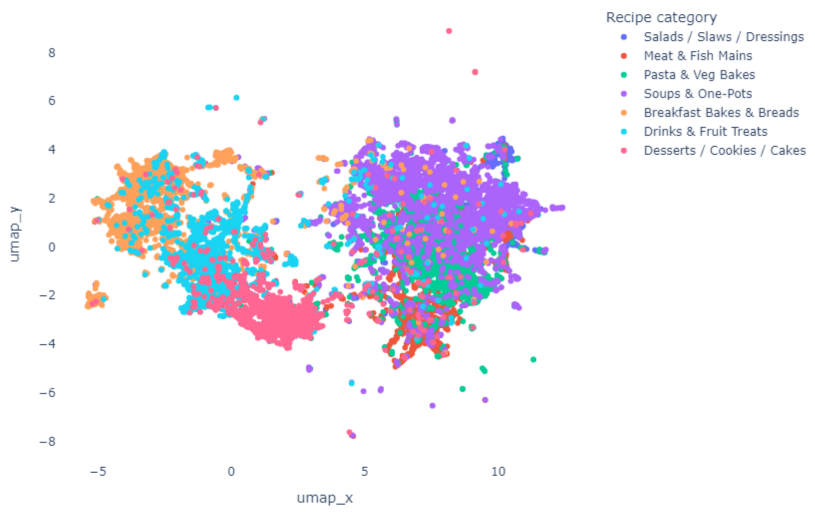


Figure 3: Silhouette Curve for Ward Linkage Clusters



(a) Dendrogram of the 7 clusters



(b) UMAP scatterplot of the 7 clusters

Figure 4: Cluster structure visualised as a dendrogram and in 2-D UMAP space.

B.1 Top Elements per BERT Cluster

Cluster	Top 3 Ingredients (wt.)	Top 3 Verbs (wt.)	Top 3 Title Words (wt.)	Count
Salads, Slaws, Dips & Dressings	couscous (1.85), cucumbers (1.70), vinaigrette (1.59)	dress (0.75), toss (–0.51), refrigerate (–0.69)	shaved (4.76), goddess (4.61), salat (4.42)	2937
Pasta & Vegetable Casseroles/Bakes	rigatoni (5.45), fettuccine (2.46), pizza (2.10)	crisp (0.16), grate (0.07), drain (–0.23)	brussels (5.65), taters (5.45), rigatoni (5.21)	5390
Soups, Stews & One-Pot Meals	bone (2.14), hocks (2.14), okra (1.78)	saute (0.33), absorb (0.26), simmer (0.08)	lima (5.01), basmati (4.61), burgundy (4.61)	3380
Meat & Fish Mains	swordfish (4.89), tilapia (2.52), shoulder (2.51)	baste (1.19), grill (1.06), marinate (0.78)	flank (5.99), teriyaki (5.99), bass (4.76)	3496
Breakfast Bakes, Breads & Pancakes	gum (4.42), yeast (2.31), bicarbonate (2.02)	sift (2.83), rise (1.37), knead (1.29)	doughnuts (5.65), play (5.21), wacky (4.76)	4656
Drinks, Punches & Fruit Treats	grenadine (2.99), gin (2.57), grapefruit (2.41)	strain (0.39), sieve (–0.32), dissolve (–0.47)	grapefruit (5.65), margaritas (5.01), wassail (4.89)	2161
Desserts, Cookies & Cakes	oreo (3.44), brownie (2.99), cookies (2.36)	sweeten (0.65), whip (0.42), frost (–0.15)	pralines (5.65), mores (5.45), ganache (5.38)	3738

Table 5: Top 3 ingredients, verbs, and title words by cluster, with cluster sizes

