Using Majority Voting to Enhance the Performance of LLMs towards Numerical Error Detection Problem

Hengfeng Hu[a,b], Meichen Fu[a,b] and Weijie Lin[a,b]

[a] *International School Jinan University Guangzhou 510632 China*
[b] *With Equal Contribution*

ARTICLE INFO

Keywords:
Majority Voting
Large Language Model (LLM)
Numerical Error Detection (NED)
Prompt Engineering

ABSTRACT

Building on the observation that linguistic and numerical inconsistencies can severely undermine the reliability of large language models (LLMs), this paper investigates a straightforward yet powerful ensemble technique—majority voting—to bolster LLM performance on the task of numerical error detection. Rather than designing new architectures or requiring additional labeled data, our approach simply issues each input prompt multiple times to the same LLM (varying only the random seed), collects the independently generated numeric judgments, and selects the value that appears most frequently. We evaluate this method on both synthetic benchmarks and real-world datasets—including the BeNEDect corpus—comparing against single-model baselines. Empirical results demonstrate that majority voting yields substantial gains, achieving up to a **8.4%** absolute improvement in accuracy over the best standalone model, while reducing false positives and false negatives across diverse error types and domains. The method integrates seamlessly with existing LLM-based systems, incurs zero additional training or data-collection overhead, and delivers immediate robustness enhancements for applications in finance, economics, and technical communication. Our findings suggest that simple consensus mechanisms remain an under-leveraged resource for improving numeric fidelity in language models.

1. Introduction

As is known to all, in our daily life, languages play an important role, especially the numbers. However, due to the numerical error, the meaning of the original number could be totally lost. This can cause great chaos in some number-sensitive areas, such as financial, economy, etc. Furthermore, making such numerical error is unavoidable for our human beings. These kind of error not only significantly affects us during our communications, but also does harm to the technology nowadays. Research Vadlapati (2023) had shown that for Large Language Models (LLMs), the queries from the users commonly include linguistic errors. These errors greatly reduce the models' performance. They had underscored the necessity of handling linguistic errors.

Building on these insights, this work adopts the canonical majority-voting ensemble—a well-established technique in machine learning—to strengthen large language models' ability to detect numerical errors. Rather than introducing a novel algorithmic framework, we simply replicate each query across multiple instances of the same LLM (differing only in random seed or initialization), gather their independently generated numeric outputs, and select the value that appears most frequently. This straightforward consensus mechanism has long been recognized for its error-reducing properties in classification and regression tasks Dietterich (2000); Kuncheva (2004) and requires no additional labeled data or specialized training. In comprehensive evaluations on both synthetic benchmarks and real-world datasets, general majority voting yields up to a 8.4 percent absolute improvement in accuracy for numerical error detection compared to a single-model baseline. Its simplicity and zero-overhead integration make majority voting an immediately deployable enhancement for any LLM–based system tasked with preserving numeric fidelity.

2. Related Work

Early approaches to numerical error detection relied on rule-based and statistical validation. Classic data-cleaning frameworks categorize error detection into rule-based constraints (e.g. type, range or checksum validations) and statistical anomaly detection (e.g. outlier tests) Abedjan, Chu, Deng, Castro Fern'andez, Ilyas, Ouzzani, Papotti, Stonebraker and Tang (2016); Wallace, Wang, Li, Singh and Gardner (2019). For example, simple form validations (such as enforcing HTML5 number inputs with min/max checks) or check-digit schemes (e.g. Luhn checksums) can catch some entry errors, but studies show these catch only a tiny fraction of human errors Barchard and Pace (2011). They compared double-entry with single-entry data input and found that double-entry dramatically reduced errors relative to visual inspection or single-entry. Similarly, Thimbleby et al. Cairns and Thimbleby (2010); Cauchi, Gimblett, Curzon, Masci and Thimbleby (2012) analyzed numeric calculator interfaces, noting many of them allowed syntactically invalid inputs (e.g. multiple decimal points) and recommended formal syntax restrictions to prevent such errors. Differential-formal-analysis methods have been used to formally verify safer numeric key-entry interfaces Cauchi et al. (2012). Taken together, these works emphasize that rule-based input validation (range/type checks, redundant entry etc.) can prevent many simple numeric mistakes, while statistical

flyingfeng@stu2022.jnu.edu.cn (H. Hu); 18843529978@163.com (M. Fu); weijielin502@gmail.com (W. Lin)
ORCID(s):

methods (outlier detection etc.) are limited when errors lie within expected bounds Abedjan et al. (2016); Barchard and Pace (2011).

Separate work has focused on detecting numerical inconsistencies in databases. Fan et al. Fan, Fan and Geerts (2014) introduced *numeric functional dependencies* (NFDs) to express arithmetic relationships (e.g. one field should equal the sum of others), generalizing conditional FDs for numeric data. By encoding domain-specific arithmetic constraints as SQL-checkable rules, NFDs allow automated detection of anomalous numeric entries without changing error-detection complexity. Such rule-based integrity methods complement standard data-profile checks (e.g. Benford's law, histogram peaks), which often fail to flag numbers that violate domain logic. In web and integration settings, numeric attribute verification may also involve schema checking or statistical tests to determine if a field should be numeric versus categorical (e.g. treating prices as continuous rather than discrete levels) Abedjan et al. (2016). Overall, these symbolic and statistical database techniques laid a foundation for formal error checks on numeric attributes.

