# NASH: Numerically Aware Scoring Heuristic for Robust Semantic Similarity

**Yu-Shiang Huang**[1,3*] **Yun-Yu Lee**[2*] **Tzu-Hsin Chou**[1] **Che Lin**[1] **Chuan-Ju Wang**[3]
[1]National Taiwan University
[2]National Yang Ming Chiao Tung University
[3]Academia Sinica
F09946004@ntu.edu.tw, raelee.cs12@nycu.edu.tw
dorachou0609@gmail.com, chelin@ntu.edu.tw, cjwang@citi.sinica.edu.tw

## Abstract

Numerical precision is critical in financial NLP, yet embedding-based semantic similarity metrics exhibit numerical blindness—failing to distinguish contradictory values within similar contexts. We introduce NASH (Numerically Aware Scoring Heuristic), a model-agnostic metric that decouples numerical verification from textual semantic evaluation through a three-stage pipeline: (1) modal separation via numeric masking, (2) dual-channel similarity estimation through masked-text similarity and context-aware numeric alignment, and (3) IDF-weighted aggregation. NASH functions as a drop-in enhancement to existing embedding-based metrics. Validated on our proposed NumFinE financial numerical evaluation benchmark and established semantic similarity datasets (STS-B, Financial-STS), NASH achieves substantial improvements in numerical sensitivity (up to +159.6% on listwise ranking) while preserving general semantic performance, establishing a reliable standard for numeracy-aware evaluation.

[1]

## 1 Introduction

Reliable evaluation is the cornerstone of trustworthy natural language generation (NLG). The rapid adoption of large language models (LLMs) in high-stakes domains—from financial reporting (Wu et al., 2023; Yang et al., 2023) and medical summarization (Singhal et al., 2023; Luo et al., 2022) to analytical reasoning (Wei et al., 2022; Kojima et al., 2022)—intensifies the need for rigorous metrics. The research community has largely transitioned from surface-level $n$-gram matching (BLEU (Papineni et al., 2002), ROUGE (Lin, 2004)) toward embedding-based approaches, most notably BERTScore (Zhang et al., 2020), BLEURT (Sellam et al., 2020), and MoverScore (Zhao et al., 2019). While these embedding-based approaches better capture semantic similarity, they exhibit a critical gap in numerical reasoning.

These semantic metrics suffer from a critical limitation: **numerical blindness**. Driven by the distributional hypothesis, language models map distinct numerical tokens (e.g., "1.0%" vs. "10%") to nearly identical vectors due to shared syntactic contexts. Consequently, factually contradictory statements receive near-perfect scores. For example, BERTScore assigns a similarity of **0.9893** to "The company reported earnings of $1 million" and "The company reported earnings of $10 million," failing to distinguish the ten-fold difference in reported value.

This systemic failure is obscured by evaluation practices. Current benchmarks like STS-B (Cer et al., 2017), the de facto standard for evaluating semantic similarity, exemplify this issue. STS-B employs subjective human annotations on a 5-point Likert scale with minimal guidance on numerical precision. Its correlation-based evaluation captures holistic semantic resemblance rather than axiomatic numerical accuracy, allowing metrics to achieve high correlations while failing catastrophically on numerical reasoning.

This limitation is unacceptable in high-stakes domains where numerical precision directly impacts decision-making. In finance, the distinction between 4% and 5% GDP growth represents a fundamental shift in economic outlook with measurable consequences for investment strategy and risk assessment. We focus on financial text as our primary evaluation domain due to its quantitative density and reliance on precise numerical values. In this context, even single-digit deviations can signal opposite market implications, making financial NLP an ideal testbed for evaluating the numerical sensitivity of semantic metrics.

To bridge this gap, we exploit a fundamental

*These authors contributed equally.
[1]The evaluation code and benchmark datasets are available at https://github.com/cnclabs/codes.fin.bertscore.



*Findings of the Association for Computational Linguistics: ACL 2026*, pages 22303–22317

property of numerical tokens: unlike subjective vocabulary, numbers possess objective magnitudes that offer unambiguous ground truth for semantic verification. We introduce **NASH** (**N**umerically **A**ware **S**coring **H**euristic), a model-agnostic metric designed to overcome numerical blindness in embedding-based similarity assessment. NASH functions as a drop-in enhancement to existing metrics through a hybrid strategy: it decomposes evaluation into separate contextual and numerical components, computes magnitude-aware similarity for numerical tokens, and integrates these signals via IDF-weighted fusion. This approach preserves semantic quality on standard benchmarks while enforcing precise numerical alignment in quantitative contexts. We validate NASH through systematic evaluation on both our proposed **NumFinE** (**Num**erical **Fin**ancial **E**valuation), a proposed diagnostic benchmark, and existing general-domain datasets. Our contributions are:

1. **NASH**, a model-agnostic enhancement that addresses numerical blindness through magnitude-aware alignment.
2. **NumFinE**, a diagnostic benchmark for numerical blindness evaluation.
3. Validation showing NASH outperforms baselines on numerical reasoning without degrading general performance.

## 2 Problem Analysis

### 2.1 Background: Embedding-based Metrics

State-of-the-art evaluation metrics rely on pretrained Transformers $f_\theta$ to map sentences into continuous vector spaces. Given a reference sentence $s$ and a candidate $\hat{s}$, the model produces sequences of contextualized embeddings: $H = \{\mathbf{h}_i\}_{i=1}^m = f_\theta(s)$ and $\hat{H} = \{\hat{\mathbf{h}}_j\}_{j=1}^n = f_\theta(\hat{s})$. Existing approaches leverage these representations through two main paradigms:

**Token-level matching.** BERTScore (Zhang et al., 2020) computes similarity via greedy alignment of token embeddings in $H$ and $\hat{H}$. It calculates recall ($R_{\text{token}}$) and precision ($P_{\text{token}}$) by matching each token to its most similar counterpart:

$$R_{\text{token}}(s,\hat{s}) = \frac{\sum_{i=1}^m \delta_i \cdot \max_j \cos(\mathbf{h}_i, \hat{\mathbf{h}}_j)}{\sum_{i=1}^m \delta_i},$$

$$P_{\text{token}}(s,\hat{s}) = \frac{\sum_{j=1}^n \hat{\delta}_j \cdot \max_i \cos(\hat{\mathbf{h}}_j, \mathbf{h}_i)}{\sum_{j=1}^n \hat{\delta}_j},$$

where $\delta_i$ and $\hat{\delta}_j$ denote importance weights for reference and candidate tokens, respectively (e.g., inverse document frequency (IDF)), and $S_{\text{token}}(s,\hat{s})$ is their harmonic mean.

**Sentence-level pooling.** Bi-encoder models (e.g., SBERT (Reimers and Gurevych, 2019)) aggregate the token representations into fixed-size vectors via pooling: $S_{\text{sent}}(s,\hat{s}) = \cos(\text{Pool}(H), \text{Pool}(\hat{H}))$. While efficient, this compression often averages out fine-grained numerical details.

### 2.2 Numerical Blindness

While robust for general semantics, embedding-based metrics fail in numeracy-sensitive contexts. Consider three sentences:

- $s_1$: *"Revenue increased by **3.56%**."*
- $s_2$: *"Revenue increased by **4%**."*
- $s_3$: *"Revenue increased by **40%**."*

Intuitively, $s_2$ is closer to $s_1$ than to $s_3$ due to magnitude proximity. However, BERTScore[2] produces the opposite ranking: $\text{sim}(s_1, s_2) = 0.9639 < \text{sim}(s_2, s_3) = 0.9764$.

### 2.3 Root Causes

We identify two structural mechanisms underlying this failure:

**Tokenization fragmentation.** Subword tokenizers fragment numbers into disjoint tokens, destroying magnitude information. The value "3.56" is split into three tokens while "40" remains atomic:

$$s_1 : [\ldots, \texttt{3}, \texttt{.}, \texttt{56}, \texttt{\%}, \texttt{.}]$$
$$s_2 : [\ldots, \texttt{4}, \texttt{\%}, \texttt{.}]$$
$$s_3 : [\ldots, \texttt{40}, \texttt{\%}, \texttt{.}]$$

This fragmentation prevents direct magnitude comparison: embeddings for multi-token numbers (e.g., "3.56" → ["3", ".", "56"]) cannot be systematically compared to single-token numbers (e.g., "40"), as token-level representations encode lexical patterns rather than numerical semantics.