C LLM Prompt for Recipe Classification

Listing 1: Prompt template for LLM recipe classification

```
prompt = f"""
<s>[INST]
Classify this recipe into exactly ONE best-fitting category:
0 = Not food
1 = Breakfast Bakes, Breads & Pancakes (pancakes, waffles, cornbread)
2 = Desserts, Cookies & Cakes (sweet treats, pies, puddings, custards)
3 = Drinks, Punches & Fruit Treats (beverages, smoothies, juices)
4 = Salads, Slaws, Dips & Dressings (dips/spreads/condiments)
5 = Pasta & Vegetable Casseroles/Bakes (pasta, veg bakes)
6 = Soups, Stews & One-Pots (soups, chilis, broths)
7 = Meat & Fish Mains (protein-centered dishes)

RULES:
- ALL dips  category 4, even if hot or meaty
- ALL soups  category 6, even if fruit-based or meaty
- Sweet puddings/custards  category 2
- Mixed meat+veg: if pasta/veg dominate  5, else  7
- Breads as sides (cornbread, hush puppies)  1

Answer EXACTLY in this format (no extras):
RATIONALE: <310 keywords>
NUMBER = <digit 07> <Category-Name>

Recipe: {recipe_name}
Ingredients: {ingredients}
Cooking verbs: {verbs}
[/INST]</s>
"""
```

D LLM class frequency

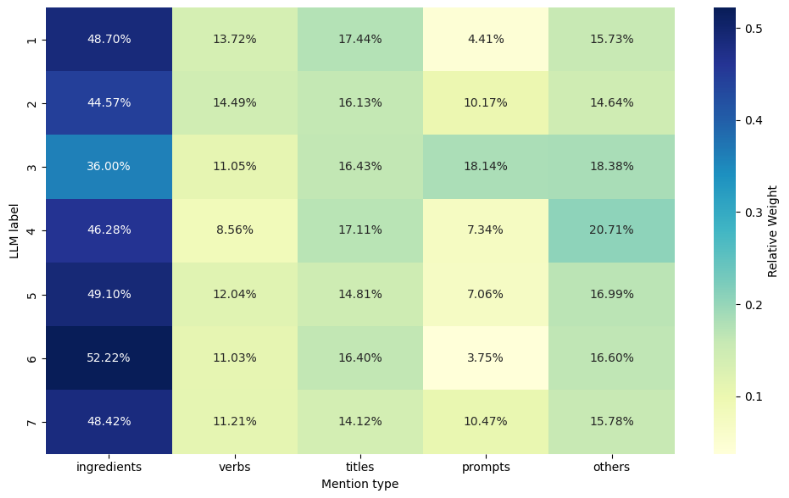
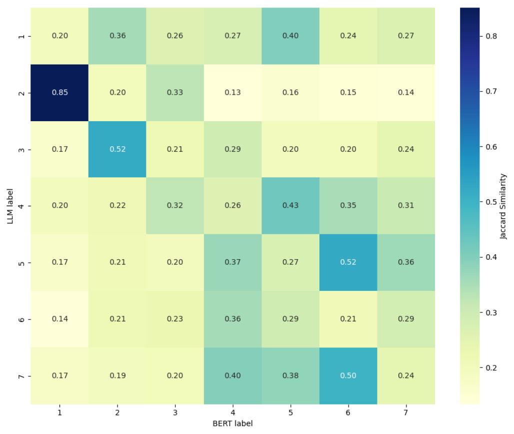
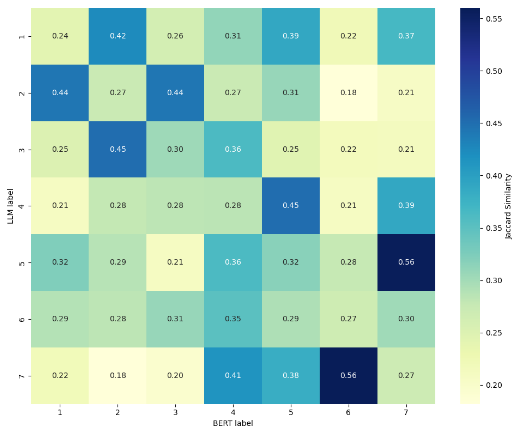