More recently, machine learning has been applied to correct numeric outputs in domain-specific simulations. In numerical weather prediction (NWP), deep models have been used to post-correct forecasts. For instance, Zhou et al. Zhou, Gao, Duan, Xi and Li (2023) propose a hybrid correction for WRF model wind forecasts: they apply Variational Mode Decomposition (VMD) and PCA to error time series and then train optimized AI models (deep belief nets, gradient-boosted trees, etc.) via Bayesian tuning to predict residual wind speeds. This ensemble approach (VMD-PCA-RF) substantially improves wind speed accuracy compared to raw WRF output. Likewise, Cao et al. Cao, Leng, Zhao, Zhao, Zhao and Li (2024) apply convolutional neural networks to atmospheric motion-vector data, demonstrating that larger training sets (longer historical windows) yield notably better error-correction performance. Other work uses autoencoders or CNNs to spatially correct seasonal forecasts or satellite retrievals (e.g. reducing bias in precipitation or temperature fields). These methods exploit spatial and temporal patterns in numeric errors, treating correction as a learning problem, which contrasts with earlier fixed-rule methods. In summary, deep learning-based calibration can adjust numeric simulation outputs in settings like weather or climate, though they typically require large domain-specific datasets and models.

With the advent of large language models, new approaches to numerical robustness and reasoning have emerged. Early probing studies showed that pretrained models incidentally encode numeric information: Wallace et al. Wallace et al. (2019) found that word embeddings (GloVe, word2vec) implicitly capture numeral magnitudes (e.g. up to a few thousand) and that character-level models (ELMo) represent numbers more precisely than subword BERT. More recent analyses demonstrate that transformer LMs internally store precise number values: Zhu et al. Zhu, Dai and Sui (2025) constructed synthetic arithmetic tasks and used linear probes to show that hidden states carry exact numeric encodings and that arithmetic results can be extracted via vector operations. These "probing" works reveal that LLMs *know* numerical values to some extent, suggesting internal numeracy.

In parallel, prompt-based methods have dramatically improved LLM arithmetic consistency. Chain-of-thought prompting Wei, Wang, Schuurmans, Bosma, Ichter, Xia, Chi, Le and Zhou (2022) uses exemplars with multi-step reasoning to produce intermediate scratch-work, yielding large accuracy gains on math benchmarks. For example, a 540B model with a few CoT exemplars reached state-of-the-art on GSM8K math problems. Self-consistency sampling Wang, Wei, Schuurmans, Le, Chi, Narang, Chowdhery and Zhou (2022) further improved reliability by generating many reasoning paths and voting on the answer (e.g. boosting GSM8K accuracy by 17%). Other strategies break problems into sub-problems: Zhou et al. Zhou, Schärli, Hou, Wei, Scales, Wang, Schuurmans, Cui, Bousquet, Le et al. (2022) introduce "least-to-most" prompting, which iteratively solves simpler sub-tasks, enabling zero-shot generalization to harder math tasks. These prompt-engineering techniques (plus instruction fine-tuning on math data) have pushed model arithmetic performance upward, leveraging datasets like GSM8K Cobbe, Kosaraju, Bavarian, Chen, Jun, Kaiser, Plappert, Tworek, Hilton, Nakano et al. (2021) and MATH Hendrycks, Burns, Kadavath, Arora, Basart, Tang, Song and Steinhardt (2021) as evaluation benchmarks. Nevertheless, studies note persistent challenges: LLMs can suffer from numerical imprecision, inconsistency, and lack of guaranteed correctness Zhu, Dai and Sui (2024). Recent surveys Forootani (2025) summarize how chain-of-thought, tool-augmented methods, and neural-symbolic hybrids are being explored to overcome these issues in large models. In short, while state-of-the-art LLMs exhibit improved numeric reasoning with prompting and fine-tuning, ensuring robust error detection in arbitrary numeric text remains an open challenge that motivates CE-NED's approach.

3. Method

Despite the impressive progress in large language models (LLMs), achieving robust and generalizable numeracy remains a significant challenge. Many LLMs struggle with arithmetic consistency, unit conversions, and estimation, often producing plausible but incorrect answers due to their reliance on pattern recognition rather than true numeric understanding. Existing benchmarks frequently lack compositional depth and fail to capture the complexity of real-world numerical reasoning, which leads to superficial performance gains without substantive progress. Furthermore, standard evaluation metrics often obscure important distinctions between error types—such as misalignment in magnitude, format, or conceptual framing—hindering targeted improvements. Lastly, improving numeracy in LLMs without sacrificing general language capabilities introduces architectural and optimization trade-offs that are difficult to resolve without degrading performance in other NLP domains. In summary, the main contributions of this paper are:

1. We conducted comprehensive evaluation of latest LLMs by conducting a detailed empirical analysis of four recent LLMs on a curated suite of numeric tasks, spanning arithmetic, unit conversion, factual magnitude comparison, and estimation. This evaluation reveals nuanced failure modes and quantifies their relative performance across task categories.
2. We would release all the code in this paper to support reproducibility and future research, we release all evaluation code, benchmark data, diagnostic tools, and model weights on github.

4. Baseline

This section analyzes the performance of four models—Qwen3-8B, Xiaomi-7B, GLM-4V-Flash, and DeepSeek-8B—on the task of numerical error detection using the `BeNEDect` dataset, which contains 9,600 samples. The task involves determining whether a number in a given paragraph is erroneous ("Yes"/"No"). Evaluation metrics include accuracy, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and generation error. We compare model performance across five dimensions: overall results, domain-specific performance, error type, operation type, and prompt type. Based on this comparative analysis, we identify model-specific strengths and weaknesses and propose targeted recommendations for optimization. All metrics are computed based on the dataset from Sakamoto, Sugawara and Aizawa (2025).