**Contextual anisotropy.** Even with aligned tokens, numerical embeddings are distorted by context. We trace the layer-wise cosine similarity of the numeral "4" in $s_2$ against: (1) small deviation ($s_1$: 3.56%), (2) large deviation ($s_3$: 40%), and (3) identical value in different context ($s_4$: *"The tax*

[2]Using the implementation from https://huggingface.co/spaces/evaluate-metric/bertscore with default settings.

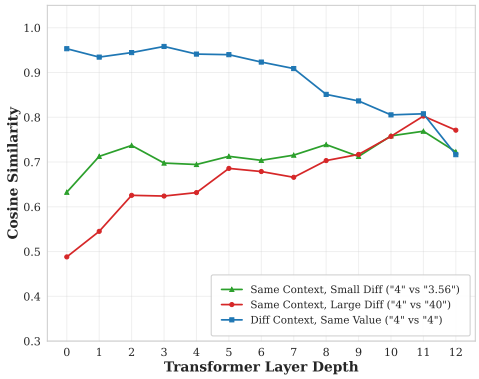


Figure 1: **Layer-wise cosine similarity between the anchor numeral "4" (from $s_2$) and numerals in target sentences.**

*rate is 4 percent this year.*"). Multi-token numbers use mean-pooling.

Figure 1 reveals a fundamental decoupling between numerical value and representation. In identical contexts ($s_1, s_3$), embeddings converge rapidly regardless of magnitude. Paradoxically, in deeper layers, the contradictory "40" becomes more similar to the anchor than the numerically closer "3.56", indicating that the model prioritizes syntactic context over numerical content. Conversely, the identical token "4" in a shifted context ($s_4$) drifts apart, eventually ranking lower than numerically distinct values. BERT representations are dominated by contextual semantics, overriding the objective identity of numbers.

This pattern demonstrates that Transformer representations are structurally unsuited for numerical evaluation. To address this, numerical similarity must be computed directly from raw values before contextual encoding distorts their magnitudes.

## 3 Methodology

To address the limitations identified in Section 2, we propose **NASH**, a model-agnostic enhancement that decouples numerical verification from textual semantics. As illustrated in Figure 2, NASH operates through a three-stage pipeline: (1) modal separation via numeric masking, (2) dual-channel similarity estimation through masked-text similarity and context-aware numeric alignment, and (3) IDF-based hybrid aggregation.

### 3.1 Modal Separation via Numeric Masking

Given a sentence pair $(s, \hat{s})$, NASH separates textual and numerical modalities to address the contextual anisotropy identified in Section 2.3, where embeddings prioritize linguistic context over numerical content. We employ a rule-based extractor to identify diverse numeric formats, including integers, decimals, leading-dot decimals (e.g., `.26`), and values with financial suffixes (e.g., `%`, `bp`, `k/m/b`). Each detected numeric mention is replaced with a special token `[NUM]`, transforming, for instance, "*Revenue grows 15%*" into "*Revenue grows* `[NUM]%`". We denote the resulting masked sentence pair as $(s^*, \hat{s}^*)$, representing the textual modality isolated from numeric magnitudes.

### 3.2 Dual-Channel Similarity Estimation

NASH estimates semantic similarity through two independent channels: textual similarity and context-aware numerical similarity. Both channels operate on the same input pair $(s, \hat{s})$ but capture complementary semantic aspects.

#### 3.2.1 Textual Similarity

To assess linguistic similarity independent of numerical values, we compute similarity on the masked sentence pair $(s^*, \hat{s}^*)$. We use the embedding-based approaches introduced in Section 2.1 as backends. The textual similarity score $S_{\text{text}}$ is computed by applying the respective metric to the masked inputs:

$$S_{\text{text}}(s, \hat{s}) = \mathcal{S}_{\text{base}}(s^*, \hat{s}^*), \tag{1}$$

where $\mathcal{S}_{\text{base}}$ denotes any backend similarity metric and the range of $S_{\text{text}}$ is $[-1, 1]$.[3]

#### 3.2.2 Context-Aware Numeric Similarity

To explicitly model quantitative relations, we compute numerical similarity on the original sentence pair $(s, \hat{s})$. Our design is motivated by two key insights: (1) numbers with similar contextual embeddings likely serve similar semantic roles (e.g., both represent growth rates), and (2) once semantic roles are aligned, numerical deviation reflects meaningful differences in magnitude. We therefore employ a two-stage process: first, we use contextual embeddings to align numbers serving comparable semantic functions; second, we apply a precision-recall framework to compute magnitude differences between aligned pairs while penalizing missing alignments.

**Contextual alignment.** Let $V_s$ and $V_{\hat{s}}$ denote the sets of numeric mentions extracted from $s$ and $\hat{s}$. For each mention $v \in V_s$, we obtain its vector

[3]While theoretically unbounded in $[-1, 1]$, empirical scores for semantically plausible pairs seldom fall below zero.

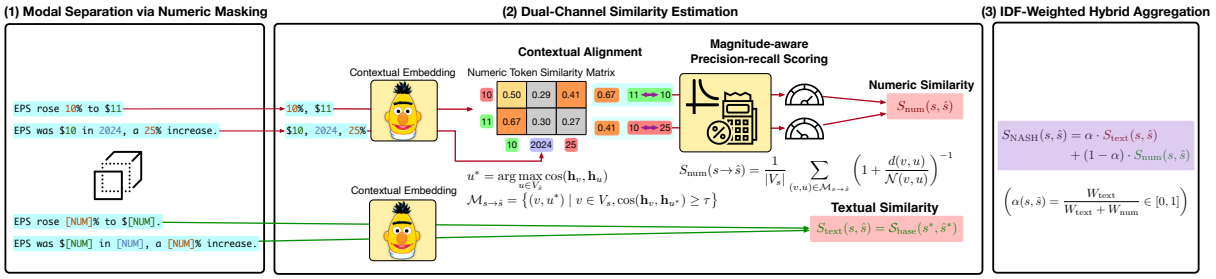


Figure 2: **Overview of the NASH pipeline.** Given a sentence pair, NASH separates textual and numerical semantics through a three-stage process. First, numeric tokens are masked to isolate the textual context, enabling pure semantic similarity computation. Second, numbers are independently aligned using context-aware matching with threshold gating, followed by magnitude-based similarity computation. Third, textual similarity (from masked sentences) and numerical similarity are combined through IDF-weighted aggregation to produce the final score.

representation $\mathbf{h}_v$ by averaging the corresponding token embeddings. To ensure that comparisons are restricted to semantically compatible contexts (e.g., preventing alignment of year "2025" with monetary value "2,025 USD"), we leverage contextual semantics from token embeddings. We align $v$ to its closest compatible counterpart $u^* \in V_{\hat{s}}$ by maximizing cosine similarity:

$$u^* = \arg\max_{u \in V_{\hat{s}}} \cos(\mathbf{h}_v, \mathbf{h}_u).$$

A match $(v, u^*)$ is valid only if similarity exceeds the threshold $\tau$. We define the directional match set for $s$:

$$\mathcal{M}_{s \to \hat{s}} = \{(v, u^*) \mid v \in V_s, \cos(\mathbf{h}_v, \mathbf{h}_{u^*}) \geq \tau\}.$$

Analogously, we define $\mathcal{M}_{\hat{s} \to s}$ for matches from $\hat{s}$ to $s$.