Figure 5: Relative frequency of keyword categories in LLM classes

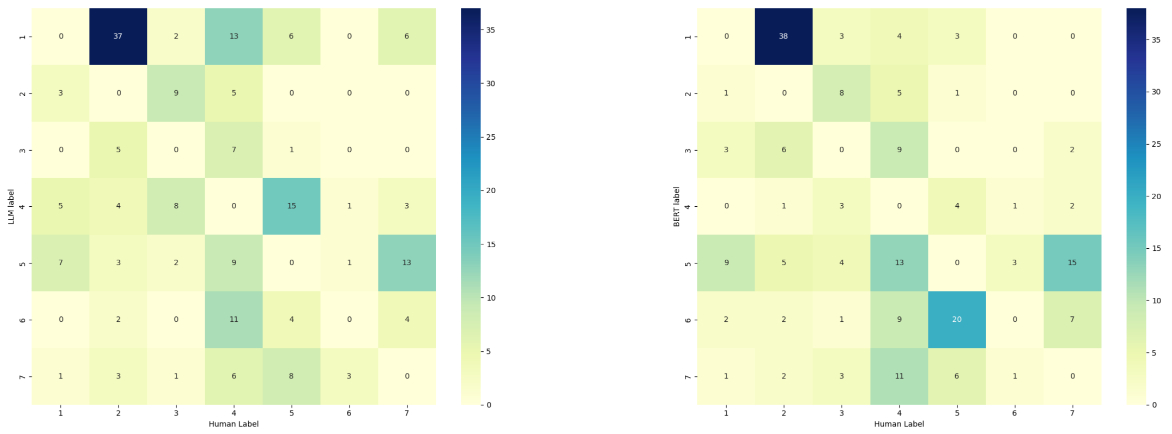


(a) Right LLM classification and wrong BERT



(b) Wrong LLM classification and right BERT

Figure 6: Jaccard Similarity with respect to Human labeled recipes.



(a) Wrong LLM against Human Label

(b) Wrong BERT against Human Label

Figure 7: Jaccard Similarity with respect to Human labeled recipes.

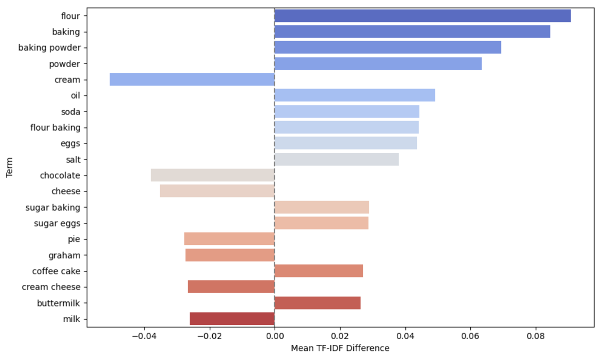


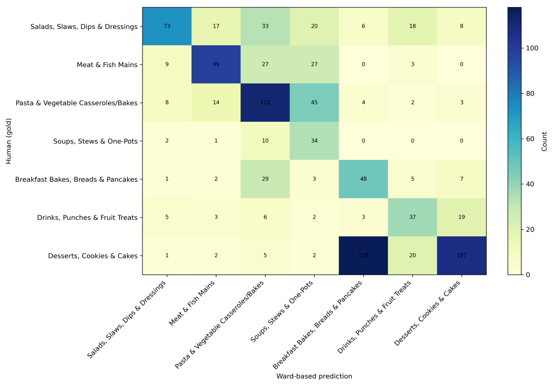
Figure 8: TF-IDF for misclassified Human label 2 as 1: both LLM and BERT