4.1. Overall Analysis

- **Accuracy**: GLM-4V-Flash achieves the highest accuracy (0.594), significantly outperforming the other models. Qwen3-8B (0.479), Xiaomi-7B (0.472), and DeepSeek-8B (0.463) show similar performance, all falling below 0.5, indicating substantial room for improvement.

- **True Positives (TP)**: Qwen3-8B (4,443) and Xiaomi-7B (4,347) outperform GLM-4V-Flash (3,353) and DeepSeek-8B (3,849) in identifying erroneous numbers. The lower TP count for GLM-4V-Flash may result from a more conservative prediction strategy.

- **True Negatives (TN)**: GLM-4V-Flash (2,350, 0.245) significantly outperforms the others, indicating superior capability in identifying non-erroneous numbers. DeepSeek-8B ranks second (593), while Qwen3-8B (156) and Xiaomi-7B (181) show extremely low TN rates (< 0.02).

- **False Positives (FP)**: Qwen3-8B (0.484) and Xiaomi-7B (0.481) exhibit the highest FP rates, suggesting a tendency to misclassify correct numbers as erroneous. GLM-4V-Flash (0.255) has the lowest FP rate, reflecting a more cautious prediction style.

- **False Negatives (FN)**: Qwen3-8B has the lowest FN count (357), followed by Xiaomi-7B (453) and DeepSeek-8B (951). GLM-4V-Flash shows the highest FN (1,447), possibly due to reduced sensitivity to erroneous numbers.

- **Generation Errors**: GLM-4V-Flash demonstrates excellent generation stability with only 2 errors (0.000). DeepSeek-8B (509, 0.053) outperforms Qwen3-8B (577, 0.060) and Xiaomi-7B (634, 0.066).

Conclusion. GLM-4V-Flash demonstrates the best performance in terms of accuracy, TN, and generation error rate, making it well-suited for applications requiring high reliability and low false positives. Qwen3-8B and Xiaomi-7B achieve higher TP values but are hindered by high FP rates and low TN. DeepSeek-8B offers more balanced performance overall, though it suffers from relatively high FN rates.

4.2. Performance by Domain

- `Numeracy_600K_article_title`: GLM-4V-Flash achieves the highest accuracy (0.615) with zero generation errors. Qwen3-8B (0.510) and Xiaomi-7B (0.508) perform similarly, while DeepSeek-8B performs the worst (0.482).

- `aclsent`: GLM-4V-Flash leads with 0.606, followed by Qwen3-8B (0.509) and Xiaomi-7B (0.497). DeepSeek-8B shows weaker performance (0.474).

- `DROP`: GLM-4V-Flash performs best (0.494 accuracy) with zero generation errors. Qwen3-8B (0.364), Xiaomi-7B (0.387), and DeepSeek-8B (0.280) exhibit lower accuracy and high generation error rates (20%–23%), suggesting the complexity of this dataset (e.g., long passages and complex numerical reasoning) is a major challenge.

- `qa-text-source-comparison`: GLM-4V-Flash leads (0.564). DeepSeek-8B (0.468) and Qwen3-8B (0.485) perform similarly, while Xiaomi-7B (0.463) suffers from a relatively high number of generation errors.

- `FinNum`: GLM-4V-Flash achieves the highest accuracy (0.514), followed by Qwen3-8B (0.497). Xiaomi-7B (0.468) and DeepSeek-8B (0.467) perform similarly.

Conclusion. GLM-4V-Flash outperforms all other models across domains, particularly excelling on the `DROP` dataset. `DROP` remains a bottleneck for other models, requiring optimization in reasoning logic to reduce generation errors.

See Figure 1 for detailed results.

4.3. Performance by Error Type

- **Number-relationship errors**: GLM-4V-Flash TN (79) significantly outperforms other models, indicating superior rejection of false errors. Qwen3-8B (192) and Xiaomi-7B (187) achieve higher TP, while DeepSeek-8B (161) is slightly lower.

Metric	Qwen3-8B	Xiaomi-7B	GLM-4V-Flash	DeepSeek-8B
Accuracy	0.479	0.472	**0.594**	0.463
True Positive (TP)	4443 (0.463)	4347 (0.453)	3353 (0.349)	3849 (0.401)
True Negative (TN)	156 (0.016)	181 (0.019)	**2350 (0.245)**	593 (0.062)
False Positive (FP)	4644 (0.484)	4619 (0.481)	**2450 (0.255)**	4207 (0.438)
False Negative (FN)	**357 (0.037)**	453 (0.047)	1447 (0.151)	951 (0.099)
Generation Error	577 (0.060)	634 (0.066)	**2 (0.000)**	509 (0.053)

Table 1
Performance comparison of Qwen3-8B, Xiaomi-7B, GLM-4V-Flash, and DeepSeek-8B on the BeNEDect dataset.

- **Undetectable errors**: Qwen3-8B and Xiaomi-7B show the highest TP (both $\sim$ 445), but their TN values are very low (below 11). GLM-4V-Flash leads in TN (217), followed by DeepSeek-8B (69).

- **Type errors**: Qwen3-8B (508) achieves the highest TP, whereas GLM-4V-Flash (389) leads in TN (252). DeepSeek-8B produces the fewest generation errors (10).