**Magnitude-aware precision-recall scoring.** To compute sentence-level numerical similarity, we aggregate pairwise scores bidirectionally using a precision-recall framework. For each directional match set $\mathcal{M}_{s \to \hat{s}}$, the score is defined as:

$$S_{\text{num}}(s \to \hat{s}) = \frac{1}{|V_s|} \sum_{(v,u) \in \mathcal{M}_{s \to \hat{s}}} \left(1 + \frac{d(v,u)}{\mathcal{N}(v,u)}\right)^{-1},$$

where $d(v,u) = |v - u|$ is the absolute magnitude difference and $\mathcal{N}(u,v)$ is a scale-adaptive normalization factor that adjusts to the magnitude range of the aligned pair.[4] By utilizing $|V_s|$ as the denominator, mentions that fail to align (i.e., $v \in V_s$ but $v \notin \mathcal{M}_{s \to \hat{s}}$) implicitly contribute zero, penalizing low recall. The reverse score $S_{\text{num}}(\hat{s} \to s)$ is computed analogously. The final numerical similarity is their arithmetic mean:[5]

$$S_{\text{num}}(s, \hat{s}) = \frac{1}{2}\Big(S_{\text{num}}(s \to \hat{s}) + S_{\text{num}}(\hat{s} \to s)\Big). \tag{2}$$

### 3.3 IDF-Weighted Hybrid Aggregation

To dynamically balance the textual and numerical channels, we compute an IDF-weighted coefficient $\alpha$. We accumulate the IDF scores of tokens in the mask pair $(s^*, \hat{s}^*)$ into textual mass $W_{\text{text}}$ and numeric mass $W_{\text{num}}$ (treating `[NUM]` as tokens):[6]

$$\alpha(s, \hat{s}) = \frac{W_{\text{text}}}{W_{\text{text}} + W_{\text{num}}} \in [0, 1].$$

This mechanism grants higher importance to the numeric channel (lower $\alpha$) when numerical information density increases. We integrate textual similarity (Eq. 1) and numerical similarity (Eq. 2) through a convex combination:

$$S_{\text{NASH}}(s, \hat{s}) = \alpha S_{\text{text}}(s, \hat{s}) + (1 - \alpha) S_{\text{num}}(s, \hat{s}). \tag{3}$$

This formulation ensures robust performance: when numeric information is absent ($W_{\text{num}} = 0 \to \alpha = 1$), NASH reduces to the textual baseline; as numerical density increases, the score enforces quantitative consistency.

## 4 NumFinE Construction

To systematically investigate numerical blindness, we construct NumFinE, a diagnostic benchmark of

[4] We use $\mathcal{N}(u,v) = 1 + \frac{1}{2}(|v| + |u|)$ in this paper.

[5] Edge cases: if $V_s = V_{\hat{s}} = \emptyset$, we set $S_{\text{num}} = 1$; if numeric mentions exist in only one sentence, $S_{\text{num}} = 0$.

[6] IDF statistics are computed over a corpus of masked sentences, treating each masked sentence as a document. Both word tokens and the special token `[NUM]` are included in IDF computation.

9,227 authentic financial sentences with controlled numerical perturbations. We extract sentences from diverse financial corpora and systematically modify numerical values while preserving syntactic structure, generating counterfactual pairs differing solely in magnitude. This generates counterfactual pairs differing solely in numerical magnitude, ensuring that metric divergence reflects numerical sensitivity rather than lexical or syntactic variation.

### 4.1 Data Sources

NumFinE draws from three complementary financial text sources to ensure broad domain coverage. Social media is represented by the FinNum1 dataset (Chen et al., 2018). Financial news is sourced from Financial Phrasebank (Malo et al., 2014), retaining only sentences with at least 50% annotator agreement. Regulatory disclosures are sampled from 10-K filings (fiscal years 2023–2024) of 55 publicly traded companies: the five largest by market capitalization in each of the eleven GICS (Global Industry Classification Standard) sectors. This stratified sampling ensures representation across industries, including technology, healthcare, finance, and energy. Together, these sources yield 9,227 sentences containing diverse numerical expressions—percentages, dollar amounts, dates, and financial metrics—serving as the foundation for controlled augmentation.

### 4.2 Numerical Augmentation

#### 4.2.1 Numerical Extraction and Categorization

We identify every numeral as a potential augmentation target and categorize it according to FinNum1's taxonomy (Chen et al., 2018), which distinguishes seven types: temporal expressions, monetary values, percentages, quantities, product numbers, indicators, and option values. For Financial Phrasebank and 10-K sentences, categories are automatically assigned using GPT-4o [7]. Each sentence containing at least one numeral becomes a *base sentence* with target number $t$, where $\nu(t)$ denotes the numerical value (e.g., $\nu(t) = 15$ for "15%" or "15M").

#### 4.2.2 Augmentation Strategy

To systematically test numerical sensitivity, we apply random augmentation approaches to modify target numbers while preserving grammatical structure. Since base sentences may contain multiple target numbers, each modifiable through different methods, we define an *evaluation unit* as a tuple: $(s, t, \alpha)$, where $s$ is a base sentence, $t$ is a target number in $s$, and $\alpha$ specifies the augmentation types. Each evaluation unit produces $k$ augmented variants: $(s, t, \alpha) \mapsto \{\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_k\}$, where $\tilde{s}_i$ results from applying augmentation $\alpha$ to $t$ in $s$.

The augmentation perturbs each target number within category-specific bounds to test metric sensitivity to magnitude variation. For each base sentence with a target number, we generate $k = 9$ variants by randomly increasing or decreasing $\nu(t)$ (e.g., shifting percentages by ±10%, ±25%, ±50%, ±100%). This produces a spectrum of numerical deviations from the original value. Detailed augmentation ranges for each category appear in Appendix G.

### 4.3 Quality Assurance

To ensure augmented sentences remain grammatically valid and semantically plausible, we employ GPT-4o-mini as an automatic validator.[8] The model evaluates each variant along two dimensions: (1) **temporal validity** (whether dates and times follow calendar conventions, e.g., no "February 31st") and (2) **numerical plausibility** (whether shifted values remain contextually reasonable, e.g., not "a 500% profit margin"). GPT-4o-mini assigns each variant a validity score in $[0.0, 1.0]$. We discard evaluation units where more than 3 variants out of 9 receive scores below 0.5, indicating systematic augmentation issues. This filtering retains 75.66% of augmentation units, resulting in 14,651 valid units.[9] Detailed statistics appear in Appendix D.

Having constructed a validated set of 14,651 evaluation units spanning seven numerical categories, we now describe how these units are used to assess metric performance.

### 4.4 Evaluation Protocols

To assess metric sensitivity to numerical changes, we employ two evaluation protocols: **anchor-based evaluation**, which compares augmented variants against their base sentence, and **Cross-Pair evaluation**, which tests ranking consistency

[7] Prompts provided in Appendix F.

[8] Prompts provided in Appendix F.

[9] To validate this automated process, we manually inspected 50 randomly sampled evaluation units. Human judges agreed with GPT-4o-mini's accept/reject decisions in 95% of cases (48/50 evaluated units), confirming the reliability of automated filtering. Detailed annotation instructions are provided in Appendix H.

| Protocol | # Units | # Sentences |
|---|---|---|
| ***Anchor-based Evaluation*** | | |
| Triplet (each level) | 14,651 | 43,953 |
| Listwise ranking | 14,651 | 146,510 |
| ***Cross-Pair Evaluation*** | | |
| Cross-Pair | 7,320 | 29,280 |

Table 1: **Statistics of the NumFinE datasets.**

across different sentence contexts. Table 1 summarizes benchmark statistics.

#### 4.4.1 Anchor-based Evaluation

Anchor-based evaluation tests whether metrics assign higher similarity to numerically closer variants. Given a base sentence $s$ with target number $t$ and its augmented variants $\{\tilde{s}_i\}_{i=1}^{k}$ with modified values $\{\tilde{t}_i\}_{i=1}^{k}$, we define numerical distance as $d_\nu(s, \tilde{s}_i) = |\nu(t) - \nu(\tilde{t}_i)|$.

**Triplet evaluation.** We evaluate whether a metric correctly ranks the numerically closest variant higher than a more distant one. Let $\{\tilde{s}_{(j)}\}_{j=1}^{k}$ denote the variants sorted by ascending numerical distance $d_\nu(s, \tilde{s}_i)$. The task is to distinguish the positive sample $\tilde{s}^+ = \tilde{s}_{(1)}$ (closest) from a negative sample $\tilde{s}^-$ selected based on difficulty:

- **Easy:** $\tilde{s}^- = \tilde{s}_{(k)}$ (farthest variant).
- **Medium:** $\tilde{s}^- = \tilde{s}_{(\lfloor k/2 \rfloor)}$ (median variant).
- **Hard:** $\tilde{s}^- = \tilde{s}_{(2)}$, (second closest variant).

