BRIDGING EXPERT KNOWLEDGE AND AUTOMATED FEATURE ENGINEERING VIA SELF-EVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

In high-stakes settings such as brand compliance, clinical care, and content moderation, machine learning cannot be deployed as opaque oracles: practitioners must inspect the features driving model decisions, and models must leverage the expert documentation governing these domains. In practice, the data arrives as unstructured content, and features extracted from it must be interpretable, discriminative, and aligned with what experts consider important. Existing methods fall short: they target tabular inputs, lack demonstrated expert alignment, and cannot operationalize qualitative criteria such as "maintain professional tone" into precise features. We present FEST (Feature Engineering with Self-evolving Trees), combining dual-stream feature generation (semantic and deterministic), semantic deduplication, and tree-guided iterative evolution to discover auditable features from raw text and images. FEST leads in 17 of 20 classifier-task combinations across brand classification, content authenticity detection, and stress detection, with a mean gain of 4.2 pp over the strongest baseline across five classifiers. An LLM-as-judge evaluation shows FEST achieves 60–80% coverage of expert-designed brand features at strict semantic-alignment thresholds. Separately, practitioners rate FEST features highly on relevance, clarity, and actionability. When seeded with expert guidelines, FEST refines qualitative criteria into operational features, improving accuracy by 6–12 pp on average across brands. To enable systematic evaluation of expert alignment in automated feature engineering, we release BrandGuide, the first dataset pairing expert-designed features with 1M+ assets across 2,683 unique brands. By grounding feature engineering in expert knowledge, FEST opens a practical pathway for interpretable ML in domains demanding human oversight.

1 INTRODUCTION

Practitioners cannot deploy machine learning systems they cannot interrogate. In high-stakes domains such as clinical decision support, content moderation, and brand compliance, machine learning and domain expertise must operate in conjunction: decisions affect people in ways that demand both automation at scale and expert oversight against the standards practitioners maintain (Rudin, 2019). A misjudged advertising campaign can damage a brand and trigger public backlash and regulatory scrutiny (Cresci, 2016); a clinical risk model whose decision criteria a physician cannot verify against medical knowledge cannot be trusted; a content classifier whose flagging criteria a moderator cannot audit cannot be deployed. This requirement runs in two directions. First, the features driving automated decisions must be ones experts can inspect, validate, and recognize as domain-relevant. Second, when experts have already documented domain criteria (brand style guides, clinical protocols, editorial standards), automated systems must ingest and apply this accumulated knowledge rather than discover features from scratch. Features are the natural interface for both directions: they are what models operate on, and what experts can match against their own specifications.

This makes feature engineering the critical bridge between automated systems and domain expertise. The core technical challenge has two parts: producing features from raw unstructured data that are simultaneously discriminative, interpretable, and aligned with what experts regard as meaningful; and ingesting existing expert documentation when available. Post-hoc explanation methods (LIME, SHAP, GradCAM; Ribeiro et al., 2016; Lundberg & Lee, 2017; Selvaraju et al., 2017) are not a substitute: they produce approximate attributions over a black-box model's internal representations, not features that practitioners can compare against expert specifications. Producing such features automatically, with empirical evidence of expert alignment, is the problem this paper addresses. Even measuring this alignment requires ground-truth expert features paired with the data they describe, a benchmark that did not exist prior to this work (§4).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Existing automated feature engineering methods do not meet this requirement (Appendix A). Classical systems such as AutoFeat and OpenFE (Horn et al., 2019; Zhang et al., 2023) search predefined transformations over tabular columns and cannot operate on raw text or images. Recent LLM-based approaches narrow part of the gap: FeatLLM (Han et al., 2024) and LLM-FE (Abhyankar et al., 2025) use language models for feature discovery but remain restricted to tabular inputs, while Felix (Malberg et al., 2024) extends to unstructured text using single-pass generation without iterative refinement. Crucially, all of these systems optimize for downstream accuracy alone: they do not demonstrate that the features they discover match what experts consider important, and they provide no mechanism for incorporating existing expert documentation.

The second part of the challenge arises when expert documentation is available, which is common: brand managers maintain style guides, clinicians follow diagnostic protocols, content platforms publish editorial standards. These specifications encode valuable domain knowledge but are expressed as high-level qualitative criteria ("maintain professional tone" or "use high-quality product images") rather than reproducible, measurable features. LLMs produce inconsistent outputs when directly prompted with such ambiguous criteria (Zheng et al., 2023). Operationalizing expert knowledge requires transforming qualitative guidelines into precise features grounded in empirical data: a system must learn what "professional tone" concretely means by observing examples that satisfy or violate it. No prior system has demonstrated both expert-aligned feature discovery from unstructured data and the ability to operationalize expert documentation.

We present **FEST** (Feature Engineering with Self-evolving Trees), an iterative framework that generates semantic features (LLM-assessed, e.g., "professional tone") and deterministic features (executable code, e.g., emoji count) from contrastive sample pairs, then consolidates and refines them based on discriminative power. This same engine supports two complementary modes. *Without expert seeding*, FEST discovers features from unstructured data that achieve 60–80% coverage of expert-authored specifications under strict LLM-as-judge evaluation; separately, practitioners rate their relevance, clarity, and actionability above 3.8/5. *With expert guidelines*, FEST operationalizes ambiguous criteria into precise definitions while discovering complementary patterns, improving accuracy by 6–12 pp on average across brands. Across brand classification (text/images), content authenticity detection, and stress detection, FEST leads in 17 of 20 classifier-task combinations (mean gain 4.2 pp), with ablations confirming complementary contributions of the semantic and deterministic streams. Even measuring expert alignment requires ground-truth expert features paired with the data they describe. To spur research in this direction, we release **BrandGuide**, the first dataset pairing expert-designed features with unstructured content (§4). **Our key contributions:**

1. **Problem formalization**: We formalize a deployment-critical problem: producing interpretable features from unstructured data that domain experts recognize as meaningful, and operationalizing expert documentation when available. We bring this problem to the community and propose expert alignment as a measurable objective for automated feature engineering.
2. **BrandGuide dataset**: To enable systematic evaluation of expert alignment in automated feature engineering, we release **BrandGuide**, the first dataset pairing expert-designed features with unstructured content: 1M+ assets across 2,683 unique brands, 80 sectors, and 103 regions.
3. **FEST framework**: We propose FEST, combining dual-stream feature generation, semantic deduplication, and tree-guided evolution. FEST leads in 17 of 20 classifier-task combinations across five classifiers (mean gain 4.2 pp), while maintaining interpretability.
4. **Expert alignment**: FEST achieves 60–80% coverage of expert features under strict LLM-as-judge thresholds (Felix drops to 0% on certain brands). Separately, practitioners rate FEST features above 3.8/5 on relevance, clarity, and actionability.
5. **Expert operationalization**: With expert seeds, FEST operationalizes qualitative criteria into more precise features, improving accuracy by 6–12 pp on average across brands.

2 Problem Formulation

Given training data $\mathcal{D}_{\text{train}} = \{(x_i, y_i)\}_{i=1}^{n}$ with raw inputs $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$, we seek an optimal feature set $\boldsymbol{\phi}^\star = \{\phi_1, \ldots, \phi_{|\boldsymbol{\phi}|}\}$ from hypothesis space Φ of feature functions $\phi : \mathcal{X} \to \mathbb{R}$:

$$\begin{aligned} \boldsymbol{\phi}^\star \;=\; \arg\max_{\boldsymbol{\phi} \subset \Phi} \quad & \mathcal{J}(f,\, \boldsymbol{\phi}\,;\, \mathcal{D}_{\text{train}}) \\ \text{s.t.} \quad & \forall\, \phi_j \in \boldsymbol{\phi} :\; \phi_j \text{ is interpretable} \end{aligned} \tag{1}$$

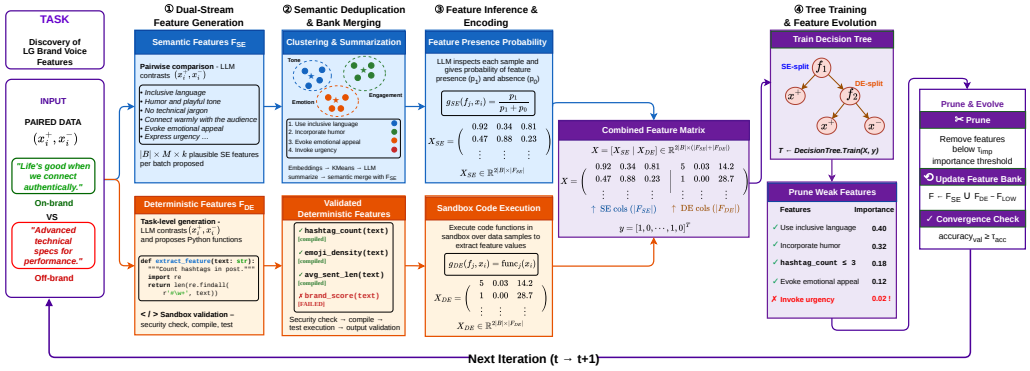


Figure 1: One iteration of FEST on LG brand voice feature discovery. Given paired on-brand and off-brand samples, FEST executes two complementary streams: (i) a *semantic (SE)* stream, where an LLM generates pairwise discriminative hypotheses (e.g., "inclusive language," "emotional appeal"), followed by semantic clustering and deduplication; and (ii) a *deterministic (DE)* stream producing executable Python functions (e.g., `num_hashtags`, `count_punctuations`). Semantic features are encoded via LLM exponentiated log-probability confidence scores; deterministic features yield numeric values through execution. The concatenated representation trains a decision tree whose importance scores guide iterative pruning and refinement, yielding a compact, discriminative feature bank.

where $f : \mathbb{R}^{|\phi|} \to \mathcal{Y}$ is a downstream classifier operating on feature representation $\phi(x) = [\phi_1(x), \ldots, \phi_{|\phi|}(x)]$, and $\mathcal{J}$ measures empirical accuracy: $\mathcal{J}(f, \phi; \mathcal{D}_{\text{train}}) = \frac{1}{|\mathcal{D}_{\text{train}}|} \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \mathbb{I}(f(\phi(x)) = y)$. Notably, the interpretability constraint is what distinguishes this formulation from standard accuracy maximization: each ϕ_j must be expressible as a natural-language predicate or a short executable function, ensuring the resulting model is auditable by domain practitioners. Feature bank size is not imposed as a hard constraint but is empirically stabilized through semantic clustering (§3.2.2).

Expert seeding. When expert-authored guidelines $G = \{g_1, \ldots, g_m\}$ are available, the hypothesis space Φ is seeded with features derived from G. The system should operationalize these qualitative specifications into measurable features and discover complementary patterns beyond the documentation. The optimization objective remains unchanged (§3.3).