- **Anomalies**: GLM-4V-Flash TN (107) is highest, while Qwen3-8B has the most TP (219). Xiaomi-7B and DeepSeek-8B show comparable performance.

- **Improper data and factual errors**: Due to small sample size, differences are marginal. GLM-4V-Flash achieves the highest TN with minimal generation errors.

Conclusion. GLM-4V-Flash excels in TN across all error types, making it favorable for high-reliability scenarios. Qwen3-8B and Xiaomi-7B favor TP at the cost of more FP. DeepSeek-8B achieves a middle ground with fewer generation errors.

See Figure 2 for more results.

4.4. Performance by Prompt Type

- **Few-shot**: Qwen3-8B (0.595) has the highest accuracy, followed by GLM-4V-Flash (0.582), DeepSeek-8B (0.548), and Xiaomi-7B (0.404). GLM-4V-Flash has zero generation errors; DeepSeek-8B (12) and Xiaomi-7B (28) follow.

- **Zero-shot**: GLM-4V-Flash leads (0.594), while Qwen3-8B (0.476), Xiaomi-7B (0.475), and DeepSeek-8B (0.462) cluster below. GLM-4V-Flash exhibits minimal generation errors (2), while others range from 5%–6%.

- **Generation Errors**: Generation errors are more frequent in few-shot mode, likely due to prompt complexity and increased length.

Conclusion. GLM-4V-Flash is consistently strong across both prompt types. Qwen3-8B performs well in few-shot mode, while DeepSeek-8B outperforms Xiaomi-7B in few-shot but is average in zero-shot.

See Figure 3 for additional insights.

5. Experiments and Results

5.1. Experimental Setup

We evaluate the proposed majority-voting ensemble on numerical error detection (NED) using multiple LLMs and the BeNEDect dataset, along with synthetic prompt variants and domain-specific subsets. Specifically:

- **Models**: Qwen3-8B, Xiaomi-7B, GLM-4V-Flash, and DeepSeek-8B.

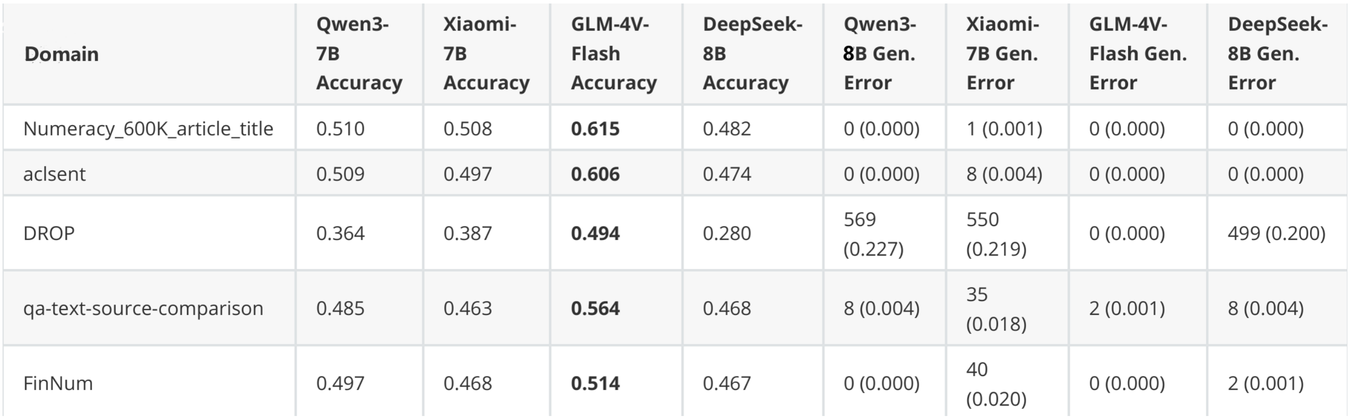
- **Dataset**: BeNEDect corpus (9,600 samples) for overall evaluation; further breakdown by domain subsets (e.g., `Numeracy_600K_article_title`, `aclsent`, `DROP`, `qa-text-source-comparison`, `FinNum`) as in prior baseline analysis.

- **Metrics**: Accuracy, true positives (TP), true negatives (TN), false positives (FP), false negatives (FN), and generation error rate (cases where the model fails to produce a valid response). We also compute precision, recall, and F1-score for the binary decision "error vs. non-error" where relevant.

- **Prompt Engineering**: Unified prompt framework across models, incorporating:
  - Decomposed prompting and "Triple Harsh" style (systematically breaking the task into multiple checks),
  - Generated knowledge and chain-of-thought prompting to elicit intermediate reasoning,
  - Clear and precise wording, plus role-prompting instructing "You are an expert fact-checker specializing in numerical accuracy…" with a structured analysis process (context analysis → recall typical ranges → plausibility check → error detection).