We report accuracy as the proportion of Triplets where $\text{sim}(s, \tilde{s}^+) > \text{sim}(s, \tilde{s}^-)$.

**Listwise evaluation.** We rank all $k$ variants by their similarity to $s$ and compare this predicted ranking against the gold ranking (variants sorted by ascending numerical distance $d_\nu$). Larger numerical deviations should yield lower similarity scores. This measures the ability of similarity metrics to perform fine-grained numerical comparisons across multiple sentences. We measure ranking quality using Kendall's $\tau_b$ correlation (Kendall, 1938), where $\tau_b = 1$ indicates perfect agreement, $\tau_b = 0$ indicates no correlation, and $\tau_b = -1$ indicates complete reversal.

#### 4.4.2 Cross-Pair Evaluation

While anchor-based evaluation tests metrics within a single sentence context, Cross-Pair evaluation tests whether numerical distance relationships generalize across different sentences. This is more challenging because metrics must maintain consistent numerical sensitivity regardless of the surrounding text.

We randomly sample two base sentences $s$ and $s'$ from the same numerical category, then independently sample one augmented variant from each to form pairs $(s, \tilde{s}_i)$ and $(s', \tilde{s}'_j)$. The gold ordering is determined by numerical distance: if $d_\nu(s, \tilde{s}_i) < d_\nu(s', \tilde{s}'_j)$, then a metric should satisfy $\text{sim}(s, \tilde{s}_i) > \text{sim}(s', \tilde{s}'_j)$: the pair with smaller numerical deviation should receive higher similarity. This tests whether metrics preserve correct similarity orderings across different sentence contexts. We report accuracy, which is the proportion of correctly ordered pairs.

## 5 Experiments

### 5.1 Baselines

We evaluate NASH across diverse embedding models spanning different architectures, scales, and training objectives:[10]

- **Token-level encoders:** `bert-base-uncased`, `FinBERT` (domain-specific financial model), and `DeBERTa-xlarge-mnli` (large-scale encoder).
- **Sentence-level bi-encoders:** `all-MiniLM-L6-v2`, `bge-m3`, `unsup-simcse-bert-base-uncased`, `Qwen3-Embedding-0.6B`, and `text-embedding-3-small` (OpenAI).[11]

NASH functions as a plug-and-play enhancement applicable to any embedding-based metric. We compare baseline performance against NASH-enhanced scores to demonstrate consistent improvements independent of model architecture.

### 5.2 Experimental Setup

#### 5.2.1 Threshold Selection

The alignment threshold $\tau$ is determined in a data-driven manner. We randomly pair base sentences and compute token-level cosine similarity using `bert-base-uncased` to estimate a background similarity distribution. Let $\mu$ and $\sigma$ denote the mean and standard deviation of these scores; the threshold is set as $\tau = \mu + \sigma$. Performance is stable across threshold values; Appendix B demonstrates robustness to $\tau$ selection.

#### 5.2.2 Similarity Computation

Textual similarity $S_{\text{text}}$ uses BERTScore F1 for token-level encoders and cosine similarity for sentence-level bi-encoders (see Section 2.1).

[10] Model checkpoints are available in Appendix C.

[11] Proprietary APIs do not provide token-level embeddings required for NASH's numerical channel; we report `text-embedding-3-small` as baseline-only.

| Model | Triplet (Easy) | | Triplet (Med) | | Triplet (Hard) | | Listwise | |
|---|---|---|---|---|---|---|---|---|
| | Base | +NASH | Base | +NASH | Base | +NASH | Base | +NASH |
| *Token-level Encoders* | | | | | | | | |
| `bert-base-uncased` | **0.9214** | 0.9857 $_{\uparrow 7.0\%}$ | 0.8359 | 0.9745 $_{\uparrow 16.6\%}$ | **0.7058** | 0.8821 $_{\uparrow 25.0\%}$ | 0.5409 | 0.7834 $_{\uparrow 44.8\%}$ |
| `FinBERT` | 0.9186 | 0.9795 $_{\uparrow 6.6\%}$ | **0.8371** | 0.9602 $_{\uparrow 14.7\%}$ | 0.7017 | 0.8647 $_{\uparrow 23.2\%}$ | 0.5344 | 0.7514 $_{\uparrow 40.6\%}$ |
| `DeBERTa-xlarge-mnli` | 0.8715 | **0.9859** $_{\uparrow 13.1\%}$ | 0.7903 | **0.9774** $_{\uparrow 23.7\%}$ | 0.6879 | **0.8906** $_{\uparrow 29.5\%}$ | **0.6286** | **0.8028** $_{\uparrow 27.7\%}$ |
| *Sentence-level Bi-encoders* | | | | | | | | |
| `all-MiniLM-L6-v2` | 0.8440 | 0.9743 $_{\uparrow 15.4\%}$ | 0.7982 | 0.9577 $_{\uparrow 20.0\%}$ | 0.6782 | 0.8633 $_{\uparrow 27.3\%}$ | 0.3450 | 0.7289 $_{\uparrow 111.3\%}$ |
| `unsup-simcse-bert-base-uncased` | 0.7568 | 0.9301 $_{\uparrow 22.9\%}$ | 0.7676 | 0.9038 $_{\uparrow 17.7\%}$ | 0.7028 | 0.8095 $_{\uparrow 15.2\%}$ | 0.2265 | 0.5879 $_{\uparrow 159.6\%}$ |
| `bge-m3` | 0.8335 | 0.9325 $_{\uparrow 11.9\%}$ | 0.7993 | 0.9093 $_{\uparrow 13.8\%}$ | 0.6809 | 0.8230 $_{\uparrow 20.9\%}$ | 0.3481 | 0.6065 $_{\uparrow 74.2\%}$ |
| `Qwen3-Embedding-0.6B` | 0.8408 | 0.9802 $_{\uparrow 16.6\%}$ | 0.8061 | 0.9627 $_{\uparrow 19.4\%}$ | 0.6791 | 0.8748 $_{\uparrow 28.8\%}$ | 0.3548 | 0.7677 $_{\uparrow 116.4\%}$ |
| `text-embedding-3-small` | 0.8759 | | 0.8247 | | 0.6899 | | 0.4111 | |

Table 2: **Evaluation on NumFinE (Anchor-based).** Results show the original backbone (Base) versus the NASH-enhanced performance (+NASH). Subscripts denote relative improvement percentages. **Bold** indicates best performance in each column.

Numerical similarity $S_{\text{num}}$ uses the proposed precision-recall framework for all models.

### 5.3 Evaluation on NumFinE

We evaluate performance on NumFinE under two anchor-based protocols: Triplet and listwise. Table 2 summarizes results across all baselines. NASH consistently outperforms corresponding baselines under both protocols, demonstrating the effectiveness of its magnitude-aware numerical comparison mechanism.

NASH delivers the most substantial gains in the Listwise setting (up to $+159.6\%$ $\tau_b$ for `unsup-simcse-bert-base-uncased`), highlighting that magnitude-aware alignment is crucial for ranking tasks. Token-level encoders demonstrate strong performance even without NASH enhancement (`bert-base-uncased`: 92.1% Easy baseline; `DeBERTa-xlarge-mnli`: 87.2% Easy baseline), yet NASH consistently elevates all architectures to higher accuracy. Notably, with NASH enhancement, `DeBERTa-xlarge-mnli` achieves the best overall results (98.6% Easy, 97.7% Medium, 89.1% Hard Triplet; $\tau_b = 0.803$ Listwise), while weaker sentence-level models show the most dramatic relative improvements, effectively closing the gap with stronger encoders.

Baseline performance degrades substantially as task difficulty increases (e.g., `bert-base-uncased` drops $23.4\%$ from Easy to Hard). In contrast, NASH maintains high accuracy across all difficulty levels. Crucially, performance gaps widen with difficulty: for `bert-base-uncased`, improvements increase from $+7.0\%$ (Easy) to $+25.0\%$ (Hard). This trend demonstrates that NASH is particularly effective at resolving fine-grained numerical ambiguities where standard metrics struggle most. Due to space constraints, Cross-Pair evaluation is deferred to Appendix A, and additional results for larger language models are provided in Table 7.