E Top Elements per LMM Cluster

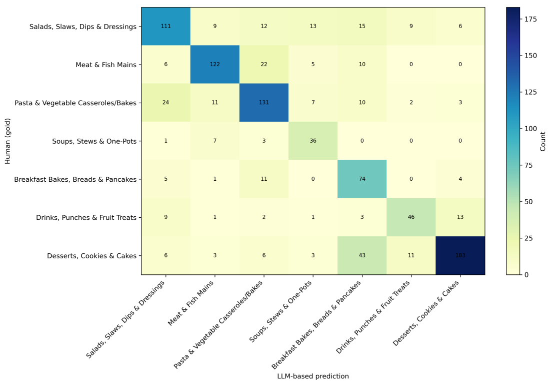
Cluster	Ingredients (wt.)	Verbs (wt.)	Titles (wt.)	Prompts (wt.)	Other (wt.)
Salads, Slaws, Dips & Dressings	peanut oil (5.899), radicchio (5.899), extra-virgin olive oil (5.899)	reserve (6.199), stem (6.199), dress (3.491)	cole (7.585), pecans (7.298), condiment (7.115)	slaws (5.660), dressings (4.905), salads (4.319)	cheese dip (8.148), pickling (7.880), brine (7.617)
Pasta & Vegetable Casseroles/Bakes	fettuccine (6.994), spaghetti (6.920), linguine (6.753)	end (2.164), absorb (1.953), repeat (1.883)	pilaf (8.772), macaroni (7.856), primavera (6.940)	pasta (3.442), chicken (1.259), vegetable (1.116)	marinara sauce (7.026), fries (6.803), asian dumplings (6.803)
Soups, Stews & One-Pot Meals	ham bone (6.945), rice wine (5.695), fresh tomatoes (5.695)	return (6.554), halve (6.554), taste (6.554)	broth (8.460), bisque (7.544), stock (7.544)	one-pots (4.921), soups (2.528), stews (1.929)	cold soup (9.189), chowder (7.754), savory broth (7.244)
Meat & Fish Mains	chops (7.231), rabbit (6.864), pork roast (6.746)	rub (6.823), continue (6.131), medium (3.697)	burger (8.296), meatloaf (8.201), hamburgers (7.978)	shrimp (7.989), chilis (6.891), salmon (6.199)	mexican-style dish (8.272), burger patties (7.733), slow cooked (7.579)
Breakfast Bakes, Breads & Pancakes	rye flour (5.928), yeast (5.911), bagels (5.641)	lay (6.501), light (5.809), plate (5.809)	pancake (8.824), waffle (8.131), popovers (8.051)	morning (8.886), breads (7.340), breakfast (5.142)	yeast (9.037), toasted (7.867), baked goods (7.867)
Drinks, Punches & Fruit Treats	simple syrup (9.091), crushed ice (8.541), grenadine (8.318)	shake (4.411), strain (4.229), enjoy (3.286)	smoothie (10.379), drink (9.182), juice (9.182)	drinks (7.572), punches (6.520), beverages (5.578)	hot beverage (10.356), alcoholic drink (10.143), limeade (8.452)
Desserts, Cookies & Cakes	graham cracker crust (8.187), chocolate pudding (7.360), caramels (6.955)	incorporate (6.826), fit (5.443), like (5.443)	fudge (9.726), frosting (9.491), toffee (7.971)	custards (8.683), desserts (5.046), puddings (3.588)	candy (9.166), chocolate dessert (8.894), dairy-based dessert (8.137)

Table 6: Top ingredients, verbs, title words, prompts, and other words by cluster.

F Prediction Analysis



(a) BERT vs. human gold



(b) LLM vs. human gold

Figure 9: Model predictions compared with human gold classifications.