Distinction from classical AFE. Classical systems (Horn et al., 2019; Zhang et al., 2023) operate on tabular inputs $\mathcal{X} = \mathbb{R}^m$ with a fixed transformation grammar over predefined columns. Here, $\mathcal{X}$ consists of raw unstructured inputs (text, images) with no column structure, and Φ is the open-ended space of interpretable predicates. Even *proposing* plausible features requires a generative model, converting an intractable search into a guided discovery process.

Terminology. *Expert guidelines* are qualitative criteria authored by practitioners (e.g., "use inclusive language"). *Features* are operational characteristics: either natural-language descriptions assessed via LLM confidence or executable functions. *Feature encodings* are the numerical scores from applying features to samples; *feature representations* are the concatenated vectors fed to the classifier.

3 METHODOLOGY

FEST operates in two modes corresponding to the two parts of the problem formulated above (§1). In *discovery mode*, the system receives only labeled data and discovers features from scratch. In *expert-seeded mode*, the system additionally receives expert-authored guidelines that initialize the feature bank, which FEST then refines and augments through the same iterative process. The architecture is identical in both modes; the only difference is initialization. We describe each component below.

FEST iteratively discovers and refines ϕ through LLM-guided generation and tree-based selection. We now describe FEST's methodology, organized around three key architectural innovations: (1) dual-stream discovery generating both semantic and deterministic features, (2) semantic consolidation via conditional embeddings and clustering, and (3) tree-guided iterative evolution using importance-based pruning. We explain each of them next. Figure 1 illustrates the complete pipeline, and Algorithm 1 provides the formal pseudocode.

3.1 Pairwise Comparison for Relative Discrimination

Rather than analyzing samples in isolation, FEST discovers features through pairwise comparisons of positive and negative instances $\mathcal{P} = \{(x_i^+, x_i^-)\}$, which naturally filters common attributes and surfaces discriminative characteristics (Geng et al., 2025). Pairwise comparison is used only for feature *discovery*; feature *encoding* (§3.2.3) scores each sample independently, ensuring generalization to unseen instances. See Appendix D for extended motivation.

3.2 Iterative Feature Discovery and Refinement

FEST processes training pairs in sequential batches, with each iteration comprising four stages: dual-stream feature generation, semantic deduplication, feature inference and encoding, and tree-based evolution. We initialize with feature bank $F = \emptyset$ (or with expert-designed features when available, see Section 3.3) and importance history $H_{importance} = \emptyset$, then iterate until convergence (validation accuracy exceeds threshold $\tau_{accuracy}$) or data exhaustion.

3.2.1 Dual-Stream Feature Generation

Human-interpretable features span a spectrum from perceptual patterns requiring interpretation ("professional tone", "natural daylight") to precisely measurable quantities ("image brightness", "capitalization ratio"). The key distinction is epistemological: some features can only be assessed through judgment, while others are deterministically computable by code. To capture both, FEST generates two complementary feature types in each iteration t:

Semantic (SE) Features capture perceptual, interpretive characteristics that require judgment to assess. For each pair (x^+, x^-) in batch B, we prompt an LLM with multiple templates to generate M features explaining what differentiates x^+ from x^-. This pairwise prompting occurs for every pair, creating a large pool F_{SE} of candidate features that capture diverse perspectives on discriminative patterns.

Deterministic (DE) Features capture objective, precisely measurable characteristics through executable Python functions. Given the overhead of code generation and validation, we operate at task-level rather than pair-level: the LLM analyzes a small subset of pairs from B (typically 5) along with the task description to propose general deterministic features (e.g., "image aspect ratio", "emoji count"). Each proposed feature is implemented as Python code, executed in a sandbox on validation samples, and iteratively refined if errors occur. Task-level generation produces focused set F_{DE} of general features rather than pair-specific measurements, and the LLM's awareness of previously generated deterministic features prevents redundancy across iterations.

This dual-stream architecture provides complementary representations: semantic features capture nuanced perceptual patterns (e.g., "emotional storytelling," "natural daylight") while deterministic features provide precise quantitative measurements (e.g., capitalization ratio, color saturation), together enabling richer feature spaces than either alone. At all LLM prompting stages (generation, summarization, encoding), we explicitly instruct the model to exclude superficial brand identifiers (names, logos, hashtags) to prevent shortcut learning (see Appendix T for exact constraints).

3.2.2 Semantic Deduplication and Bank Merging

The semantic stream can generate thousands of features in F_{SE} across pairs in a batch, many expressing identical concepts through varied phrasing ("uses informal language", "casual tone", "conversational style"). Without deduplication, this inflates dimensionality and dilutes signal. We perform semantic consolidation through three steps: (1) compute conditional embeddings of all features in F_{SE} conditioned on the task description, by prepending the task description as an instruction prefix before encoding with Qwen3-Embedding-4B (Zhang et al., 2025), making the embedding space domain-aware so that features with similar semantic roles cluster together regardless of surface phrasing, (2) cluster semantically similar features using K-means, and (3) prompt an LLM to summarize each cluster into a single representative feature preserving the core concept. This reduces redundancy while amplifying true signal by unifying equivalent features, producing deduplicated set $\bar{F}_{SE}$.

These deduplicated semantic features $\bar{F}_{SE}$ are then merged with existing semantic features in feature bank F through semantic similarity checking to prevent duplication with previous iterations. Deter-

ministic features F_{DE} require no explicit deduplication: their task-level generation, small number per iteration and LLM context-awareness about prior features naturally prevent redundancy. This produces the current iteration's complete feature set: $F^{(t)} = F \cup \bar{F}_{SE} \cup F_{DE}$, where $|F^{(t)}| = |F_{SE}| + |F_{DE}|$ with $|F_{SE}|$ denoting total semantic features and $|F_{DE}|$ denoting total deterministic features.

3.2.3 Feature Inference and Encoding

While pairwise comparisons guide feature *discovery*, encoding scores each sample independently to ensure generalization. Each pair (x_i^+, x_i^-) is split into individual samples. Semantic features are encoded via LLM log-probability confidence: the LLM evaluates whether feature f_k is present or absent in sample x, and we compute a normalized confidence score from the output probabilities, yielding a continuous value in $[0, 1]$ that captures LLM certainty (Appendix C). Deterministic features are encoded by executing the associated Python function directly. The two feature matrices are concatenated into a combined representation $\mathbf{X} = [\mathbf{X}_{SE} \mid \mathbf{X}_{DE}]$ for decision tree training. Decision trees handle the heterogeneous types (probability scores and numeric values) naturally via Gini impurity reduction without normalization.

3.2.4 Tree-Based Evolution and Convergence

We train a decision tree classifier $\mathcal{T}$ on $(\mathbf{X}, \mathbf{y})$ and evaluate on a held-out validation set $\mathcal{P}_{val}$. Decision trees serve multiple critical purposes for FEST:

Why Decision Trees? Trees provide (1) transparent decision paths for practitioner inspection, (2) feature importance via impurity reduction that directly guides evolution, and (3) automatic threshold learning (e.g., "emoji count < 5") that operationalizes vague criteria into precise splits.

Evolution Mechanism: After training $\mathcal{T}$, we extract feature importance vector $\mathbf{I}$ and update history $H_{importance}$ tracking scores across iterations. Features in $F^{(t)}$ with consistently low importance over the last three iterations (mean importance below threshold $\tau_{importance}$) are pruned, producing $F^{(t)'}$ which becomes the global feature bank for the next iteration: $F \leftarrow F^{(t)'}$. This history-based pruning prevents spurious removal from single-iteration noise, ensuring only genuinely weak features are discarded. The feature bank thus evolves: high-value features persist while weak features are pruned, focusing capacity on predictive patterns.

Convergence: The algorithm terminates when validation accuracy exceeds $\tau_{accuracy}$ or all training pairs are processed. A held-out test set is used only for final evaluation. See Algorithm 1 for full procedural details.

3.3 Leveraging Expert Knowledge

When expert-designed features exist (e.g., brand style guidelines, clinical criteria), FEST initializes the feature bank F with these features rather than starting from $\emptyset$. This enables three capabilities: (1) *Validation*: the iterative process retains high-importance expert features while pruning weak ones. (2) *Refinement*: data-driven hypotheses often produce more specific reformulations of expert principles; the decision tree selects whichever version is more discriminative (e.g., "high-quality images" $\rightarrow$ "close-up shots with resolution > 1000 pixels showing product texture"). (3) *Augmentation*: FEST discovers features beyond expert knowledge, surfacing patterns experts may not have articulated.

4 BrandGuide Dataset

Evaluating expert alignment requires ground-truth expert features paired with the content they describe, yet no such benchmark exists. In practice, organizations employ brand strategists who formulate comprehensive guidelines codifying how brand assets must be constructed: color palettes with exact codes, typography hierarchies, logo placement rules, and tone-of-voice specifications. These represent deliberate, expert-validated design decisions, precisely the kind of domain knowledge that automated systems should be evaluated against.

We release **BrandGuide**, pairing these expert specifications with corresponding brand assets. Our pipeline extracts structured guidelines across **2,683** unique brands from the web and retrieves corresponding imagery, connecting expert-defined rules to their practical instantiations. The dataset

comprises **1M+** brand images and text across 80 sectors, 103 regions, and 28 languages (2014–2025). See Appendix U for collection methodology, verification, examples, and statistics.

5 EXPERIMENTS

5.1 TASKS AND DATASETS

We evaluate FEST across multiple tasks demonstrating its effectiveness in high-stakes domains requiring interpretable features:

1. **Brand Classification**: Brand consistency is critical for business success: consistent brand presentation increases revenue by 10–20% on average (Marq (formerly Lucidpress), 2024), while strong brand identity drives customer recognition, loyalty, and competitive differentiation (Acar et al., 2024; Keller, 2003). However, maintaining consistent brand identity across marketing campaigns is challenging as inconsistent messaging damages brand equity and customer trust. We classify social media content as on-brand or off-brand for 5 brands (Porsche, Adobe, Emirates, Louis Vuitton, Pizza Hut) spanning automotive, technology, aviation, luxury fashion, and food service sectors, using posts from the EngagingImageNet dataset (Khurana et al., 2025). Both modalities are evaluated: *text* (captions, based on linguistic style and tone) and *images* (promotional visuals, based on aesthetics and composition). Posts from competing brands in the same sector serve as off-brand samples.
2. **Content Authenticity Detection**: Distinguishing AI-generated from human-written content is increasingly important for content moderation and information integrity. We evaluate FEST on detecting whether a story is written by humans or AI systems using the GPT-GC dataset from Zhou et al. (2024).
3. **Stress Detection**: Identifying psychological stress from text enables mental health support applications. We evaluate FEST on detecting whether Reddit post authors exhibit stress using the Dreaddit dataset (Turcan & McKeown, 2019), which contains posts from stress-related and neutral subreddits.