### 5.4 Evaluation on STS-B and Financial-STS

To assess NASH's impact on broader semantic tasks, we evaluate two datasets: general-domain **STS-B** (Cer et al., 2017) and domain-specific **Financial-STS** (Liu et al., 2024). For STS-B, we report Spearman's $\rho$ and Pearson's $r$ correlations following standard practice. For Financial-STS (Liu et al., 2024), we report ROC-AUC and PR-AUC following the established protocol. Table 3 summarizes the results.

On Financial-STS, NASH delivers consistent improvements across most models, particularly in PR-AUC, confirming that numerical alignment provides necessary precision in financial contexts where value differences fundamentally alter meaning. Conversely, on STS-B, we observe minor performance variations. This is likely because STS-B relies on 5-point Likert scale annotations that prioritize broad topical similarity over numerical precision (e.g., "I have 2 dogs" and "I have 3 dogs" receive high similarity scores). NASH's enforcement of numerical accuracy, while appropriate for reasoning tasks, may occasionally diverge from these annotation conventions.

### 5.5 Alignment Verification

We verify that the alignment mechanism correctly identifies contextually corresponding numeric mentions by testing on anchor-variant pairs $(s, s^+)$, where $s^+$ is the numerically closest variant to base sentence $s$. We measure directional alignment accuracy (whether the target number in $s$ correctly aligns to the corresponding number in $s^+$) and bidi-

| Model | STS-B (Spearman) | | STS-B (Pearson) | | Fin-STS (ROC) | | Fin-STS (PR) | |
|---|---|---|---|---|---|---|---|---|
| | Base | +NASH | Base | +NASH | Base | +NASH | Base | +NASH |
| *Token-level Encoders* | | | | | | | | |
| `bert-base-uncased` | 0.4567 | 0.5007 $_{\uparrow 9.6\%}$ | 0.4507 | 0.5086 $_{\uparrow 12.9\%}$ | 0.6799 | 0.6849 $_{\uparrow 0.7\%}$ | 0.6496 | 0.6519 $_{\uparrow 0.4\%}$ |
| `FinBERT` | 0.5587 | 0.5720 $_{\uparrow 2.4\%}$ | 0.5330 | 0.5779 $_{\uparrow 8.4\%}$ | 0.6796 | 0.6833 $_{\uparrow 0.5\%}$ | 0.6541 | 0.6554 $_{\uparrow 0.2\%}$ |
| `DeBERTa-xlarge-mnli` | 0.5930 | 0.5123 $_{\downarrow 12.1\%}$ | 0.5356 | 0.5167 $_{\downarrow 3.5\%}$ | 0.6739 | 0.6726 $_{\downarrow 0.2\%}$ | 0.6384 | 0.6350 $_{\downarrow 0.5\%}$ |
| *Sentence-level Bi-encoders* | | | | | | | | |
| `all-MiniLM-L6-v2` | 0.8293 | 0.8235 $_{\downarrow 0.8\%}$ | 0.8383 | 0.8342 $_{\downarrow 0.6\%}$ | 0.6184 | 0.6282 $_{\uparrow 1.6\%}$ | 0.5707 | 0.6036 $_{\uparrow 5.8\%}$ |
| `unsup-simcse-bert-base-uncased` | 0.7815 | 0.7652 $_{\downarrow 2.1\%}$ | 0.7915 | 0.7791 $_{\downarrow 2.0\%}$ | 0.6547 | 0.6794 $_{\uparrow 3.8\%}$ | 0.5958 | 0.6485 $_{\uparrow 8.9\%}$ |
| `bge-m3` | 0.8467 | 0.8457 $_{\downarrow 0.1\%}$ | 0.8404 | 0.8399 $_{\downarrow 0.1\%}$ | 0.6529 | 0.6766 $_{\uparrow 3.6\%}$ | 0.5850 | 0.6408 $_{\uparrow 9.5\%}$ |
| `Qwen3-Embedding-0.6B` | 0.4889 | 0.4904 $_{\uparrow 0.3\%}$ | 0.5049 | 0.4936 $_{\downarrow 2.2\%}$ | 0.6113 | 0.6616 $_{\uparrow 8.2\%}$ | 0.5789 | 0.6074 $_{\uparrow 4.9\%}$ |
| `text-embedding-3-small` | 0.8462 | | 0.8609 | | 0.631 | | 0.5522 | |

Table 3: **Evaluation on STS-B and Financial-STS.** Results show the original backbone (Base) versus the NASH-enhanced performance (+NASH). Subscripts denote relative improvement percentages. **Bold** indicates best performance in each column.

rectional accuracy (whether both directions agree).

Sentence-level encoders (`all-MiniLM-L6-v2`) achieves 80.1% and 83.5% directional accuracy, yielding a bidirectional agreement of 77.3% with 6.5% of mentions filtered by the threshold. Token-level encoders (`bert-base-uncased`) achieve 96.2% accuracy in both directions, resulting in 96.2% bidirectional accuracy with no filtered mentions. These results confirm reliable alignment under both embedding paradigms, with stronger performance for token-level representations. To further illustrate this robust alignment mechanism in complex scenarios, we provide a detailed case study in Appendix E.

### 5.6 Zero-Shot Cross-Domain Generalization

To examine whether NASH is overfitted to financial texts or capable of generalizing to other numeracy-sensitive domains, we further evaluate its performance in the biomedical domain under a fully zero-shot setting. Specifically, we randomly sample base sentences containing percentage expressions (e.g., "15%") from the `context` field of PubMedQA (Jin et al., 2019)[12]. We then apply the triplet and listwise augmentation protocols used in our main experiments and evaluate all models without any domain-specific tuning.

As shown in Table 4, NASH yields substantial and consistent improvements across both sentence-level and token-level encoders. Most notably, for `all-MiniLM-L6-v2`, NASH improves the Listwise ranking performance from 0.1303 to 0.4187, representing a relative gain of +221.3%. These results indicate that NASH is not merely tailored to financial language; rather, it effectively mitigates a more general form of numerical blindness present in diverse contexts, all without requiring any domain adaptation.

[12]Link to the dataset: https://github.com/pubmedqa/pubmedqa/blob/master/data/ori_pqal.json

| Model | Triplet (Easy) | | Listwise | |
|---|---|---|---|---|
| | Base | +NASH | Base | +NASH |
| *Token-level Encoders* | | | | |
| `bert-base-uncased` | 0.7080 | 0.7740 | 0.4483 | 0.5453 |
| `FinBERT` | 0.6680 | 0.7050 | 0.3887 | 0.4185 |
| *Sentence-level Bi-encoders* | | | | |
| `all-MiniLM-L6-v2` | 0.5440 | 0.7060 | 0.1303 | 0.4187 |

Table 4: **Zero-shot evaluation on biomedical data constructed from PubMedQA.** NASH is applied completely out-of-the-box without domain-specific tuning and substantially improves numerical sensitivity in biomedical text.

## 6 Related Work

Automatic evaluation of semantic similarity has evolved from surface-form overlap metrics such as BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004) toward embedding-based approaches that better capture paraphrastic meaning. BERTScore (Zhang et al., 2020), while aligns contextualized token embeddings to compute semantic similarity, has been widely adopted across NLG tasks as both a standalone metric and a strong baseline for learned evaluators like BLEURT (Sellam et al., 2020).

Despite their effectiveness at measuring linguistic similarity, embedding-based metrics are not designed to model numerical meaning explicitly. Sentences with contradictory numeric values (e.g., "profit of 5%" vs. "50%") can receive high similarity scores, a critical limitation in numeracy-dense domains such as finance. In financial NLP, where paraphrasing and abstraction are common in long-form documents, semantic evaluation be-

yond lexical overlap is particularly important. Accordingly, financial generation and summarization studies often report BERTScore alongside ROUGE, as in ECTSum earnings-call summarization benchmark (Mukherjee et al., 2022).