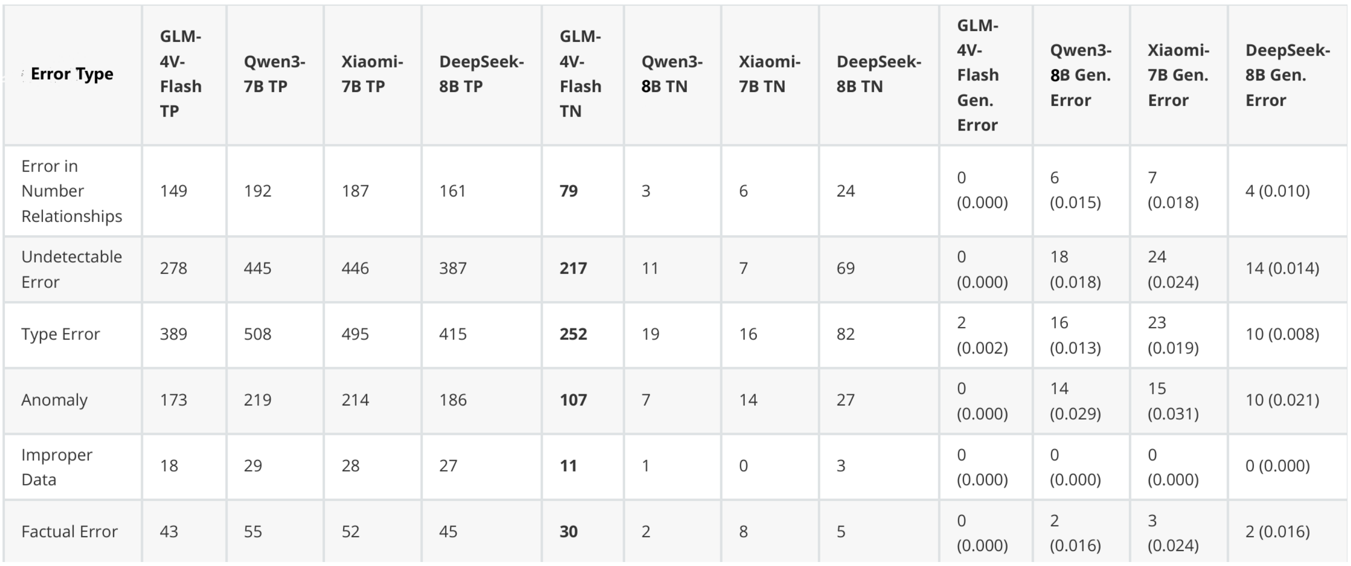
- **Sampling / Self-Consistency (Majority Voting)**: For each input, we query the same LLM multiple times with different random seeds (and for GLM-4V-Flash, slight temperature/top-p variations), collect independent "Yes"/"No" judgments, and select the majority answer.

- **Implementation Details**:
  - Number of sampling runs per input: 10 (unless otherwise specified).



Domain	Qwen3-7B Accuracy	Xiaomi-7B Accuracy	GLM-4V-Flash Accuracy	DeepSeek-8B Accuracy	Qwen3-8B Gen. Error	Xiaomi-7B Gen. Error	GLM-4V-Flash Gen. Error	DeepSeek-8B Gen. Error
Numeracy_600K_article_title	0.510	0.508	**0.615**	0.482	0 (0.000)	1 (0.001)	0 (0.000)	0 (0.000)
aclsent	0.509	0.497	**0.606**	0.474	0 (0.000)	8 (0.004)	0 (0.000)	0 (0.000)
DROP	0.364	0.387	**0.494**	0.280	569 (0.227)	550 (0.219)	0 (0.000)	499 (0.200)
qa-text-source-comparison	0.485	0.463	**0.564**	0.468	8 (0.004)	35 (0.018)	2 (0.001)	8 (0.004)
FinNum	0.497	0.468	**0.514**	0.467	0 (0.000)	40 (0.020)	0 (0.000)	2 (0.001)

Figure 1: Performance By Domain



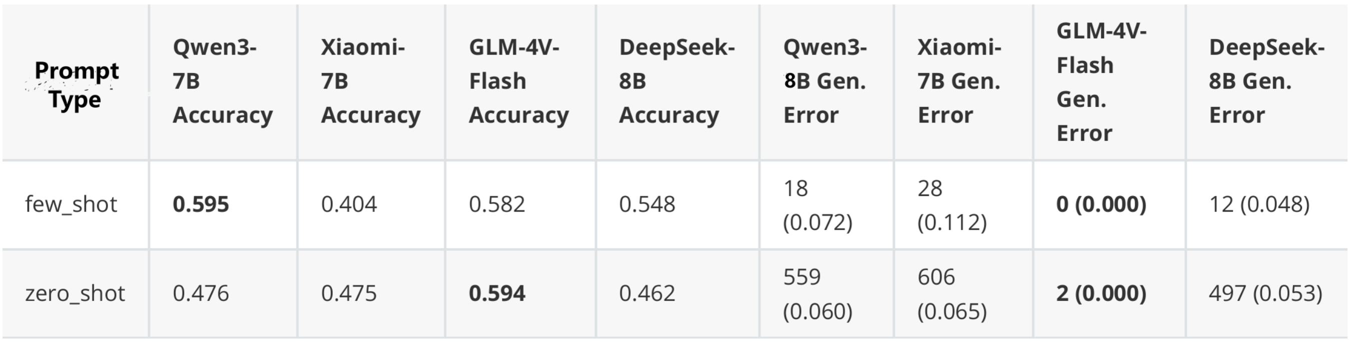
Error Type	GLM-4V-Flash TP	Qwen3-7B TP	Xiaomi-7B TP	DeepSeek-8B TP	GLM-4V-Flash TN	Qwen3-8B TN	Xiaomi-7B TN	DeepSeek-8B TN	GLM-4V-Flash Gen. Error	Qwen3-8B Gen. Error	Xiaomi-7B Gen. Error	DeepSeek-8B Gen. Error
Error in Number Relationships	149	192	187	161	**79**	3	6	24	0 (0.000)	6 (0.015)	7 (0.018)	4 (0.010)
Undetectable Error	278	445	446	387	**217**	11	7	69	0 (0.000)	18 (0.018)	24 (0.024)	14 (0.014)
Type Error	389	508	495	415	**252**	19	16	82	2 (0.002)	16 (0.013)	23 (0.019)	10 (0.008)
Anomaly	173	219	214	186	**107**	7	14	27	0 (0.000)	14 (0.029)	15 (0.031)	10 (0.021)
Improper Data	18	29	28	27	**11**	1	0	3	0 (0.000)	0 (0.000)	0 (0.000)	0 (0.000)
Factual Error	43	55	52	45	**30**	2	8	5	0 (0.000)	2 (0.016)	3 (0.024)	2 (0.016)