Data Splitting. For each task, data is pre-partitioned into three disjoint sets: (1) training pairs for iterative feature generation, (2) a validation set $\mathcal{P}_{val}$ used inside the FEST loop for convergence monitoring, and (3) a held-out test set $\mathcal{P}_{test}$ used only for final evaluation and reported in all results tables.

5.2 BASELINE METHODS

We benchmark FEST's performance against baseline methods. There are two main components in each method; the feature generator and the classifier.

Feature Generators: For feature generation, we employ 3 different backbones. The first is an LLM that discovers features given just the task description. The second backbone is similar, with the only difference being that the LLM is also passed few-shot examples. The third backbone is Felix (Malberg et al., 2024), which generates features from pairs, clusters them, and creates feature value vectors using an LLM. For the zero-shot and few-shot cases, feature vectors are obtained using the same feature inference pipeline as FEST.

Downstream Classifiers: We use the feature vectors obtained from each of the above backbones to train different downstream classifiers, namely, decision tree (DT), logistic regression (LR), random forest (RF), multi-layer perceptron (MLP), and XGBoost (XGB). Classifier hyperparameters are fixed identically across all feature generators so that accuracy differences isolate feature quality rather than confounding it with classifier optimization. Using multiple classifiers demonstrates that FEST's feature quality generalizes beyond tree-based learners.

All methods use GPT-4o-mini (OpenAI, 2024) as the LLM backbone (temperature settings in Appendix S). Since all baselines use the identical LLM on identical content, any data contamination benefits all methods equally; FEST's consistent gains across both brand and non-brand datasets confirm the improvements are methodological (Appendix O).

6 RESULTS AND DISCUSSION

We evaluate both parts of the problem (§2): whether FEST discovers discriminative, expert-aligned features, and whether it can operationalize expert documentation. We organize the evaluation into three tiers:

1. **Task performance** (§6.1): Classification accuracy across four tasks, five classifiers, and ablations isolating the contribution of each feature stream. This establishes that FEST features are discriminative.
2. **Expert evaluation** (§6.2): For brand classification, where expert guidelines exist, we measure (a) alignment with expert-authored specifications through LLM-judged coverage and (b) practical utility through human ratings of feature relevance, clarity, and actionability.
3. **Expert knowledge operationalization** (§6.3): A controlled experiment measuring FEST's ability to ingest, refine, and augment existing expert guidelines into more precise and discriminative feature sets.

6.1 Task Performance

Table 1 presents classification accuracy across four tasks using five classifiers and four feature generation methods. Brand classification results are averaged across 5 brands; brand-wise breakdowns appear in Appendix E (Table 2, 3).

Table 1: Classification accuracy (%) across tasks and downstream classifiers. FEST leads in 17 of 20 classifier-task combinations across brand classification (text and images), AI content detection, and stress detection. Results shown for Decision Tree (DT), Logistic Regression (LR), Random Forest (RF), MLP, and XGBoost (XGB) classifiers, all using GPT-4o-mini (OpenAI, 2024) as the LLM backbone. Bold indicates best performance per classifier-task combination. Brand classification results averaged over 5 brands; brand-wise breakdowns in Appendix E.

Clf.	Feature Generator	Task			
		Brand Cl. (Text)	Brand Cl. (Images)	Cont Auth.	Stress Det.
DT	Zero-Shot LLM	73.29	65.21	70.40	74.40
	Few-Shot LLM	72.98	71.29	74.00	76.80
	Felix	75.86	67.82	81.60	75.60
	FEST (Ours)	**81.70**	**78.32**	**91.20**	**78.00**
LR	Zero-Shot LLM	69.50	69.82	83.60	72.80
	Few-Shot LLM	72.96	73.05	60.80	67.20
	Felix	75.70	69.72	**88.40**	76.40
	FEST (Ours)	**81.59**	**78.92**	84.00	**81.60**
RF	Zero-Shot LLM	80.27	74.46	81.20	78.40
	Few-Shot LLM	83.17	78.71	86.80	77.60
	Felix	80.03	70.58	88.80	**84.40**
	FEST (Ours)	**85.11**	**81.55**	**97.20**	83.60
MLP	Zero-Shot LLM	75.96	70.28	78.40	65.60
	Few-Shot LLM	76.75	73.49	64.40	65.20
	Felix	78.26	68.82	87.60	**79.60**
	FEST (Ours)	**81.57**	**76.56**	**88.40**	78.40
XGB	Zero-Shot LLM	79.11	73.02	85.20	75.20
	Few-Shot LLM	83.04	76.86	83.60	77.20
	Felix	80.68	71.67	91.20	79.60
	FEST (Ours)	**84.58**	**81.29**	**94.40**	**80.80**

FEST leads in 17 of 20 classifier-task combinations (mean gain of 4.2 pp over the respective strongest baseline), with ablations confirming complementary contributions of both feature streams (Table 1). Gains are consistent across brand classification on both modalities and content authenticity detection, where FEST's largest single margin reaches 8.4 pp over Felix (RF). While FEST wins on average, brand-wise analysis reveals outliers where baselines occasionally excel for specific brands (Appendix E). On stress detection, Felix shows competitive performance, indicating semantic-only features may suffice for some psychological tasks.

Failure analysis. FEST underperforms in 3 of 20 classifier-task combinations, all involving tasks where semantic-only features are sufficient. On stress detection, Felix leads with RF (84.4% vs. 83.6%) and MLP (79.6% vs. 78.4%), and on content authenticity with LR (88.4% vs. 84.0%). The common pattern: these are short-text tasks where deterministic features (sentence length, punctuation counts) add limited signal beyond semantic descriptions, and Felix's larger unconstrained feature pool captures marginal semantic variations. FEST's controlled feature space (30 features per iteration via K-means) trades exhaustive coverage for precision, a net positive on brand tasks but occasionally limiting on simpler domains.

Generalization across classifiers. Despite using decision trees internally for evolution, FEST features transfer effectively to all five downstream classifiers (DT, LR, RF, MLP, XGBoost), demonstrating genuine discriminative patterns rather than tree-specific artifacts.

Dual-stream features outperform semantic-only approaches. Felix (Malberg et al., 2024), the strongest baseline for unstructured data, generates only semantic features through single-shot LLM prompting. An ablation study (SE-only, DE-only, SE+DE) across 7 tasks confirms complementarity: SE+DE wins 11 of 14 DT+RF combinations. DE-only is always weakest, confirming that semantic features form the core signal while deterministic features add complementary measurable precision. Full ablation in Appendix F.1.

Sources of gain. FEST's improvements arise from the interplay of pairwise contrastive discovery, dual-stream generation, iterative importance-based pruning, and semantic deduplication (300+ candidates consolidated to 30 per iteration).

Robustness to shortcuts. To verify FEST learns genuine voice/style features rather than exploiting brand identifiers, we generated synthetic off-brand content using GPT-4o-mini that matches each brand's topics but uses generic language. FEST achieves 84.4% (Adobe), 79.8% (LG), 91.2% (Porsche) on this controlled setup, and the top features are purely stylistic (e.g., "instructional tone," "exclamation usage," "sentence length variance"). See Appendix N for details.

Stability and efficiency. Across 3 independent seeds (5 brands $\times$ 5 classifiers), FEST produces consistent feature banks with standard deviations mostly below 3pp for text classification; image and smaller-dataset tasks show slightly higher variance driven by classifier sensitivity rather than feature instability (Appendix M). K-means deduplication (k=30) is the stabilizing mechanism: stochastic LLM generation converges to consistent representative features after clustering and summarization. FEST averages \$0.10 per run vs. Felix's \$8.62 (86$\times$ cheaper) and completes in 15.9 minutes vs. 69.4 minutes (4.4$\times$ faster), consuming 91$\times$ fewer tokens. This efficiency stems from iterative pruning: features eliminated by decision tree importance are not re-encoded in subsequent iterations (Appendix L).

6.2 Expert Validation

Classification accuracy alone is insufficient for high-stakes deployment: discovered features must also align with what domain experts consider important. Expert brand guidelines are the authoritative specifications that practitioners use to ensure brand consistency, and commercial compliance platforms (Adobe, 2024; Canva, 2024) ingest them as ground truth. When FEST-discovered features align with these guidelines, it provides direct evidence that FEST operates on the same semantic dimensions as human experts. We evaluate two complementary properties: semantic coverage through an LLM-as-judge protocol and practical utility through a human expert study.

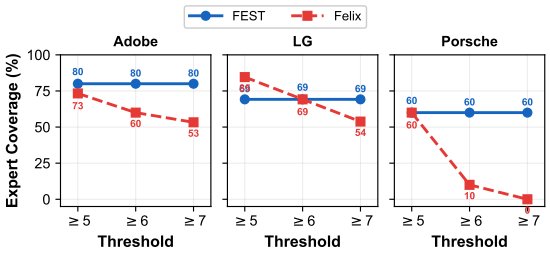


Figure 2: LLM-as-judge expert coverage (%) at thresholds 5–7 for three brands (Adobe, LG, Porsche). FEST (solid) maintains stable coverage across all thresholds, confirming strong semantic matches (7–9/10). Felix (dashed) appears comparable at threshold 5 but collapses at stricter thresholds, reaching 0% for Porsche at $\geq$7. Full per-brand sweep in Appendix H.

LLM-as-Judge Protocol. For three brands (Adobe, LG, Porsche), we compare the top-20 features discovered by FEST and Felix against expert-designed brand voice guidelines. GPT-4o rates each (guideline, feature) pair on a 0–10 semantic alignment scale; a guideline is "covered" if any feature scores at or above the threshold. This provides interpretable coverage scores without reliance on embedding similarity cutoffs.

Coverage results. Under the GPT-4o judge, at threshold $\geq$ 7, FEST achieves 60–80% coverage across brands and remains perfectly stable from threshold 5 through 7: the features covering a guideline score 7–9/10, not borderline. Felix appears comparable at threshold 5, but collapses at higher thresholds, reaching 0% on one brand at threshold 7 (Figure 2). Full per-brand sensitivity, cross-judge validation, and judge-human agreement appear in Appendix H.

Why FEST achieves higher coverage. Both methods use the same backbone LLM (GPT-4o-mini) and contrastive samples, so the coverage gap stems from what happens *after* feature proposals: cluster summarization (vs. centroid picking), iterative pruning of generic features, and multi-stage language refinement each contribute to convergence toward expert-aligned formulations. See Appendix I for the full analysis.