Existing datasets support semantic similarity and numeracy analysis. STS-B (Cer et al., 2017) provides a general-domain benchmark for graded semantic similarity, while FinSTS (Liu et al., 2024) extends this to financial narratives. FinNum (Chen et al., 2019) focuses on numeral understanding by annotating functional roles of numbers in financial text. However, these benchmarks do not explicitly evaluate whether numerical relationships between corresponding mentions are preserved, motivating evaluation frameworks that decouple textual semantics from numerical verification.

## 7 Conclusion

We identify numerical blindness as a fundamental limitation of embedding-based semantic similarity metrics: context-dominated representations assign high similarity to numerically contradictory sentences. To address this issue, we propose **NASH**, a model-agnostic, scoring heuristic that decouples textual semantics from numerical verification. NASH employs a three-stage pipeline: (1) modal separation via masking, (2) context-aware numerical alignment with threshold gating, and (3) IDF-weighted aggregation of textual and magnitude-based similarity. Evaluations on NumFinE demonstrate substantial gains in numerical sensitivity, while experiments on established semantic similarity benchmarks (STS-B and Financial-STS) show that NASH preserves general semantic performance and delivers additional gains on financial texts. These establish that numerical consistency requires explicit treatment in quantitative domains and validate modal separation as an effective design principle.

## Limitations

Our work has several limitations. First, our evaluation focuses primarily on financial text, where numerical precision is critical. While NASH is model-agnostic and applicable to other numeracy-sensitive domains (e.g., medical, scientific text), we do not thoroughly validate its effectiveness beyond finance. Second, due to computational constraints, we evaluate a diverse but not exhaustive set of embedding models. We do not cover all checkpoints from large-scale benchmarks such as MTEB, nor perform repeated runs to establish statistical significance for minor performance differences. A broader evaluation could provide stronger robustness guarantees. Third, our evaluation relies on automatic diagnostic protocols without human-centered studies. While NumFinE exposes clear numerical failure modes, we do not assess whether NASH better supports human judgment or decision-making in practical workflows. Finally, NumFinE uses controlled numerical perturbations to isolate numerical effects precisely. However, because these variations are artificially injected rather than sourced from actual system outputs, they lack real-world organic noise. Consequently, this synthetic construction may not fully capture the diversity of naturally occurring numerical errors. As with any diagnostic benchmark, results should be interpreted as indicative of numerical sensitivity rather than comprehensive real-world performance.

## References


Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14. Association for Computational Linguistics.

Chung-Chi Chen, Hen-Hsen Huang, Yow-Ting Shiue, and Hsin-Hsi Chen. 2018. Numeral understanding in financial tweets for fine-grained crowd-based forecasting. In *Proceedings of the 2018 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 136–143.

Chung-Chi Chen, Hen-Hsen Huang, Hiroya Takamura, and Hsin-Hsi Chen. 2019. Overview of the ntcir-14 finnum task: Fine-grained numeral understanding in financial social media data. In *Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies*, pages 454–465.

Jianlyu Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. M3-embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 2318–2335. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

*Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced bert with disentangled attention. In *Proceedings of the International Conference on Learning Representations*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577. Association for Computational Linguistics.

M. G. Kendall. 1938. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93.

Takeshi Kojima and 1 others. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Proceedings of Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics.

Jiaxin Liu, Yi Yang, and Kar Yan Tam. 2024. Beyond surface similarity: Detecting subtle semantic shifts in financial narratives. In *Findings of the Association for Computational Linguistics: The 2024 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 2641–2652. Association for Computational Linguistics.

Zhuang Liu, Degen Huang, Kaiyu Huang, Zhuang Li, and Jun Zhao. 2020. Finbert: A pre-trained financial language representation model for financial text mining. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI-20)*, pages 4513–4519.

Renqian Luo, Liai Sun, Yingce Xia, Tao Qin, Sheng Zhang, Hoifung Poon, and Tie-Yan Liu. 2022. Biogpt: generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, 23(6):bbac409.

Pekka Malo, Ankur Sinha, Pyry Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

Rajdeep Mukherjee, Abhinav Bohra, Akash Banerjee, Soumya Sharma, Manjunath Hegde, Afreen Shaikh, Shivani Shrivastava, Koustuv Dasgupta, Niloy Ganguly, Saptarshi Ghosh, and Pawan Goyal. 2022. Ectsum: A new benchmark dataset for bullet point summarization of long earnings call transcripts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 10893–10906. Association for Computational Linguistics.

OpenAI. 2024. New embedding models and API updates.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892. Association for Computational Linguistics.

Karan Singhal and 1 others. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*.

Jason Wei and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Shijie Wu and 1 others. 2023. BloombergGPT: A large language model for finance. *Preprint*, arXiv:2303.17564.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Hongyang Yang and 1 others. 2023. FinGPT: Open-source financial large language models. *Preprint*, arXiv:2306.06031.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *Proceedings of the International Conference on Learning Representations*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, and 1 others. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Wei Zhao and 1 others. 2019. MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 563–578. Association for Computational Linguistics.

## A Evaluation on Cross-Pair Protocols

The Cross-Pair protocol tests for numerical sensitivity, whether numerical distance relationships generalize across different sentence contexts, requiring models to compare magnitudes without shared lexical overlap. As shown in Table 5, baseline models struggle with this task. Even `text-embedding-3-small` (OpenAI) achieves only 58.9% accuracy, indicating that general-purpose embeddings capture topical similarity but lack fine-grained magnitude sensitivity. This performance gap exposes a critical blind spot in general-purpose embeddings: while they excel at capturing topical similarity, they fail to encode the fine-grained magnitude information required for financial fidelity, validating NumFinE as a necessary benchmark for uncovering these latent limitations.

NASH substantially improves performance across many models. For example, `DeBERTa-xlarge-mnli` improves from 47.2% to 64.3% (+36.4%), while `all-MiniLM-L6-v2` gains from 59.8% to 63.9% (+6.7%). Notably, all NASH-enhanced open-source models outperform the `text-embedding-3-small` baseline (58.9%), demonstrating that explicit numerical alignment is crucial for quantitative consistency in financial NLP. However, some models show minor degradation (e.g., `bert-base-uncased`: -4.7%, `FinBERT`: -6.2%). The heightened difficulty of the Cross-Pair setting, compared to anchor-based evaluation, stems from the varying lengths and semantic structures of the paired sentences. Because the contextual density differs, the informational weight of a numerical value is diluted to varying extents in each sentence. Consequently, NASH currently struggles to perfectly calibrate the scale of similarity across such heterogeneous sentence pairs. Addressing this uneven numerical dilution and developing more robust cross-context normalization mechanisms remains an important direction for future work.

| **Model** | **Cross-Pair Accuracy** | |
|---|---|---|
| | Base | +NASH |
| *Token-level Encoders* | | |
| `bert-base-uncased` | 0.6727 | 0.6410 $_{\downarrow 4.7\%}$ |
| `FinBERT` | 0.6772 | 0.6354 $_{\downarrow 6.2\%}$ |
| `DeBERTa-xlarge-mnli` | 0.4715 | 0.6556 $_{\uparrow 39.1\%}$ |
| *Sentence-level Bi-encoders* | | |
| `all-MiniLM-L6-v2` | 0.5984 | 0.6404 $_{\uparrow 7.0\%}$ |
| `unsup-simcse-bert-base-uncased` | 0.6564 | 0.6330 $_{\downarrow 3.6\%}$ |
| `bge-m3` | 0.5798 | 0.6143 $_{\uparrow 6.0\%}$ |
| `Qwen3-Embedding-0.6B` | 0.6015 | 0.6398 $_{\uparrow 6.3\%}$ |
| `text-embedding-3-small` | 0.5892 | |

Table 5: **Evaluation on NumFinE (Cross-Pair).** Results show the original backbone (Base) versus the NASH-enhanced performance (+NASH). Subscripts denote relative improvement percentages. **Bold** indicates best performance in each column.

## B Threshold Sensitivity Analysis

NASH introduces a single hyperparameter, the alignment threshold $\tau$, which gates contextual numeric alignment. We evaluate sensitivity by varying $\tau$ over five values $\{\mu-2\sigma,\ \mu-\sigma,\ \mu,\ \mu+\sigma,\ \mu+2\sigma\}$, where $\mu$ and $\sigma$ are estimated from contextual similarity scores of randomly paired numbers.