Table 7: Per-Label Classification Metrics for BERT

Category	Precision	Recall	F1-score	Support
Salads, Slaws, Dips & Dressings	0.737	0.417	0.533	175
Meat & Fish Mains	0.717	0.600	0.653	165
Pasta & Vegetable Casseroles/Bakes	0.505	0.596	0.546	188
Soups, Stews & One-Pots	0.256	0.723	0.378	47
Breakfast Bakes, Breads & Pancakes	0.268	0.505	0.350	95
Drinks, Punches & Fruit Treats	0.435	0.493	0.462	75
Desserts, Cookies & Cakes	0.743	0.420	0.536	255
Accuracy	0.510	0.510	0.510	0.51
Macro avg	0.523	0.536	0.494	1000
Weighted avg	0.602	0.510	0.526	1000

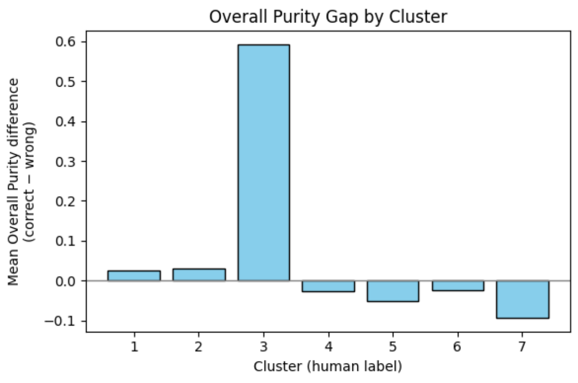
Table 8: Classification Report by Category for LLM

Category	Precision	Recall	F1-score	Support
Salads, Slaws, Dips & Dressings	0.685	0.634	0.659	175
Meat & Fish Mains	0.792	0.739	0.765	165
Pasta & Vegetable Casseroles/Bakes	0.701	0.697	0.699	188
Soups, Stews & One-Pots	0.554	0.766	0.643	47
Breakfast Bakes, Breads & Pancakes	0.477	0.779	0.592	95
Drinks, Punches & Fruit Treats	0.676	0.613	0.643	75
Desserts, Cookies & Cakes	0.876	0.718	0.789	255
Accuracy	0.703	0.703	0.703	0.703
Macro avg	0.680	0.707	0.684	1000
Weighted avg	0.728	0.703	0.709	1000

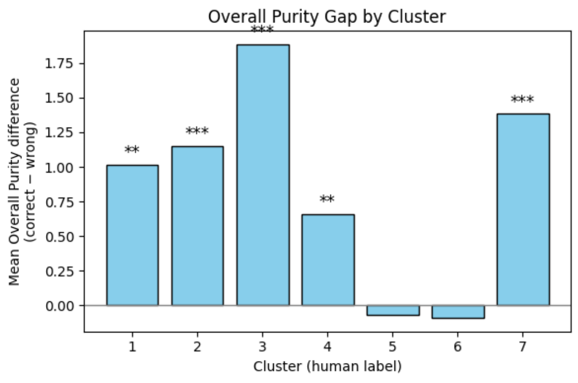
G Feature Analysis

Table 9: All Significant Log-Odds Feature Differences by Cluster

Cluster	Metric	Mean (Correct)	Mean (Wrong)	Mean difference	t-stat	p	p_{adj}
1	avg_ing_logodds	0.036399	0.013168	0.023231	3.2995	1.43e-3	1.004e-2
2	avg_ing_logodds	0.017669	0.036859	–0.019190	–5.2061	4.05e-7	9.00e-6
2	avg_verb_logodds	0.006557	0.016356	–0.009799	–2.9437	3.55e-3	1.861e-2
3	avg_title_logodds	3.572198	1.816360	1.755838	2.3366	2.22e-2	8.484e-2
5	avg_ing_logodds	0.015936	–0.000052	0.015988	4.4063	1.86e-5	1.961e-4
6	avg_ing_logodds	0.008051	–0.006322	0.014373	2.5010	2.42e-2	8.484e-2