Human expert study. Domain experts rated 15 FEST-discovered features per brand on Relevance, Clarity, and Actionability (1–5 scale). Across two brands, all three dimensions score above 3.8/5, well above the midpoint. The expert reviewed both refined and newly discovered features as a single blinded pool, assessing the practical utility of FEST's complete output. See Appendix K for the full protocol. Together, task performance, automated coverage, and practitioner ratings assess distinct properties of the feature bank.

6.3 Expert Knowledge Operationalization

Having established that FEST's discovery mode produces features aligned with expert knowledge (§6.2), we evaluate the complementary capability: can FEST operationalize qualitative specifications into precise, measurable features and discover complementary patterns? We conduct a controlled experiment using brand style guidelines as seed features.

Experimental setup. We extract expert-designed features from official brand style guides for three brands (LG, Porsche, Adobe), containing qualitative guidelines for brand identity (e.g., "use high-quality images," "maintain professional tone"). We evaluate FEST in two configurations: (1) *Evaluation Mode*, where expert features are used directly without modification; and (2) *Learning Mode*, where expert features initialize FEST's feature bank for full iterative discovery. Both configurations use identical train-test splits and evaluation protocols, isolating the effect of FEST's refinement mechanism.

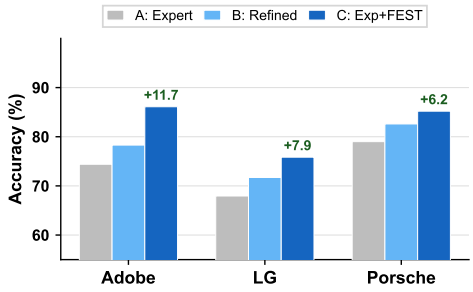


Figure 3: Expert feature refinement accuracy (averaged across DT, LR, RF, LLM classifiers) for text brand classification. A: expert features as-is, B: FEST features filtered to those aligned with experts (LLM-judge $\geq$ 7), C: full Expert+FEST. The A$\rightarrow$B$\rightarrow$C staircase shows that refinement improves most settings and augmentation improves all settings, with average A-to-C gains of 6.2–12 pp across brands. Per-classifier breakdown in Table 9 (Appendix F.2).

Results. Figure 3 summarizes the average-across-classifiers results (full per-classifier breakdown in Table 9, Appendix F.2). Three conditions disentangle the sources of improvement: (A) expert features alone, (B) expert refined, where FEST features are filtered to those aligned with expert features (LLM-judge score $\geq$ 7), isolating refinement without augmentation, and (C) Expert+FEST, the full augmented output. FEST's learning mode (C) consistently outperforms expert features alone (A), achieving average A-to-C gains of 6.2–12 pp across brands. Critically, B$>$A in most cases: refinement alone improves over static expert features (e.g., Adobe DT: A=78.4%, B=87.2%), demonstrating that FEST operationalizes ambiguous criteria into more discriminative formulations. C$>$B in all cases: augmentation provides further independent gains (e.g., Porsche LR: B=86.4%, C=90.0%), confirming FEST discovers complementary features beyond what refinement captures.

Analysis of feature evolution. FEST's feature bank evolves from expert seeds: some features are retained verbatim, ambiguous guidelines are refined into precise operational definitions, and novel features are discovered beyond the original documentation. See Appendix G for detailed examples and a visualization of each category.

7 Conclusion

Automated feature engineering and domain expertise have largely operated in isolation: existing methods primarily optimize predictive accuracy without evaluating alignment with expert priorities, while qualitative expert documentation remains difficult to translate into measurable features. FEST addresses both problems through complementary discovery and expert-seeded modes. FEST leads in 17 of 20 classifier-task combinations, achieves 60–80% coverage of expert specifications, receives practitioner ratings of 3.8/5 or higher for its complete feature banks, and improves accuracy by 6.2–12 pp across brands when operationalizing expert guidelines.

Our results suggest that, in the settings studied, architectural scaffolding for candidate feature generation and refinement can substantially narrow the gap between automated systems and expert knowledge. FEST combines contrastive grounding, iterative pruning, and cluster summarization to produce discriminative, expert-aligned features. We also release BrandGuide, to our knowledge the first dataset designed for systematic evaluation of expert alignment in automated feature engineering. This work positions feature discovery grounded in expert knowledge as a practical step toward deploying auditable ML systems in high-stakes domains.

Limitations. FEST currently addresses binary classification; extension to multi-class settings requires architectural modifications. Feature quality is bounded by the underlying LLM's capabilities and biases. While 86$\times$ cheaper than Felix, FEST requires multiple iterative LLM calls and therefore costs more than one-call baselines. See Appendix P for detailed discussion.

REFERENCES

Nikhil Abhyankar, Parshin Shojaee, and Chandan K Reddy. Llm-fe: Automated feature engineering for tabular data with llms as evolutionary optimizers. *arXiv preprint arXiv:2503.14434*, 2025. 2, 14, 15

Ayşegül Acar, Naci Büyükdağ, Burak Türten, Ersin Diker, and Gülsüm Çalışır. The role of brand identity, brand lifestyle congruence, and brand satisfaction on repurchase intention: a multi-group structural equation model. *Humanities and Social Sciences Communications*, 11(1):1–13, 2024. 6

Adobe. Adobe GenStudio for Performance Marketing: Brand Compliance. https://business.adobe.com/products/genstudio/performance-marketing/brand-compliance.html, 2024. Accessed: 2026-05-10. 8

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016. 15

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019. 15

João Eduardo Batista. Embedding domain-specific knowledge from llms into the feature engineering pipeline. *arXiv preprint arXiv:2503.21155*, 2025. 14

Canva. Canva Brand Kit. https://www.canva.com/en_in/pro/brand-kit/, 2024. Accessed: 2026-05-10. 8

Elena Cresci. Match.com ad criticised for suggesting red hair and freckles 'imperfections'. The Guardian, April 2016. URL https://www.theguardian.com/media/2016/apr/11/matchcom-ad-criticised-for-suggesting-red-hair-and-freckles-imperfections. 1

Scott Geng, Hamish Ivison, Chun-Liang Li, Maarten Sap, Jerry Li, Ranjay Krishna, and Pang Wei Koh. The delta learning hypothesis: Preference tuning on weak data can yield strong gains. *arXiv preprint arXiv:2507.06187*, 2025. 4, 16

Sungwon Han, Jinsung Yoon, Sercan O Arik, and Tomas Pfister. Large language models can automatically engineer features for few-shot tabular learning. *arXiv preprint arXiv:2404.09491*, 2024. 2, 14, 15

Franziska Horn, Robert Pack, and Michael Rieger. The autofeat python library for automated feature engineering and selection. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 111–120. Springer, 2019. 2, 3, 14

Robert Huben, Hoagy Cunningham, Logan Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *International Conference on Learning Representations*, volume 2024, pp. 7827–7845, 2024. 15

Daniel Kahneman. *Thinking, fast and slow*. macmillan, 2011. 14

Kevin Lane Keller. Brand synthesis: The multidimensionality of brand knowledge. *Journal of consumer research*, 29(4):595–600, 2003. 6

Varun Khurana, Yaman Singla, Jayakumar Subramanian, Changyou Chen, Rajiv Ratn Shah, Zhiqiang Xu, and Balaji Krishnamurthy. Measuring and improving engagement of text-to-image generation models. In *International Conference on Learning Representations*, volume 2025, pp. 38273–38304, 2025. 6, 38

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pp. 2668–2677. PMLR, 2018. 15

Jeonghyun Ko, Gyeongyun Park, Donghoon Lee, and Kyunam Lee. Ferg-llm: Feature engineering by reason generation large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 4211–4228, 2025. 14

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020. 14

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PmLR, 2023. 20

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. 1, 15

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594, 2023. 15

Simon Malberg, Edoardo Mosca, and Georg Groh. Felix: Automatic and interpretable feature engineering using llms. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 230–246. Springer, 2024. 2, 6, 7, 14, 15

Marq (formerly Lucidpress). Brand consistency—the competitive advantage and how to achieve it. Blog post, originally published 2018-2019, 2024. URL https://www.marq.com/blog/brand-consistency-competitive-advantage/. Survey of 400+ brand management experts. 6

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. *arXiv preprint arXiv:2304.06129*, 2023. 14

OpenAI. GPT-4o mini: Advancing cost-efficient intelligence. OpenAI Blog, 2024. URL https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Accessed: January 2026. 6, 7, 25

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1135–1144, 2016. 1, 15

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence*, 1(5):206–215, 2019. 1, 15

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019. 15

Mehran Sahami, Susan Dumais, David Heckerman, and Eric Horvitz. A bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop*, volume 62, pp. 98–105. Madison, Wisconsin, 1998. 14

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017. 1, 15

Edward Shortliffe. *Computer-based medical consultations: MYCIN*, volume 2. Elsevier, 2012. 14

Kimi Team, Tongtong Bai, Yifan Bai, et al. Kimi k2.5: Visual agentic intelligence, 2026. URL https://arxiv.org/abs/2602.02276. 19, 23

Elsbeth Turcan and Kathleen McKeown. Dreaddit: A reddit dataset for stress analysis in social media. In *Proceedings of the tenth international workshop on health text mining and information analysis (LOUHI 2019)*, pp. 97–107, 2019. 6, 38

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering. *arXiv preprint arXiv:2308.10248*, 2023. 15

Michael Weiss. An exploration of pattern mining with chatgpt. In *Proceedings of the 29th European Conference on Pattern Languages of Programs, People, and Practices*, pp. 1–11, 2024. 14

Tianping Zhang, Zheyu Aqa Zhang, Zhiyuan Fan, Haoyan Luo, Fengyuan Liu, Qian Liu, Wei Cao, and Li Jian. Openfe: Automated feature generation with expert-level performance. In *International Conference on Machine Learning*, pp. 41880–41901. PMLR, 2023. 2, 3, 14

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025. 4

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023. 2, 15

Yangqiaoyu Zhou, Haokun Liu, Tejes Srivastava, Hongyuan Mei, and Chenhao Tan. Hypothesis generation with large language models. In *Proceedings of the 1st Workshop on NLP for Science (NLP4Science)*, pp. 117–139, 2024. 6, 38

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, et al. Representation engineering: A top-down approach to ai transparency. *arXiv preprint arXiv:2310.01405*, 2023. 15