Figure 3 shows that performance is stable across threshold values for all three evaluation protocols. Only the most restrictive setting ($\mu - 2\sigma$) shows minor degradation, indicating that NASH does not require fine-tuning of $\tau$ and is robust to reasonable threshold variations.

Table 7 reports the detailed numerical results corresponding to the sensitivity analysis. Across all evaluated models and metrics, performance varies only marginally across a wide threshold range, including on larger models such as Qwen3-0.6B, Qwen3-8B, and Llama-3.1-8B, indicating that the proposed threshold remains stable across model scales. These results further confirm that the alignment threshold generalizes well in practice without requiring dataset- or model-specific tuning.

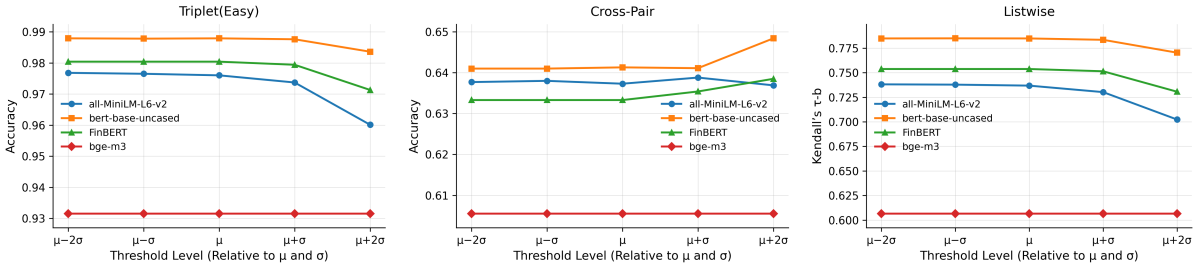


Figure 3: **Sensitivity analysis of alignment threshold $\tau$.** From left to right: Triplet accuracy, Cross-Pair accuracy, and Listwise Kendall's $\tau_b$, evaluated across five threshold levels $\{\mu - 2\sigma, \mu - \sigma, \mu, \mu + \sigma, \mu + 2\sigma\}$, where $\mu$ and $\sigma$ are estimated from contextual similarity scores of random number pairs.

| Category | Model Name | URL | Paper |
|---|---|---|---|
| **Token-level** | `bert-base-uncased` | google-bert/bert-base-uncased | Devlin et al. (2019) |
| | `FinBERT` | ProsusAI/finbert | Liu et al. (2020) |
| | `DeBERTa-xlarge-mnli` | microsoft/deberta-xlarge-mnli | He et al. (2021) |
| **Sentence-level** | `all-MiniLM-L6-v2` | sentence-transformers/all-MiniLM-L6-v2 | Wang et al. (2020) |
| | `bge-m3` | BAAI/bge-m3 | Chen et al. (2024) |
| | `unsup-simcse-bert-base-uncased` | princeton-nlp/unsup-simcse... | Gao et al. (2021) |
| | `Qwen3-Embedding-0.6B` | Qwen/Qwen3-Embedding-0.6B | Zhang et al. (2025) |
| | `Qwen3-0.6B` | Qwen/Qwen3-0.6B | Yang et al. (2025) |
| | `Qwen3-8B` | Qwen/Qwen3-8B | Yang et al. (2025) |
| | `Llama-3.1-8B` | meta-llama/Llama-3.1-8B | Grattafiori et al. (2024) |
| | `text-embedding-3-small` | OpenAI Platform Docs | OpenAI (2024) |

Table 6: **Baseline models: encoder type, parameter count, and source URLs.**

## C Experimental Details and Baseline Models

**Computational budget.** All experiments use inference-only evaluation with frozen pretrained encoders—no training or fine-tuning is performed. Computational cost is dominated by forward passes, with each complete evaluation run taking a few GPU-hours depending on model size and dataset.

**Baseline models.** Table 6 provides the specific model checkpoints and URLs.

## D Categorical Statistics of NumFinE

Table 8 shows the categorical distribution of valid evaluation units in NumFinE.

| Category | Subcategory | Count |
|---|---|---|
| Monetary | money | 1,784 |
| | quote | 1,187 |
| | change | 1,173 |
| | forecast | 585 |
| | buy price | 505 |
| | support / resistance | 290 |
| | sell price | 112 |
| | stop loss | 30 |
| Temporal | date | 3,635 |
| | time | 591 |
| Percentage | relative | 1,431 |
| | absolute | 826 |
| Quantity | quantity | 1,737 |
| Product Number | product number | 293 |
| Indicator | indicator | 272 |
| Option | exercise price | 125 |
| | maturity date | 75 |
| **Total** | | **14,651** |

Table 8: **Statistics of validated evaluation units by FinNum category (Chen et al., 2018).**

| Model | Triplet (Easy) | | | | | | Listwise | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Base | $\tau-2\sigma$ | $\tau-\sigma$ | $\tau$ | $\tau+\sigma$ | $\tau+2\sigma$ | Base | $\tau-2\sigma$ | $\tau-\sigma$ | $\tau$ | $\tau+\sigma$ | $\tau+2\sigma$ |
| *Token-level Encoders* | | | | | | | | | | | | |
| `bert-base-uncased` | 0.9214 | 0.9879 | 0.9878 | 0.9879 | 0.9857 | 0.9836 | 0.5409 | 0.7848 | 0.7849 | 0.7848 | 0.7834 | 0.7703 |
| `FinBERT` | 0.9186 | 0.9804 | 0.9804 | 0.9804 | 0.9795 | 0.9713 | 0.5344 | 0.7537 | 0.7537 | 0.7537 | 0.7514 | 0.7306 |
| *Sentence-level Bi-encoders* | | | | | | | | | | | | |
| `all-MiniLM-L6-v2` | 0.8440 | 0.9768 | 0.9765 | 0.9760 | 0.9743 | 0.9601 | 0.3450 | 0.7381 | 0.7378 | 0.7368 | 0.7289 | 0.7023 |
| `bge-m3` | 0.8335 | 0.9325 | 0.9325 | 0.9325 | 0.9325 | 0.9325 | 0.3481 | 0.6065 | 0.6065 | 0.6065 | 0.6065 | 0.6065 |
| `Qwen3-0.6B` | 0.9135 | 0.9908 | 0.9908 | 0.9908 | 0.9908 | 0.9908 | 0.4872 | 0.8129 | 0.8129 | 0.8129 | 0.8129 | 0.8129 |
| `Qwen3-8B` | 0.9109 | 0.9885 | 0.9885 | 0.9885 | 0.9885 | 0.9885 | 0.4946 | 0.8014 | 0.8014 | 0.8014 | 0.8012 | 0.8011 |
| `Llama-3.1-8B` | 0.9134 | 0.9902 | 0.9902 | 0.9902 | 0.9900 | 0.9894 | 0.5254 | 0.8097 | 0.8097 | 0.8097 | 0.8096 | 0.8090 |

Table 7: **Sensitivity analysis of the alignment threshold $\tau$ and generalization to larger language models.** Triplet (Easy) accuracy and Listwise ranking remain highly stable across threshold settings $\{\mu - 2\sigma,\ \mu - \sigma,\ \mu,\ \mu + \sigma,\ \mu + 2\sigma\}$. Consistent behavior across both encoder-scale models and larger models (e.g., Qwen3 and Llama-3.1) indicates that $\tau$ acts as a robust contextual compatibility gate rather than a precision-tuned hyperparameter.

## E Case Study on Contextual Numerical Alignmentt

To demonstrate how NASH robustly handles sentences with complex, interwoven numerical relationships, we conduct a targeted case study using a stress-test example. We evaluate NASH's contextual similarity matrices across three perturbation scenarios based on a base sentence ($s_5$). NASH performs a greedy one-to-one alignment based on these similarity scores.

The base sentence for all scenarios is:
$s_5$: "Revenue of **100M** grew **20%** while costs of **80M** grew **25%**, yielding **20M** profit."

**Scenario 1: Altered percentages & profit.** We alter the numerical values while keeping the syntactic structure identical.
$s_6$: "Revenue of **100M** grew **25%** while costs of **80M** grew **20%**, yielding **10M** profit."