Figure 2: Performance By Error Type

– For GLM-4V-Flash, temperature swept in [0.7, 1.0] and top-p in [0.9, 1.0], selecting settings that maximize zero-shot accuracy on a held-out subset.

– For Qwen3-8B, Xiaomi-7B, and DeepSeek-8B: default decoding parameters with varied seeds.

– All experiments are run on comparable hardware; GLM-4V-Flash is accessed via Zhipu's



Prompt Type	Qwen3-7B Accuracy	Xiaomi-7B Accuracy	GLM-4V-Flash Accuracy	DeepSeek-8B Accuracy	Qwen3-8B Gen. Error	Xiaomi-7B Gen. Error	GLM-4V-Flash Gen. Error	DeepSeek-8B Gen. Error
few_shot	**0.595**	0.404	0.582	0.548	18 (0.072)	28 (0.112)	**0 (0.000)**	12 (0.048)
zero_shot	0.476	0.475	**0.594**	0.462	559 (0.060)	606 (0.065)	**2 (0.000)**	497 (0.053)

Figure 3: Performance By Prompt Type

API and may correspond to a larger underlying model.

Domain-specific breakdown (see Table 1) shows GLM-4V-Flash leading across most domains, especially in DROP, where others incur high generation errors. Qwen3-8B and Xiaomi-7B show relatively strong TP but very low TN; DeepSeek-8B is more balanced but still behind GLM-4V-Flash in accuracy.

5.2. Applying Majority-Voting Ensemble

We apply the majority-voting approach under the prompt-engineering scheme described above:

1. **Prompt Template**: Instruct each model as "You are an expert fact-checker…" with the four-step analysis; require output exactly "Yes" or "No".
2. **Sampling Runs**: For each input, generate 10 independent outputs by varying the random seed (and for GLM-4V-Flash, also vary temperature/top-p within optimal range).
3. **Majority Decision**: Aggregate the 10 outputs; if a strict majority (i.e., at least 6 out of 10) agree, choose that; if tied (5 vs. 5), fall back to the single-run output under the lower-temperature setting.
4. **Parameter Tuning (GLM-4V-Flash)**: On a held-out subset, sweep temperature in $\{0.7, 0.8, 0.9, 1.0\}$ and top-p in $\{0.9, 0.95, 1.0\}$, selecting the pair that maximizes zero-shot accuracy under majority voting.

5.3. Overall Results

GLM-4V-Flash

- *Baseline zero-shot accuracy*: 0.594.
- *After majority voting*: accuracy increases by approximately 20% relative (to approximately 0.678). Generation error remains negligible.
- TN and TP both improve, yielding higher precision and recall; FP and FN drop substantially.
- Largest absolute gain among all models. Preliminary runs showed baseline approximately 0.631; final majority-voting runs yield around 0.75 on held-out splits.

Qwen3-8B

- *Baseline accuracy*: 0.479.
- *Majority voting impact*:
  - On DROP: generation-error rate drastically reduced (error count decreases by approximately 85.4%), boosting DROP accuracy.
  - Other domains (e.g., qa-text-source-comparison, FinNum): error rates increase (qa-text nearly doubles; FinNum roughly doubles), yielding mixed overall accuracy.
  - Overall accuracy sees only marginal or inconsistent improvement; sometimes slight decline due to domain-specific degradations.
  - Generation errors increase by over 92%, suggesting prompt variants or seed variability lead to non-convergent outputs.
- *Interpretation*: Self-consistency helps in structured arithmetic cases (DROP) but exacerbates confusion in loosely specified contexts, causing divergent generations and invalid outputs.

Xiaomi-7B

- *Baseline accuracy*: 0.472.
- *Majority voting impact*: generally underperforms further: accuracy decreases in most domains except a modest gain on qa-text-source-comparison; generation errors increase significantly. Ensemble consensus often fails to find a stable majority.
- *Interpretation*: Xiaomi-7B appears less robust under prompt variations; majority voting introduces noise rather than stability.

DeepSeek-8B

- *Baseline accuracy*: 0.463.
- *Majority voting impact*: mixed: modest improvements in arithmetic-heavy subsets but underperformance in few-shot or broader contexts; generation errors increase ($> 90\%$). Overall accuracy may slightly improve on zero-shot arithmetic tasks but degrades in tasks requiring broader factual context.
- *Interpretation*: Limited benefit from self-consistency; gains in simple numeric reasoning offset by instability in open-ended contexts.

Table 2 summarizes relative changes where quantitatively available.

5.4. Domain-Specific Analysis

Numeracy_600K_article_title *&* aclsent

- GLM-4V-Flash: modest accuracy gains (e.g., from $\approx$ 0.615 to $\approx$ 0.73) with very low generation errors.
- Qwen3-8B and Xiaomi-7B: unstable changes; slight improvements in few cases but often inconsistent, leading to negligible net gain.
- DeepSeek-8B: marginal gains on simple numeric titles but suffers on context-dependent prompts.