APPENDIX

A Related Work **14**

B Algorithm Pseudocode **16**

C Feature Encoding Details **16**

D Pairwise Comparison Motivation **16**

E Additional Results **16**

E.1 Direct End-to-End Text Classification 18

E.2 Full Brand-Text Classification Metrics 19

E.3 Cross-Model Replication with Kimi K2.5 19

E.4 Fine-Tuned Vision Classifier 20

F Ablations **21**

F.1 Semantic-only / Deterministic-only / SE+DE 21

F.2 Expert Refinement Disentanglement 21

G Feature Evolution Analysis **22**

H LLM-as-Judge Expert Coverage **23**

H.1 Protocol 23

H.2 Full Sensitivity Table 23

H.3 Second-Judge Cross-Check 23

H.4 Direct Judge-Human Agreement Study 23

I Why FEST Achieves Higher Coverage **24**

J Uncovered Guidelines Audit **24**

K Expert Human Study **24**

K.1 Motivation and Expert Recruitment 24

K.2 Protocol 24

K.3 Results 25

K.4 Limitations 25

L Runtime and Cost Analysis **25**

M Variance Analysis **26**

N Synthetic Off-Brand Robustness **26**

O Contamination Discussion **27**

P Limitations **27**

Q Future Work **28**

R Impact Statement **28**

S Hyperparameters **29**

S.1 Temperature Configuration 29

S.2 Other Hyperparameters 29

T Prompt Templates **29**

U BrandGuide Dataset Details **31**

U.1 Collection Pipeline 31

U.2 Quality Assurance 31

U.3 Extended Statistics 32

U.4 Research Directions 33

U.5 Dataset Examples 33

U.6 Licensing and Access 33

V AI Use Statement **33**

W Feature Examples **33**

X Licensing for Existing Assets **38**

A Related Work

1. Manual Feature Engineering. Early AI and expert systems relied on manually crafted, human-interpretable features and rules. Classic examples include the MYCIN medical diagnosis system, which encoded clinical knowledge as explicit features and logical rules (Shortliffe, 2012), and early spam filters that used hand-designed counts and lexical features to achieve practical performance (Sahami et al., 1998). Manual feature design remains common in domains that demand transparency (e.g., medicine, law and regulated industries), but it is costly, slow, and limited by human cognitive biases and domain coverage (Kahneman, 2011). As data modalities and problem complexity increased, the scalability limits of purely manual engineering motivated a broad literature on automated and semi-automated feature construction.

2. Classical Automated Feature Engineering. Classical automated feature engineering systems construct large candidate pools via predefined transformations and then select or prune useful features. Representative systems (e.g., AutoFeat and OpenFE) enumerate combinations, nonlinear transforms, or symbolic expressions and rely on statistical selection, regularization, or search procedures to find compact, predictive subsets (Horn et al., 2019; Zhang et al., 2023). These approaches reduce human effort and can produce strong tabular baselines, but they generally assume an existing set of input columns (tabular format), depend on handcrafted transformation templates, and do not natively operate on raw unstructured inputs (e.g., images or free text). Consequently, classical pipelines transfer poorly to multimodal raw-data settings and offer limited semantic control over the meaning of constructed features.

3. LLM-Based Feature Engineering. LLMs have recently been used as feature proposers in both structured and unstructured settings.

Structured data. LLMs have been applied to structured and tabular settings where feature transformations can be expressed programmatically. (Han et al., 2024) propose FeatLLM, an in-context prompting approach that elicits interpretable rule-style features from an LLM for few-shot tabular tasks; generated candidates are evaluated with simple downstream learners and selected when useful.(Abhyankar et al., 2025) frame feature engineering as program search in LLM-FE, combining evolutionary optimization with LLM generation so that the model proposes programmatic transforms and a performance signal guides evolution of the feature population. Batista (2025) study seeding classical pipelines with domain-aware LLM proposals to accelerate symbolic or evolutionary search, showing modest speedups and occasional accuracy gains from such knowledge injection. Ko et al. (2025) introduce FeRG-LLM, a fine-tuned LLM that uses iterative reasoning and local feedback to generate compact, locally interpretable features while remaining computationally efficient.

Unstructured data. Recent work has explored using large language models as pattern miners and feature proposers for unstructured inputs. (Weiss, 2024) documents how interactive prompting with ChatGPT can extract structured patterns and human-readable rules from heterogeneous data sources, illustrating a co-creative pattern-mining workflow that leverages LLM fluency for interpretability and hypothesis extraction. For text classification specifically, FELIX prompts LLMs to generate high-level textual descriptors or symbolic features from raw documents; these LLM-derived features feed lightweight classifiers and often outperform bag-of-words or raw embedding baselines while remaining human-interpretable (Malberg et al., 2024). Such systems demonstrate that LLMs can surface semantically meaningful, deployable features from unstructured inputs, but many are implemented as one-shot or human-in-the-loop procedures rather than fully automated iterative pipelines.

These methods demonstrate that LLMs can propose semantically meaningful features, but most operate in a single generation pass, assume tabular inputs, or lack iterative refinement with downstream validation signals.

4. Concept Bottleneck Models and Concept Extraction. Concept Bottleneck Models (CBMs) pursue a complementary goal: mapping raw inputs to human-interpretable intermediate concepts before prediction (Koh et al., 2020). CBMs require predefined concept sets with labeled annotations, limiting scalability. Label-Free CBMs address this by using foundation models to automatically generate concept sets without labeled concept data, scaling to ImageNet (Oikarinen et al., 2023). TCAV provides a post-hoc alternative, using concept activation vectors to quantify concept importance

in trained networks without modifying them (Kim et al., 2018). Concept extraction methods such as probing classifiers on hidden states (Alain & Bengio, 2016), steering directions (Turner et al., 2023; Zou et al., 2023), and sparse-autoencoder feature dictionaries (Huben et al., 2024) likewise operate on internal representations and, like TCAV, require access to a target model's hidden activations. Unlike these methods, FEST requires no access to model internals, no manual annotations, and no prior concept pool. FEST constructs the feature space directly from raw text and image inputs via contrastive prompting, and returns natural-language criteria and executable functions that can be evaluated directly on new inputs. Furthermore, FEST operationalizes qualitative expert guidelines (such as "use high-quality images") into precise feature definitions computed from raw data.

5. Inherent Interpretability vs. Post-hoc Explanation. Rudin (2019) argues that high-stakes domains should use inherently interpretable models rather than explaining black boxes post hoc, since post-hoc methods (LIME, SHAP, GradCAM) produce approximate attributions that may not faithfully reflect model reasoning (Ribeiro et al., 2016; Lundberg & Lee, 2017; Selvaraju et al., 2017). FEST aligns with this principle: its features are explicit, human-verifiable predicates (e.g., "uses inclusive language," "exclamation count") that practitioners can audit before deployment, and its decision tree paths provide transparent classification logic. This contrasts with deep embedding approaches where interpretability requires auxiliary explanation tools.

6. Iterative Self-Refinement. FEST's generate-evaluate-refine loop connects to the broader paradigm of iterative self-improvement in LLM systems. Self-Refine (Madaan et al., 2023) demonstrated that LLMs can generate output, critique it, and iteratively improve, achieving 20% average gains across diverse tasks. FEST applies an analogous loop to feature engineering: LLMs generate candidate features, decision trees evaluate their discriminative value, and weak features are pruned while new candidates are proposed in subsequent iterations. The key difference is that FEST's refinement signal comes from an external classifier (decision tree importance) rather than LLM self-critique, grounding the evolution in empirical task performance.

7. LLM-as-Judge Evaluation. Zheng et al. (2023) established that strong LLMs can approximate human judgment with $>80\%$ agreement, enabling scalable evaluation of open-ended outputs. FEST adopts this paradigm for expert coverage evaluation: GPT-4o judges whether discovered features capture the same brand dimensions as expert guidelines, providing interpretable 0–10 scores that eliminate reliance on arbitrary embedding similarity thresholds.

8. Comparison with FEST: While prior work demonstrates the promise of LLMs for feature discovery across both unstructured and structured settings, FEST departs from these approaches along several dimensions. First, FEST targets raw multimodal observational data (text, images, and tabular inputs) by prompting LLMs to generate *interpretable* feature candidates directly from example pairs rather than requiring an initial tabular feature bank. Second, rather than performing a single stage of proposal and selection, FEST implements a *self-evolving* generate, deduplicate, validate loop: (i) LLMs propose diverse semantic and deterministic features from pairwise comparisons, (ii) semantic embeddings and clustering compress paraphrastic or duplicate proposals, and (iii) decision trees evaluate, provide feature-importance signals, and guide iterative pruning and feature-bank evolution. This closed-loop design (including the use of probabilistic LLM feature inference scores and a feature importance history for robust pruning) enables continuous refinement and avoids the redundancy and spuriousness typical of one-shot generators. Third, FEST explicitly prioritizes semantic control and human interpretability: deduplicated feature summaries and tree decision paths provide transparent, human-readable logic that practitioners can inspect and adjust, contrasting with approaches that rely primarily on opaque performance signals or implicit model internals (Han et al., 2024; Abhyankar et al., 2025; Malberg et al., 2024). Finally, FEST's pairwise comparison strategy and tree-based validation make it robust to common-attribute noise and permit recovery of compositional logical relationships (as validated on controlled benchmarks), positioning FEST as a more general, controllable, and interpretable automated feature engineering methodology for both raw and structured data.

9. Connection to Robust Optimization. FEST shares a conceptual affinity with Invariant Risk Minimization (IRM) (Arjovsky et al., 2019) and Group DRO (Sagawa et al., 2019): both seek features stable across environments rather than spuriously correlated with labels. FEST's iterative pruning of

low-importance features echoes invariance-seeking, as features that do not generalize across batches are progressively eliminated. The key distinction is that IRM and Group DRO operate on a *fixed* feature space with explicit environment labels, whereas FEST *constructs* the feature space from raw unstructured data. FEST thus addresses a problem upstream of robust optimization: discovering interpretable features in the first place. These are complementary; robust optimization could be applied atop FEST-discovered features for further out-of-distribution gains.

B Algorithm Pseudocode

Algorithm 1 presents the complete FEST pseudocode.

C Feature Encoding Details

This section provides the full mathematical formulation of feature encoding summarized in §3.2.3.

Semantic Features Encoding. For each individual sample x and each semantic feature $f_k \in F_{SE}$, we prompt the LLM to evaluate whether f_k is present or absent. Rather than using binary responses, we extract richer signal from LLM uncertainty: we obtain output probabilities $p_\theta(y = 1 \mid f_k, x)$ and $p_\theta(y = 0 \mid f_k, x)$ for tokens "1" (present) and "0" (absent), then compute normalized confidence:

$$g_{SE}(f_k, x) \;=\; \frac{p_\theta(y = 1 \mid f_k, x)}{p_\theta(y = 1 \mid f_k, x) + p_\theta(y = 0 \mid f_k, x)} \tag{2}$$

where p_θ denotes the LLM parameterized by θ, and $y \in \{0, 1\}$ represents the prediction. This confidence score in $[0, 1]$ captures LLM certainty about feature presence, enabling more nuanced encoding than binary labels. We construct the semantic feature matrix $\mathbf{X}_{SE} \in \mathbb{R}^{2|B| \times |F_{SE}|}$ where each element $X_{SE,ij} = g_{SE}(f_j, x_i)$.