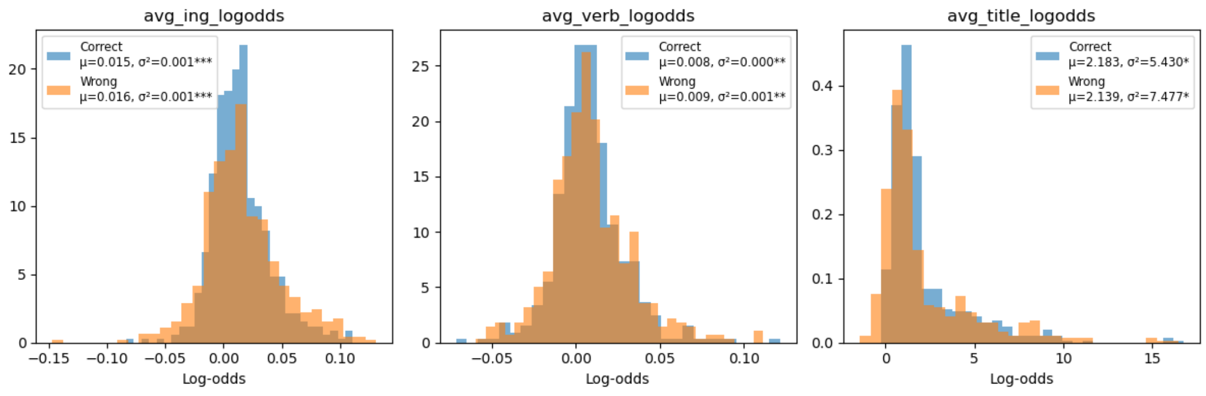


(a) BERT feature purity

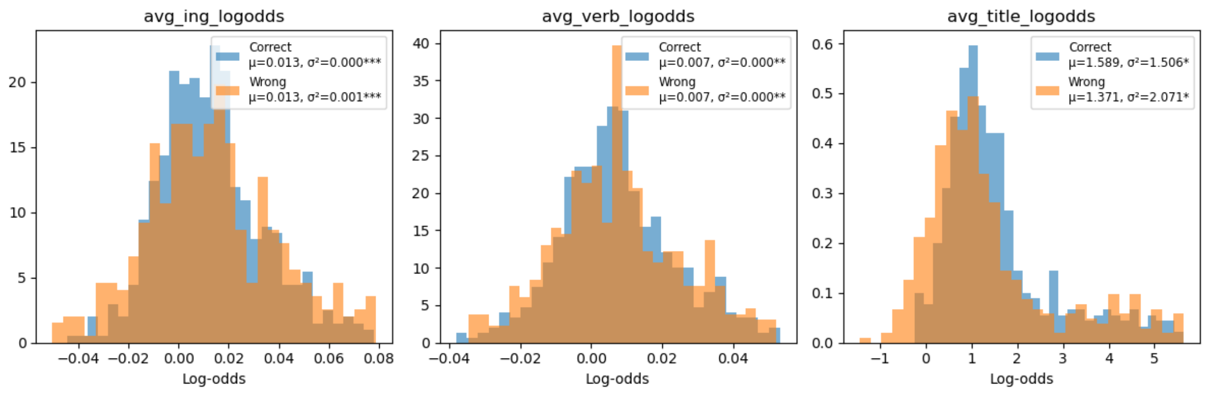


(b) LLM feature purity

Figure 10: Feature-purity profiles for the two models.