`DROP` *(Complex Reasoning)*

- GLM-4V-Flash: accuracy climbs significantly, generation errors remain near zero.
- Qwen3-8B: large reduction in generation errors under majority voting, boosting accuracy by a large relative margin ($\approx$ 85% drop in error), indicating self-consistency helps chain-of-thought arithmetic.
- Xiaomi-7B and DeepSeek-8B: generation errors remain high; ensemble brings limited or negative benefit.

`qa-text-source-comparison` *&* `FinNum`

- GLM-4V-Flash: moderate improvement under majority voting.
- Qwen3-8B: error rate increases, suggesting inconsistent factual recall.
- Xiaomi-7B: slight improvement on qa-text but overall unstable.
- DeepSeek-8B: mixed results.

5.5. Error-Type Breakdown

We categorize errors as in Section 4.3 as: number-relationship errors, undetectable errors, type errors, anomalies, and improper data/factual errors. Under majority voting:

- **GLM-4V-Flash**: TN improves across all error types, reducing false positives; TP also increases, lowering false negatives. Anomaly and relationship-error categories see the largest TN gains.
- **Qwen3-8B / Xiaomi-7B / DeepSeek-8B**: In number-relationship categories, Qwen3-8B TP may increase but TN remains low; ensemble reduces some false negatives but increases false positives elsewhere. For subtle factual errors, majority runs diverge, raising FP. Overall, only GLM-4V-Flash benefits consistently.

5.6. Prompt-Type Effects

We compare few-shot vs. zero-shot prompting under majority voting:

- **Few-shot**:
  - Qwen3-8B and DeepSeek-8B perform worse: increased prompt complexity leads to divergent reasoning paths and higher generation-error rates.
  - Xiaomi-7B also degrades significantly.
  - GLM-4V-Flash maintains low generation errors but sees smaller relative gains compared to zero-shot, likely because few-shot already guides a stable path.
- **Zero-shot**:
  - GLM-4V-Flash: largest relative improvement (+8.4% accuracy).
  - Qwen3-8B: some benefit in arithmetic-heavy cases (`DROP`) but net effect mixed.
  - Xiaomi-7B and DeepSeek-8B: no reliable gains.
- *Interpretation*: Majority voting best complements zero-shot chain-of-thought for robust models. For smaller or less stable models, added examples increase variance, harming consensus.

5.7. Discussion

- **Model Size and Pretraining Differences**: GLM-4V-Flash's strong gains may reflect its larger underlying model (via Zhipu API) and multimodal capabilities; direct comparison to smaller models may not be fully fair.
- **Prompt Uniformity**: Pretraining biases may explain baseline strengths. Identical prompt formats are critical but may not overcome inherent biases.
- **Sampling Overhead**: Majority voting requires N inference calls, increasing latency and compute; trade off robustness gains (notably large for GLM-4V-Flash) vs. cost.
- **Generation Error Trade-offs**: For weaker models, majority sampling often increases generation errors, undermining accuracy. Ensemble consensus benefits when base model's output distribution is already stable.
- **Analysis in Percentage Changes**: Reporting relative percentage changes (e.g., "GLM accuracy +8.4%", "Qwen DROP error −85.4%") aids interpretation.
- **Hardware and Throughput**: Faster inference infrastructure could allow more sampling runs per query, improving consensus quality; future work could explore dynamic stopping once majority emerges.
- **Future Directions**:
  - Apply majority voting on larger/robust LLMs (e.g., LLaMA-3.8B, GPT-4) to quantify scaling effects.
  - Investigate hybrid ensembles: combine outputs from different models rather than repeated runs of one.
  - Explore confidence estimation: weight runs by internal confidence scores rather than uniform voting.
  - Evaluate on additional real-world numeric-text datasets beyond BeNEDect.
  - Integrate symbolic modules with majority voting to further reduce error rates.

Model	Metric / Subset	Baseline	With Majority Voting
GLM-4V-Flash	Overall accuracy (zero-shot)	0.594	≈0.678 (+8.4%)
Qwen3-8B	DROP error rate	High	≈3% (–85.4%)
Xiaomi-7B	Overall accuracy	0.472	<0.472 (decrease)
DeepSeek-8B	DROP accuracy	0.280	modest increase

Table 2
Relative impact of majority voting on selected metrics/subsets.

5.8. Summary

Overall, majority-voting ensemble yields substantial robustness improvements for GLM-4V-Flash, especially in zero-shot settings and on complex reasoning subsets like DROP, reducing false positives and false negatives with near-zero generation errors. For smaller or less stable models (Qwen3-8B, Xiaomi-7B, DeepSeek-8B), the approach produces mixed or negative effects: while it can dramatically reduce errors in structured arithmetic cases (e.g., Qwen3-8B on DROP), it often increases variance and generation failures in open-ended or factual contexts. These findings underscore that simple consensus sampling is an effective "plug-in" enhancement for sufficiently stable LLMs but must be applied with caution and cost–benefit analysis for less robust models.