Deterministic Features Encoding. For each sample x and each deterministic feature $f_k \in F_{DE}$, we execute the associated Python function: $g_{DE}(f_k, x) = \texttt{function}_k(x)$, returning numeric values. We construct the deterministic feature matrix $\mathbf{X}_{DE} \in \mathbb{R}^{2|B| \times |F_{DE}|}$ where each element $X_{DE,ij} = g_{DE}(f_j, x_i)$.

Combined Feature Matrix. We concatenate the semantic and deterministic feature matrices to obtain the final feature representation: $\mathbf{X} = [\mathbf{X}_{SE} \mid \mathbf{X}_{DE}] \in \mathbb{R}^{2|B| \times (|F_{SE}| + |F_{DE}|)}$. The corresponding label vector is $\mathbf{y} \in \{0, 1\}^{2|B|}$ where $y_i = 1$ for positive class samples and $y_i = 0$ for negative class samples.

D Pairwise Comparison Motivation

Absolute assessment of individual samples suffers from two fundamental limitations: (1) inability to distinguish universally present attributes from discriminative features, and (2) dependence on subjective absolute thresholds. For example, in news headline analysis, absolute assessment might identify "contains numbers" as relevant without recognizing that numbers appear equally in both successful and unsuccessful headlines. Pairwise comparison identifies features that actually differentiate: successful headlines use specific question formats while unsuccessful ones use generic statements. This design is supported by recent work (Geng et al., 2025) showing that the quality delta between samples in a pair provides a richer learning signal than individual samples. FEST's convergence and pruning thresholds ($\tau_{accuracy}$, $\tau_{importance}$) are algorithmic flow-control parameters, distinct from the semantic feature-definition thresholds criticized in absolute assessment approaches.

E Additional Results

- Brand classification (text) results in Table 2
- Brand classification (image) results in Table 3

Algorithm 1 FEST: Feature Engineering with Self-evolving Trees

Input: Dataset D with binary labels, hyperparameters $\tau_{accuracy}$, $\tau_{importance}$, K (batch size)
Output: Feature bank F and trained decision tree $\mathcal{T}$
Initialize:
$F \leftarrow \emptyset$ {Global feature bank (or initialize with expert features if available)}
$\mathcal{P} \leftarrow$ ConstructPairs(D) {Create comparison pairs (x^+, x^-)}
$\mathcal{P}_{train}, \mathcal{P}_{val}, \mathcal{P}_{test} \leftarrow$ Split($\mathcal{P}$) {Train/validation/test split}
$H_{importance} \leftarrow \emptyset$ {Feature importance history}
$t \leftarrow 0$ {Iteration counter}

while validation accuracy $< \tau_{accuracy}$ and pairs remain **do**
  $t \leftarrow t + 1$
  $B \leftarrow$ NextBatch($\mathcal{P}_{train}$, K) {Get batch of $|B|$ pairs}

  // Stage 1: Dual-Stream Feature Generation
  /* Semantic Feature Generation */
  $F_{SE}^{(t)} \leftarrow \emptyset$
  for each pair $(x_i^+, x_i^-) \in B$ **do**
    for each prompt template $p \in \{1, \ldots, M\}$ **do**
      $f \leftarrow$ LLM.GenerateSEFeature(x_i^+, x_i^-, template p) {Pairwise comparison}
      $F_{SE}^{(t)} \leftarrow F_{SE}^{(t)} \cup \{f\}$
    end for
  end for

  /* Deterministic Feature Generation */
  $B_{sample} \leftarrow$ Sample(B, size=5) {Sample few pairs for task-level generation}
  $F_{DE}^{(t)} \leftarrow$ LLM.GenerateDEFeatures(B_{sample}, task_desc, F) {Task-level, returns Python functions}
  $F_{DE}^{(t)} \leftarrow$ ValidateAndRefine($F_{DE}^{(t)}$) {Execute in sandbox, fix errors}

  // Stage 2: Semantic Deduplication and Bank Merging
  $E_{SE} \leftarrow$ ConditionalEmbedding($F_{SE}^{(t)}$, task_desc) {Domain-aware embeddings}
  $C_{clusters} \leftarrow$ KMeansClustering(E_{SE}) {Cluster semantically similar SE features}
  $\bar{F}_{SE}^{(t)} \leftarrow$ LLM.SummarizeClusters($C_{clusters}$) {One representative per cluster}
  $\bar{F}_{SE}^{(t)} \leftarrow$ SemanticMerge($\bar{F}_{SE}^{(t)}$, F) {Remove duplicates with existing bank}
  $F^{(t)} \leftarrow F \cup \bar{F}_{SE}^{(t)} \cup F_{DE}^{(t)}$ {Current iteration's complete feature set}

  // Stage 3: Feature Inference and Encoding
  Split pairs in B into individual samples: $S \leftarrow \{x_1^+, x_1^-, \ldots, x_{|B|}^+, x_{|B|}^-\}$ with $|S| = 2|B|$

  for each sample $x_i \in S$ and SE feature $f_j \in F_{SE}$ (all semantic features in $F^{(t)}$) **do**
    $p_1, p_0 \leftarrow$ LLM.GetProbs(f_j, x_i) {Probabilities for tokens "1" and "0"}
    $X_{SE,ij} \leftarrow g_{SE}(f_j, x_i) = \frac{p_1}{p_1 + p_0}$ {Normalized confidence}
  end for

  for each sample $x_i \in S$ and DE feature $f_j \in F_{DE}$ (all deterministic features in $F^{(t)}$) **do**
    $X_{DE,ij} \leftarrow g_{DE}(f_j, x_i) = \texttt{function}_j(x_i)$ {Execute Python function}
  end for

  $\mathbf{X} \leftarrow [\mathbf{X}_{SE} \mid \mathbf{X}_{DE}]$ {Concatenate: $\mathbf{X} \in \mathbb{R}^{2|B| \times (|F_{SE}| + |F_{DE}|)}$}
  $\mathbf{y} \leftarrow [1, 0, \ldots, 1, 0]^\top$ {Labels: 1 for x^+, 0 for x^-, $\mathbf{y} \in \{0, 1\}^{2|B|}$}

  // Stage 4: Tree Training and Evolution
  $\mathcal{T} \leftarrow$ DecisionTree.Train($\mathbf{X}, \mathbf{y}$) {Train decision tree}
  $acc_{val} \leftarrow$ Evaluate($\mathcal{T}$, $\mathcal{P}_{val}$, $F^{(t)}$) {Validation accuracy}
  if $acc_{val} \geq \tau_{accuracy}$ **then**
    break {Convergence achieved}
  end if

  $\mathbf{I}^{(t)} \leftarrow \mathcal{T}$.GetFeatureImportance() {Extract importance scores}
  $H_{importance} \leftarrow$ UpdateHistory($H_{importance}$, $\mathbf{I}^{(t)}$, $F^{(t)}$)
  $F^{(t)'} \leftarrow$ Prune($F^{(t)}$, $H_{importance}$, $\tau_{importance}$) {Remove low-importance features}
  $F \leftarrow F^{(t)'}$ {Update global bank for next iteration}
end while

return $F, \mathcal{T}$

Table 2: Brand-wise classification accuracy (%) for text-based brand validation across 5 brands (Emirates, Adobe, Porsche, Louis Vuitton, Pizza Hut) and 5 classifiers (DT, LR, RF, MLP, XGB). Main text Table 1 reports averages across brands. Bold indicates best per brand-classifier combination.

Classifier	Feature Generator	Brand				
		Emirates	Adobe	Porsche	Louis Vuitton	Pizza Hut
DT	Zero-Shot LLM	69.60	72.80	60.40	85.60	78.05
	Few-Shot LLM	69.20	81.73	68.93	**86.53**	75.20
	Felix	67.60	69.60	63.20	80.00	**81.09**
	FEST (Ours)	**73.60**	**88.40**	**77.20**	83.60	78.05
LR	Zero-Shot LLM	64.40	66.40	68.00	78.40	70.00
	Few-Shot LLM	61.06	82.80	64.00	87.06	69.91
	Felix	72.00	77.20	66.80	80.40	82.92
	FEST (Ours)	**73.60**	**86.00**	**76.80**	**88.00**	**84.14**
RF	Zero-Shot LLM	80.40	76.40	71.20	88.00	85.37
	Few-Shot LLM	77.99	**88.80**	75.73	89.60	83.73
	Felix	76.80	81.60	71.60	84.80	85.36
	FEST (Ours)	**82.80**	88.00	**78.80**	**90.00**	**85.97**
MLP	Zero-Shot LLM	69.20	76.80	69.60	86.80	77.43
	Few-Shot LLM	72.00	82.53	67.33	87.73	74.18
	Felix	72.40	77.20	**74.00**	84.80	**82.92**
	FEST (Ours)	**78.80**	**85.20**	72.00	**90.80**	81.09
XGB	Zero-Shot LLM	78.40	75.20	70.80	87.60	83.53
	Few-Shot LLM	77.86	87.46	73.30	89.40	**87.19**
	Felix	78.00	80.80	74.00	84.00	86.58
	FEST (Ours)	**83.60**	**88.00**	**78.40**	**90.00**	82.92

Table 3: Brand-wise classification accuracy (%) for image-based brand validation across 5 brands (Emirates, Adobe, Porsche, Louis Vuitton, Pizza Hut) and 5 classifiers (DT, LR, RF, MLP, XGB). Main text Table 1 reports averages across brands. Bold indicates best per brand-classifier combination.