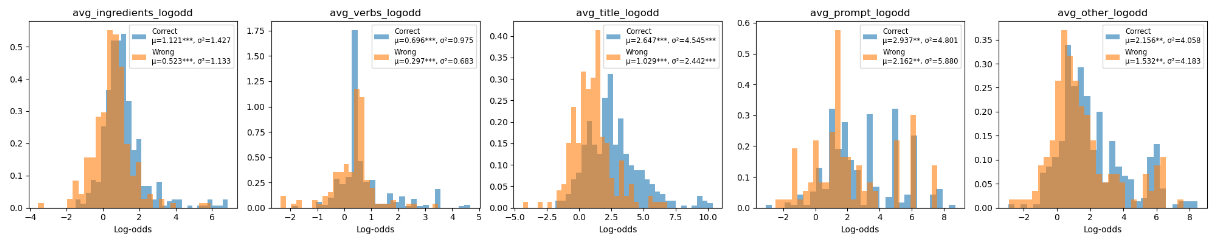


(a) BERT histogram before outlier removal

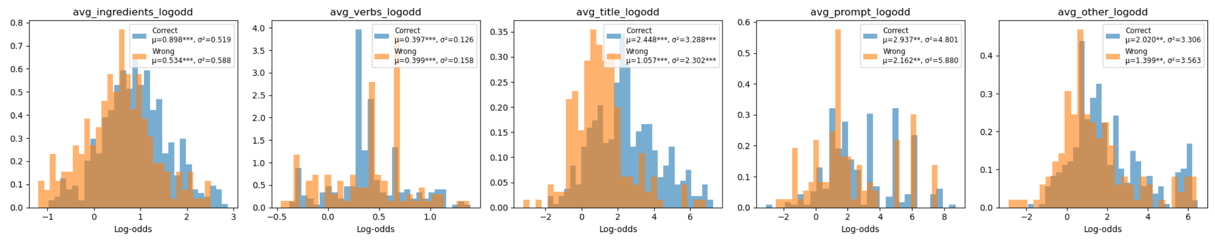


(b) BERT histogram before outlier removal

Figure 11: BERT log-odds histograms (correct vs. wrong), before and after dropping extreme values (1.5×IQR). Mean μ and variance σ^2 are shown in the legend.