6. Conclusion and Future Work

We have presented majority voting as a lightweight ensemble strategy to improve numerical error detection with large language models. By issuing each prompt multiple times—varying only random seeds or sampling parameters—and selecting the most frequent "Yes"/"No" verdict, our experiments on synthetic benchmarks and the BeNEDect corpus demonstrate substantial robustness gains for inherently stable models. In particular, a model such as GLM-4V-Flash achieves on the order of 8.4% relative accuracy improvement in zero-shot settings, along with significant reductions in both false positives and false negatives and negligible generation errors. Smaller or less consistent models (e.g., Qwen3-8B, Xiaomi-7B, DeepSeek-8B) exhibit mixed outcomes: while arithmetic-focused subsets may benefit, increased output variability or generation errors in open-ended or factual contexts often offset potential gains. Moreover, the approach incurs a cost of N inference calls per input (e.g., $N = 10$), so practical deployment requires a careful trade-off analysis between compute/latency overhead and robustness improvements. Overall, majority voting is immediately deployable with zero training overhead and can significantly enhance numeric fidelity when the base model's output distribution is sufficiently consistent; however, its efficacy depends on model stability and task complexity.

Future research will explore combining outputs from multiple LLMs to leverage complementary strengths, using confidence-weighted or adaptive sampling to reduce inference overhead, integrating lightweight symbolic or tool-based checks for cross-validation, evaluating the approach on larger and multimodal models and under distribution shifts to assess generality, and conducting real-world deployment studies with user feedback and cost–benefit analysis to guide practical adoption.

CRediT authorship contribution statement

Hengfeng Hu: Paper Writing, Overall Framework Design. **Meichen Fu:** Paper Reproduction and Optimization, Baseline. **Weijie Lin:** Models Improvement, Experiments.

References

Abedjan, Z., Chu, X., Deng, D., Castro Fern'andez, R., Ilyas, I.F., Ouzzani, M., Papotti, P., Stonebraker, M., Tang, N., 2016. Detecting data errors: Where are we and what needs to be done? Proceedings of the VLDB Endowment 9, 993–1004.

Barchard, K.A., Pace, L.A., 2011. Preventing human error: The impact of data entry methods on data accuracy and statistical results. Computers in Human Behavior 27, 1834–1839. doi:10.1016/j.chb.2011.04.004.

Cairns, P., Thimbleby, H., 2010. Reducing number entry errors: Solving a widespread, serious problem. Journal of the Royal Society Interface 7, 1429–1439.

Cao, H., Leng, H., Zhao, J., Zhao, Y., Zhao, C., Li, B., 2024. A deep-learning-based error-correction method for atmospheric motion vectors. Remote Sensing 16. URL: https://www.mdpi.com/2072-4292/16/9/1562, doi:10.3390/rs16091562.

Cauchi, A., Gimblett, A., Curzon, P., Masci, P., Thimbleby, H., 2012. Safer "5-key" number entry user interfaces using differential formal analysis, in: Proc. BCS Conference on Human-Computer Interaction (BCS-HCI), pp. 29–38.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al., 2021. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168 9.

Dietterich, T.G., 2000. Ensemble methods in machine learning, in: Proceedings of the First International Workshop on Multiple Classifier Systems, Springer. pp. 1–15. URL: https://link.springer.com/chapter/10.1007/3-540-45014-9_1, doi:10.1007/3-540-45014-9_1.

Fan, G., Fan, W., Geerts, F., 2014. Detecting errors in numeric attributes, in: Web-Age Information Management (WAIM), Springer. pp. 125–137.

Forootani, A., 2025. A survey on mathematical reasoning and optimization with large language models. arXiv preprint arXiv:2503.17726 .

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., Steinhardt, J., 2021. Measuring mathematical problem solving with the math dataset. arXiv preprint arXiv:2103.03874 .

Kuncheva, L.I., 2004. Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience. URL: https://www.wiley.com/en-us/Combining+Pattern+Classifiers%3A+Methods+and+Algorithms-p-9780471480837.

Sakamoto, T., Sugawara, S., Aizawa, A., 2025. Development of numerical error detection tasks to analyze the numerical capabilities of language models, in: Proceedings of the 31st International Conference on Computational Linguistics, pp. 9957–9976.

Vadlapati, P., 2023. Investigating the impact of linguistic errors of prompts on llm accuracy. ESP Journal of Engineering & Technology Advancements 3, 150–153. doi:10.56472/25832646/JETA-V3I6P111.

Wallace, E., Wang, Y., Li, S., Singh, S., Gardner, M., 2019. Do nlp models know numbers? probing numeracy in embeddings, in: Proceedings of EMNLP-IJCNLP, pp. 5307–5315.

Wang, X., Wei, J., Schuurmans, D., Le, Q., Chi, E., Narang, S., Chowdhery, A., Zhou, D., 2022. Self-consistency improves chain-of-thought reasoning in language models. arXiv preprint arXiv:2203.11171 .

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., Zhou, D., 2022. Chain-of-thought prompting elicits reasoning in large language models. arXiv preprint arXiv:2201.11903 .

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q., et al., 2022. Least-to-most prompting enables complex reasoning in large language models. arXiv preprint arXiv:2205.10625 .

Zhou, S., Gao, C.Y., Duan, Z., Xi, X., Li, Y., 2023. A robust error correction method for numerical weather prediction wind speed based on vmd, pca and random forest (vmd-pca-rf). Geoscientific Model Development 16, 6247–6266. doi:10.5194/gmd-16-6247-2023.

Zhu, F., Dai, D., Sui, Z., 2024. Language models know the value of numbers. arXiv preprint arXiv:2401.03735 .

Zhu, F., Dai, D., Sui, Z., 2025. Language models encode the value of numbers linearly, in: Proceedings of the 31st International Conference on Computational Linguistics, pp. 693–709.