Classifier	Feature Generator	Brand				
		Emirates	Adobe	Porsche	Louis Vuitton	Pizza Hut
DT	Zero-Shot LLM	60.49	67.33	58.40	62.70	77.16
	Few-Shot LLM	66.27	75.50	64.80	66.40	83.53
	Felix	60.40	77.20	58.00	58.80	84.70
	FEST (Ours)	**76.80**	**82.40**	**71.20**	**72.80**	**88.41**
LR	Zero-Shot LLM	62.96	74.59	67.20	70.08	77.16
	Few-Shot LLM	62.65	78.31	68.40	72.40	83.53
	Felix	62.40	**82.40**	60.80	66.80	79.26
	FEST (Ours)	**75.60**	81.20	**68.80**	**72.80**	**89.02**
RF	Zero-Shot LLM	64.60	79.43	70.80	74.18	83.33
	Few-Shot LLM	75.10	83.94	74.80	75.60	84.14
	Felix	59.60	83.60	59.60	64.80	85.30
	FEST (Ours)	**81.60**	**85.60**	**70.40**	**78.70**	**91.46**
MLP	Zero-Shot LLM	67.90	74.19	66.40	68.85	74.07
	Few-Shot LLM	70.28	73.09	**70.40**	70.80	82.90
	Felix	63.20	**80.80**	57.60	58.40	84.14
	FEST (Ours)	**78.00**	79.20	64.80	**71.20**	**89.63**
XGB	Zero-Shot LLM	65.02	77.01	68.40	72.54	82.17
	Few-Shot LLM	70.28	80.32	**74.80**	74.80	84.14
	Felix	59.20	84.00	60.80	67.20	87.19
	FEST (Ours)	**82.00**	**84.40**	71.20	**78.00**	**90.85**

E.1 Direct End-to-End Text Classification

The Zero-Shot LLM and Few-Shot LLM results above in Table 2 evaluate generated feature banks with downstream classifiers. Direct end-to-end LLM classification instead predicts labels without extracting features, producing a single opaque decision with no inspectable feature basis. Table 4 reports this comparison on the same five brand-text tasks. Zero-shot direct classification performs near chance on this binary setup. Few-shot direct classification improves performance but remains below FEST with RF (72.9% versus 85.1%).

Table 4: Direct end-to-end LLM classification accuracy (%) on the five brand-text tasks, compared with FEST features and an RF classifier. Direct baselines predict labels without producing an inspectable feature bank.

Brand	Zero-shot direct	Few-shot direct	FEST (RF)
Emirates	43.2	58.4	**82.8**
Adobe	50.8	67.6	**88.0**
Porsche	51.6	70.4	**78.8**
Louis Vuitton	52.4	86.0	**90.0**
Pizza Hut	58.5	82.3	**86.0**
Mean	51.3	72.9	**85.1**

E.2 Full Brand-Text Classification Metrics

Table 5 reports brand-averaged Accuracy, Recall, Precision, and F1 on the five text-classification tasks under GPT-4o-mini. FEST leads on every classifier across Accuracy, Precision, and F1, and leads Recall for four of five classifiers; Few-Shot LLM has the highest RF Recall (83.7% versus 82.4%).

Table 5: Brand-averaged text-classification metrics (%) using GPT-4o-mini. Bold marks the best result for each classifier and metric.

Classifier	Method	Accuracy	Recall	Precision	F1
DT	Zero-Shot LLM	73.3	75.6	72.2	73.8
	Few-Shot LLM	73.0	76.8	75.8	76.3
	Felix	75.9	75.5	71.1	73.2
	FEST (Ours)	**81.7**	**79.2**	**80.8**	**79.8**
MLP	Zero-Shot LLM	76.0	78.3	75.0	76.6
	Few-Shot LLM	76.8	76.4	75.4	75.8
	Felix	78.3	77.8	78.3	78.0
	FEST (Ours)	**81.6**	**83.4**	**82.9**	**83.1**
LR	Zero-Shot LLM	69.5	70.7	69.2	69.9
	Few-Shot LLM	73.0	73.0	71.6	72.0
	Felix	75.7	76.3	75.4	75.8
	FEST (Ours)	**81.6**	**80.9**	**82.4**	**81.6**
RF	Zero-Shot LLM	80.3	81.5	79.3	80.3
	Few-Shot LLM	83.2	**83.7**	82.7	83.0
	Felix	80.0	80.9	79.4	80.1
	FEST (Ours)	**85.1**	82.4	**86.7**	**84.4**
XGB	Zero-Shot LLM	79.1	81.7	77.5	79.5
	Few-Shot LLM	83.0	82.0	82.0	82.0
	Felix	80.7	82.0	79.8	80.9
	FEST (Ours)	**84.6**	**82.4**	**85.8**	**85.0**

E.3 Cross-Model Replication with Kimi K2.5

To evaluate robustness across model families, we executed the complete FEST pipeline using the open-weight reasoning model Kimi K2.5 (thinking) (Team et al., 2026) in place of GPT-4o-mini across all five text-brand tasks with no algorithmic changes. FEST leads Felix and all feature-generation baselines across Accuracy, Precision, and F1 over all five downstream classifiers under Kimi K2.5 (DT 85.1%, MLP 85.1%, LR 87.6%, RF 88.3%, XGB 85.6%), corroborating the method's transfer across model families and open-weight reasoning architectures (Table 6).

Table 6: Brand-averaged text-classification metrics (%) using Kimi K2.5 (thinking). Bold marks the best result for each classifier and metric.

Classifier	Method	Accuracy	Recall	Precision	F1
DT	Zero-Shot LLM	67.9	71.3	69.6	68.7
	Few-Shot LLM	75.0	78.8	74.3	76.1
	Felix	80.0	**86.2**	76.8	81.2
	FEST (Ours)	**85.1**	85.7	**84.9**	**85.2**
MLP	Zero-Shot LLM	68.1	71.2	70.1	69.0
	Few-Shot LLM	75.2	77.4	75.4	75.9
	Felix	83.1	**88.5**	80.1	84.0
	FEST (Ours)	**85.1**	84.3	**85.9**	**85.1**
LR	Zero-Shot LLM	66.9	68.4	70.1	67.4
	Few-Shot LLM	74.0	75.0	74.3	74.5
	Felix	83.0	78.4	86.2	82.0
	FEST (Ours)	**87.6**	**84.6**	**90.1**	**87.2**
RF	Zero-Shot LLM	68.2	72.7	69.6	69.3
	Few-Shot LLM	75.2	78.1	75.0	76.0
	Felix	85.1	**89.7**	82.8	85.8
	FEST (Ours)	**88.3**	86.1	**90.0**	**87.9**
XGB	Zero-Shot LLM	68.5	71.7	70.3	69.3
	Few-Shot LLM	75.3	78.3	75.1	76.2
	Felix	82.7	**89.1**	80.3	84.3
	FEST (Ours)	**85.6**	84.5	**86.3**	**85.4**

E.4 Fine-Tuned Vision Classifier

FEST and standard vision classifiers answer different questions. FEST produces interpretable, expert-auditable features; a vision classifier produces an opaque label with no inspectable feature basis for a practitioner.

To evaluate raw performance context, we benchmarked a fine-tuned BLIP-2 classifier (Q-Former plus a linear head) (Li et al., 2023) on the same five brand image tasks. BLIP-2 reaches 94% to 99% test accuracy (Emirates 95.2%, Adobe 94.0%, Porsche 95.2%, Louis Vuitton 94.8%, Pizza Hut 99.4%), outperforming FEST's approximately 81.6% mean accuracy (Table 7).

Table 7: Test accuracy (%) on the five brand-image tasks. BLIP-2 uses a fine-tuned Q-Former plus a linear classifier head; FEST uses its RF classifier. Bold marks the best result per brand and on average.

Method	Emirates	Adobe	Porsche	Louis Vuitton	Pizza Hut	Mean
FEST (RF)	81.6	85.6	70.4	78.7	91.5	81.6
BLIP-2	**95.2**	**94.0**	**95.2**	**94.8**	**99.4**	**95.7**

We extracted BLIP-2 Q-Former attention maps (early, mid, and late layers) and mapped them into concepts using GPT-4o-mini to analyze the underlying feature basis:

1. *Executable vs. Opaque*: Many of FEST's image features are executable Python functions with learned numerical thresholds (e.g., Porsche edge density via `cv2.Canny()` for sharp automotive lines and widescreen aspect ratio; Pizza Hut luminance-weighted contrast and text-area ratio; Louis Vuitton color-diversity count and subject-background contrast). BLIP-2 attention maps yield only prose descriptions with no computable criteria to audit, threshold, or enforce.
2. *Directive and Brand-Specific vs. Generic Saliency*: Described without brand context, BLIP-2 attention maps reduce to generic visual primitives (faces, high-contrast regions, text bounding boxes, object edges) that recur near-verbatim across all brands. FEST's features provide directive, brand-specific criteria a designer can act on (e.g., Pizza Hut "vibrant warm, red-dominant palette evoking appetite"; Porsche "avoid dark, muted, earthy palettes, use vibrant high-key lighting").

FEST trades approximately 14 percentage points of raw accuracy for a feature basis that is computable, auditable, brand-specific, and actionable for a practitioner.

F ABLATIONS

F.1 SEMANTIC-ONLY / DETERMINISTIC-ONLY / SE+DE

We run FEST in three configurations across 7 tasks (5 brand text classification tasks + content authenticity detection + stress detection) to isolate the contribution of each feature stream.

Table 8: Dual-stream ablation (accuracy %) with Decision Tree and Random Forest classifiers across 7 tasks (5 brand text classification + content authenticity + stress detection). SE+DE (full FEST) outperforms both single-stream variants in 11 of 14 task-classifier combinations. The three exceptions (Louis Vuitton DT, Adobe RF, Porsche RF) show SE-only marginally outperforming by 0.4–3.2pp, indicating DE features add value in most settings and do not hurt in the rest. DE-only is consistently the weakest, confirming semantic features form the core signal while deterministic features add measurable precision. Bold indicates best per task-classifier.

	Decision Tree			Random Forest		
Task	**DE**	**SE**	**SE+DE**	**DE**	**SE**	**SE+DE**
Emirates	67.2	72.8	**73.6**	70.8	76.4	**82.8**
Adobe	68.0	83.2	**88.4**	74.8	**90.4**	88.0
Porsche	72.8	71.2	**77.2**	74.0	**79.2**	78.8
Louis Vuitton	79.2	**86.8**	83.6	84.8	89.8	**90.0**
Pizza Hut	74.8	72.0	**78.1**	79.2	78.0	**86.0**
Content Authenticity	63.2	87.6	**91.2**	91.2	94.4	**97.2**
Stress Detection	58.4	72.0	**78.0**	63.2	78.8	**83.6**

SE+DE wins in 11 of 14 DT+RF combinations. The three exceptions (Louis Vuitton DT, Adobe RF, Porsche RF) show SE-only marginally outperforming by 0.4–3.2pp, indicating that DE features add value in most settings and do not hurt in the rest. DE-only is always the weakest, confirming that semantic features form the core signal while deterministic features contribute complementary measurable discriminative power.

F.2 EXPERT REFINEMENT DISENTANGLEMENT

To disentangle the contributions of refinement and augmentation in expert feature refinement, we evaluate three conditions for brand text classification:

- **A (Expert alone)**: Expert-designed features used as-is.
- **B (Expert Refined)**: FEST features filtered to those semantically similar to expert features (cosine ≥ 0.7), capturing refinement without augmentation.
- **C (Expert+FEST)**: Full FEST output including augmented features.