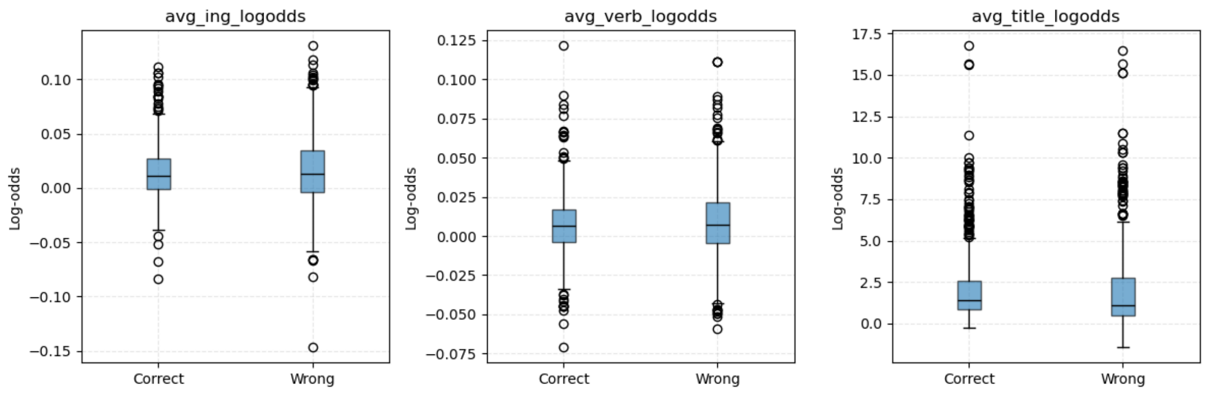


(a) LLM histogram before outlier removal

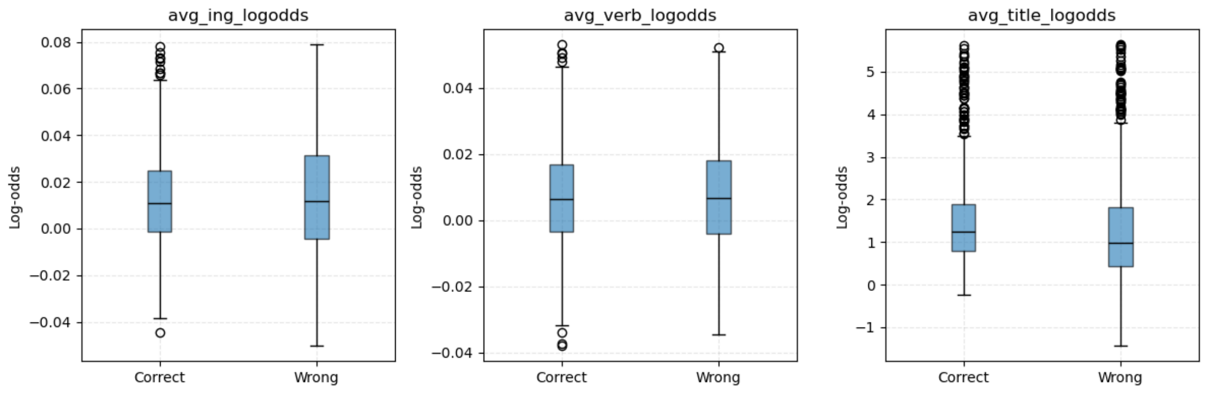


(b) LLM histogram after outlier removal

Figure 12: LLM log-odds histograms (correct vs. wrong), before and after dropping extreme values (1.5×IQR). Mean μ and variance σ^2 are shown in the legend.

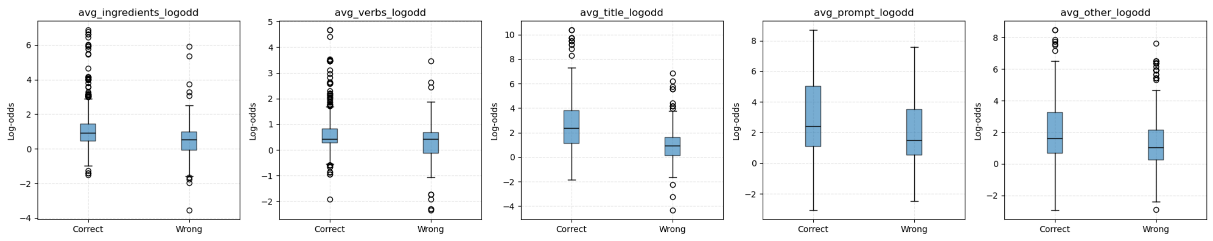


(a) BERT boxplot before outlier removal

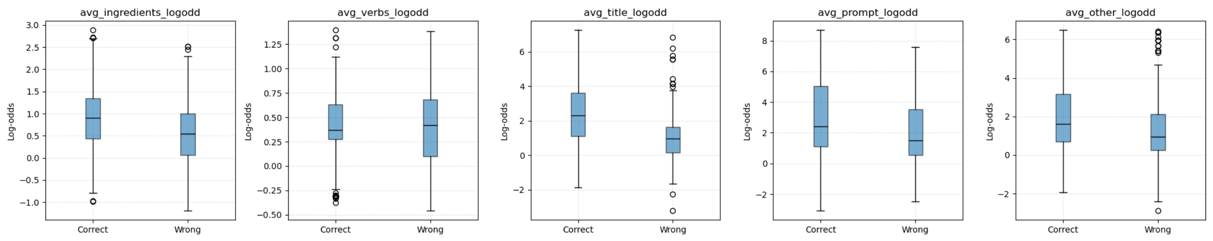


(b) BERT boxplot after outlier removal

Figure 13: Comparison of BERT log-odds boxplots (correct vs. wrong) before and after dropping outliers (1.5×IQR).



(a) LLM boxplot before outlier removal



(b) LLM boxplot after outlier removal

Figure 14: Comparison of LLM log-odds boxplots (correct vs. wrong) before and after dropping outliers (1.5×IQR).