**Scenario 2: Altered syntax, identical numbers.** We move the profit mention to the beginning of the sentence to test positional bias.
$s_7$: "**20M** profit was yielded as revenue of **100M** grew **20%** while costs of **80M** grew **25%**."

**Scenario 3: Same syntax, altered base amounts.** We double the base financial amounts while keeping the growth rates identical.
$s_8$: "Revenue of **200M** grew **20%** while costs of **160M** grew **25%**, yielding **40M** profit."

**Discussion.** As the highest similarity scores (bolded) in each matrix demonstrate, NASH consistently and perfectly aligns the correct numbers based on their semantic roles. Particularly in Scenario 2, despite the "20M profit" being moved to the very beginning of the sentence, NASH successfully avoids positional bias and correctly aligns it with the "20M profit" located at the end of $s_5$ (score: 0.87). This confirms that NASH robustly handles complex constraints by evaluating magnitude differences only between semantically equivalent entities, rather than relying on superficial numerical values or token positions.

**(a) Scenario 1: Altered percentages & profit**

| $s_5 \backslash s_6$ | **100** | **25** | **80** | **20** | **10** |
|---|---|---|---|---|---|
| **100** | **1.00** | 0.60 | 0.80 | 0.57 | 0.42 |
| **20** | 0.63 | **0.94** | 0.65 | 0.84 | 0.23 |
| **80** | 0.80 | 0.57 | **1.00** | 0.59 | 0.35 |
| **25** | 0.54 | 0.85 | 0.52 | **0.95** | 0.19 |
| **20** | 0.38 | 0.22 | 0.39 | 0.27 | **0.94** |

**(b) Scenario 2: Altered syntax (profit moved to front)**

| $s_5 \backslash s_7$ | **20** | **100** | **20** | **80** | **25** |
|---|---|---|---|---|---|
| **100** | 0.43 | **1.00** | 0.63 | 0.80 | 0.55 |
| **20** | 0.32 | 0.63 | **1.00** | 0.65 | 0.86 |
| **80** | 0.43 | 0.80 | 0.65 | **1.00** | 0.52 |
| **25** | 0.21 | 0.54 | 0.79 | 0.52 | **0.84** |
| **20** | **0.87** | 0.38 | 0.28 | 0.39 | 0.20 |

**(c) Scenario 3: Altered base amounts**

| $s_5 \backslash s_8$ | **200** | **20** | **160** | **25** | **40** |
|---|---|---|---|---|---|
| **100** | **0.89** | 0.63 | 0.70 | 0.54 | 0.39 |
| **20** | 0.62 | **1.00** | 0.60 | 0.79 | 0.24 |
| **80** | 0.78 | 0.65 | **0.82** | 0.52 | 0.43 |
| **25** | 0.51 | 0.79 | 0.46 | **1.00** | 0.19 |
| **20** | 0.39 | 0.28 | 0.38 | 0.21 | **0.91** |

Table 9: **Contextual similarity matrices for complex numerical relationships.** Rows represent the numbers from $s_1$, and columns represent the numbers from $s_2$ for each scenario. The bolded values indicate the highest similarity score for each row, representing the optimal semantic alignment.

## F Prompts Used

**Financial Sentence Validation JSON Prompt**

Key Requirements:

You are a strict financial text validator.
You ONLY output strict JSON, nothing else.
Decide if the input sentence is valid or invalid based on numeric/time rules.

Input:
**Sentence:** {sentence}

Rules:
- Only treat as invalid in cases like:
  * Invalid time/date format
    Example: `"$CMCM at $10.11 - Buy Stock Market Alert sent to members at 9:4 AM ET #stocks"`
    → reason: `"Time format '9:4 AM' is invalid; minutes should have two digits (09:04)."`
Note: Shortened but still valid expressions like `"7:00"` (without AM/PM) or `"11/14"` (without year) are acceptable and should NOT be flagged.
  * Numeric logic inconsistent with the claim
    Example: `"$PEIX,is Head-Fake news, all experts point to $6.02 to $14 dollars per share that is 130% upside from here. Art of deal says buy More !"`
    → reason: `"Numeric logic inconsistent: going from $6.02 to $14 is more than a 130% increase."`

Output format (strict JSON):
```
{
  "valid": true/false,
  "reason": "..."  # reason required only if invalid
}
```

**Number Identification Prompt**

You are a financial text classifier. For each input sentence, identify **all numeric values** (numbers or percentages) present. For each numeric value, output a separate classification row.
Use the FinNum Categories and Subcategories below for classification. The category must start with an uppercase letter. If the subcategory is not the same as the category, start it with a lowercase letter; otherwise, start it with an uppercase letter.

Categories and Subcategories:

- **Monetary**: money, quote, change, buy price, sell price, forecast, stop loss, support or resistance
- **Percentage**: relative, absolute
- **Option**: exercise price, maturity date
- **Indicator**: Indicator
- **Temporal**: date, time
- **Quantity**: Quantity
- **Product Number**: Product Number

Instructions:

- Output exactly one classification per block with this format:

```
Sentence: "<full sentence>"
Target: "<number>"
Category: "<Category>"
Subcategory: "<Subcategory>"
```

- Output digits only (no symbols, but keep commas).
- If no subcategory applies, output "None" for both Category and Subcategory.
- Repeat the sentence for each numeric value.
- If the sentence contains no numeric values, output a single classification with Target: None, Category and Subcategory both "None".
- If there is only a month and year without a specific date, output only the year.

Example:
```
Sentence: "During 2024, the Company's net sales through its direct and indirect distribution channels accounted for 38% and 62%, respectively, of total net sales."
Target: 2024
Category: Temporal
Subcategory: date

Sentence: "During 2024, the Company's net sales through its direct and indirect distribution channels accounted for 38% and 62%, respectively, of total net sales."
Target: 38
Category: Percentage
Subcategory: absolute

Sentence: "During 2024, the Company's net sales through its direct and indirect distribution channels accounted for 38% and 62%, respectively, of total net sales."
Target: 62
Category: Percentage
```

```
Subcategory: absolute

Sentence: "Our common stock has experienced a
low of $138.80 per share."
Target: 138.80
Category: Monetary
Subcategory: quote
Sentences:
```

## G Numerical Augmentation Details

We describe the category-specific random augmentation strategies used to construct NumFinE. All perturbations preserve sentence structure while varying only numerical values.

**Temporal.**

- *Date*: Randomly shift the date within ±30 days, bounded by common constraints (e.g., days in month).
- *Time*:
  - If base is 0: sample from [1,60].
  - If base $\leq$ 12: sample from [1,12].
  - If base $<$ 60: sample from [1, 60].
  - If base $\geq$ 60: multiply by random factor in [0.8, 1.2].

**Monetary.**

- *Money, quote, forecast, buy/sell price, support or resistance, stop loss*: Multiply by random factor in [0.5, 2.0].
- *Change*: Multiply by random factor in [0.5, 2.0].

**Percentage.**

- *Relative, absolute*: Multiply by random factor in [0.5, 2.0].

**Quantity.**

- If base $\leq$ 5: multiply by random factor in [0.25, 4.0].
- Otherwise: multiply by random factor in [0.5, 2.0] (round to integer).

**Product Number.** Multiply by random factor in [0.33, 3.0].

**Indicator.** Multiply by random factor in [0.33, 3.0].

**Option.**

- *Maturity date*: Shift randomly within ±30 days, bounded by common constraints.
- *Exercise price*: Multiply by random factor in [0.33, 3.0].

## H Human Validation of the Quality Assurance Process

To validate the automated filtering process conducted by GPT-4o-mini in Section 4.3, we recruited three human annotators with backgrounds in computer science and finance. These annotators were tasked with evaluating a randomly sampled subset of 50 instances. The final labels regarding grammatical validity and semantic plausibility were determined via majority voting, establishing a high-quality reference set. We then evaluated the GPT-4o-mini quality assurance pipeline against this human-annotated reference set. The automated filter achieved 80% accuracy, 95% precision, and 83% recall, indicating that it effectively removes invalid records while reliably preserving high-quality ones.