Table 9 presents the full per-classifier breakdown. B>A in most cases demonstrates that refinement alone improves over static expert features. C>B in all cases demonstrates that augmentation provides further independent gains. The one exception (LR Adobe: B=78.4 $<$ A=80.4) may reflect LR sensitivity to feature set size changes, but C still wins by +8.0pp.

Table 9: Expert feature refinement accuracy (%) for text brand classification across three brands (Adobe, LG, Porsche) and four classifiers. Column shading encodes the performance tier: **A (bronze)** uses expert-authored features as-is; **B (silver)** replaces them with FEST-discovered features semantically aligned to expert guidelines (LLM-judge score ≥ 7), isolating the effect of refinement without augmentation; **C (gold)** adds all remaining FEST features (Expert+FEST), measuring the further benefit of augmentation. B>A in most cases confirms that FEST operationalizes ambiguous guidelines into more discriminative definitions; C>B in all cases confirms that augmented features provide complementary gains. Gain (C−A) is reported in percentage points. Avg rows (bold) average over DT, LR, RF, and LLM classifiers.

Clf.	Brand	A (Expert)	B (Refined)	C (Exp+FEST)	Gain (C−A)
DT	Adobe	78.40	87.20	88.40	+10.00
	LG	69.60	74.15	75.42	+5.82
	Porsche	82.00	84.40	88.80	+6.80
LR	Adobe	80.40	78.80	90.40	+10.00
	LG	69.20	75.42	83.47	+14.27
	Porsche	77.60	86.40	90.00	+12.40
RF	Adobe	84.40	91.20	92.00	+7.60
	LG	79.60	82.62	86.44	+6.84
	Porsche	85.20	86.40	88.40	+3.20
LLM	Adobe	54.40	56.00	73.60	+19.20
	LG	53.38	54.66	58.05	+4.67
	Porsche	71.20	73.20	73.60	+2.40
Avg	Adobe	**74.40**	**78.30**	**86.10**	**+11.70**
	LG	**67.94**	**71.71**	**75.84**	**+7.90**
	Porsche	**79.00**	**82.60**	**85.20**	**+6.20**

G Feature Evolution Analysis

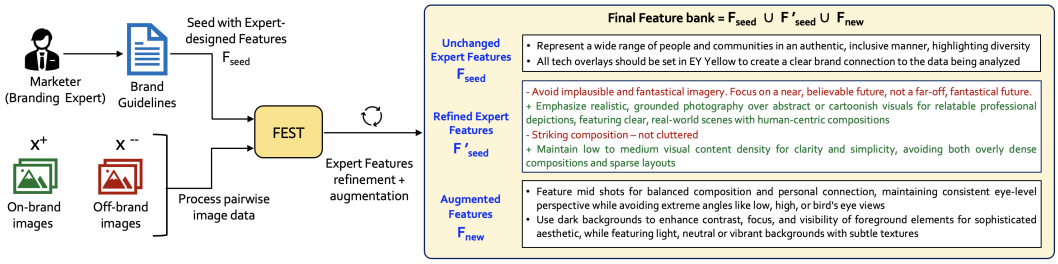


Figure 4: FEST refines and augments expert features (EY imagery). From seeds F_{seed}, FEST produces unchanged, refined (F'_{seed}), and augmented (F_{new}) features.

Figure 4 visualizes the feature bank evolution for EY brand image classification. The final features fall into three groups:

1. *Unchanged Expert Features* (F_{seed}): features retained verbatim because they are already precise and actionable (e.g., "All tech overlays should be set in EY Yellow to create a clear brand connection").
2. *Refined Expert Features* (F'_{seed}): ambiguous guidelines transformed into precise operational definitions. For example, "Avoid implausible and fantastical imagery. Avoid unoriginal or unsophisticated imagery…" was refined to "Emphasize realistic, grounded photography over abstract or cartoonish visuals for relatable professional depictions, featuring clear, real-world scenes with human-centric compositions." Similarly, Porsche's text guideline "Narrative told through the voice of people and their personal stories" was operationalized as "Emphasize emotional storytelling and personal connections over technical details and promotional language." These refinements convert implicit expert knowledge into explicit, measurable criteria that LLMs can reliably apply.

3. *Augmented Features* (F_{new}): novel features discovered by FEST that extend beyond documented guidelines, surfacing implicit domain knowledge (e.g., "Feature mid shots for balanced composition and personal connection, maintaining consistent eye-level perspective while avoiding extreme angles" for EY images).

H LLM-AS-JUDGE EXPERT COVERAGE

H.1 PROTOCOL

For each brand (Adobe, LG, Porsche), we evaluate the top-20 features from FEST and Felix against all expert-designed brand voice characteristics. GPT-4o serves as the judge, rating each (expert guideline, discovered feature) pair on a 0–10 semantic alignment scale. A guideline is "covered" if any discovered feature scores at or above the threshold. This removes dependence on embedding similarity cutoffs and provides interpretable, judgment-based quality scores.

H.2 FULL SENSITIVITY TABLE

Table 10: GPT-4o-judged coverage (%) of expert brand voice characteristics at varying semantic alignment thresholds. Under this judge, FEST coverage is perfectly stable from threshold 5 through 7 (zero change for all 3 brands), confirming covered guidelines are strong matches (7+/10). Felix collapses at each step: 0% for Porsche at $\geq$7 vs. FEST's 60%.

Brand	Method	$\geq$5	$\geq$6	$\geq$7	$\geq$8
Adobe	FEST (Ours)	**80.0**	**80.0**	**80.0**	**40.0**
	Felix	73.3	60.0	53.3	40.0
LG	FEST (Ours)	69.2	**69.2**	**69.2**	**61.5**
	Felix	**84.6**	69.2	53.8	38.5
Porsche	FEST (Ours)	**60.0**	**60.0**	**60.0**	**40.0**
	Felix	60.0	10.0	**0.0**	**0.0**

Notably, at the lenient threshold $\geq$5, Felix actually leads on LG (84.6% vs. 69.2%), but these are weak matches that collapse at stricter thresholds, while all of FEST's matches persist through $\geq$7.

H.3 SECOND-JUDGE CROSS-CHECK

We repeated the full coverage evaluation across all 1,520 feature-guideline pairs, covering the complete set across Adobe, LG, and Porsche with zero sampling, using a second independent judge model, Kimi K2.5 (Team et al., 2026). Continuous 0–10 scores between GPT-4o and Kimi K2.5 correlate strongly (Pearson $r = 0.87$; Spearman $\rho = 0.87$). FEST leads Felix under both judges at every brand and across every coverage threshold (5 through 8). Furthermore, we found zero pairs where one judge scored a strong match (≥ 7) and the other scored unrelated (≤ 2), confirming that the judges do not disagree arbitrarily and that the coverage gap between FEST and Felix is not an artifact of GPT-4o specifically.

H.4 DIRECT JUDGE-HUMAN AGREEMENT STUDY

We conducted a blinded human agreement study across a pool of 100 feature-guideline pairs stratified by GPT-4o score for representative sampling across the full score distribution, covering both FEST and Felix. Raters saw two blinded, neutrally labeled statements, with no indication of which was the guideline and which was the feature and no LLM score shown, and rated semantic similarity from 0 to 10 using the identical rubric and scale as the LLM judge. In each of 19 independent rating sessions, a human rater evaluated 20 randomly sampled pairs from the pool, yielding 380 total ratings. Human ratings correlate with GPT-4o at Pearson $r = 0.64$ and Spearman $\rho = 0.64$, a moderate-to-strong agreement for this kind of semantic-similarity task.

I WHY FEST ACHIEVES HIGHER COVERAGE

Both FEST and Felix use contrastive samples and the same backbone LLM (GPT-4o-mini), so the coverage gap stems from what happens *after* feature proposals. Three architectural differences drive FEST's superior alignment:

1. *Cluster summarization*: Felix selects the feature closest to each cluster centroid, preserving idiosyncratic LLM phrasing. FEST summarizes clusters into canonical descriptions, producing language that converges toward how experts naturally express concepts.
2. *Iterative pruning*: Felix is one-shot, so all features survive regardless of quality. FEST's tree-guided importance scores identify and prune generic features (e.g., "uses positive language") that lack discriminative power. Since expert features are inherently discriminative (experts select what distinguishes classes), iterative selection for discriminative power naturally converges toward expert-aligned features.
3. *Multi-stage language refinement*: FEST edits feature language at discovery, during cluster summarization, and during bank merging across iterations. Each consolidation step distills toward more precise, canonical formulations, while Felix's single-pass centroid-pick cannot improve feature framing based on feedback from data.

J UNCOVERED GUIDELINES AUDIT

An audit of all expert guidelines not covered by FEST at threshold ≥ 7 reveals they fall into categories that are structurally unobservable from social media post text:

- **Organizational policies**: "Zero-tolerance for hate speech," "DEI copy-editing policy." These are internal editorial standards not manifested in the style of published posts.
- **Abstract brand philosophy**: High-level brand values (e.g., "building a better working world") that do not translate to measurable textual style features.
- **Platform meta-guidelines**: "Design messages to fit the platform," "Adapt content for each channel." These require cross-platform context unavailable from individual posts.

These guidelines receive LLM-judge scores of 4–6/10 (semantically proximate but not strongly covered), indicating that FEST discovers related features in the same semantic neighborhood but cannot fully capture structurally unmeasurable dimensions. This gap is not specific to FEST; no text-only feature generator (FEST, Felix, or zero-shot LLM) could measure these from post content alone.

K EXPERT HUMAN STUDY

K.1 MOTIVATION AND EXPERT RECRUITMENT

Task accuracy measures discriminative power of features but not whether practitioners find features meaningful in practice. We complement the automatic evaluation with a structured expert evaluation study. One domain practitioner per brand with direct professional experience in brand marketing or content strategy rated the FEST feature bank for their assigned brand. Brand-guideline evaluation requires brand-specific institutional knowledge that cannot be crowd-sourced: a large panel of non-specialist annotators would produce low-signal ratings, while a single qualified practitioner provides authoritative practitioner acceptance. This follows the specialist-rater paradigm used in expert-evaluation studies where the relevant expertise is scarce and non-commoditizable.

K.2 PROTOCOL

For each brand, FEST produces a final feature bank containing both refined expert features and newly discovered features. They are presented to the expert as a *single blinded pool*: the expert does not know which features are refined from expert guidelines and which are newly discovered by FEST. For each feature, the interface displayed up to 10 content samples receiving the highest attribution

score for that feature, providing concrete evidence of how the feature manifests in real brand content. Top 15 features were rated per brand across 2 brands (Zomato images, Adobe images). Expert details are anonymized. Experts rated each feature on three dimensions (1–5 Likert scale):

- **Relevance**: Does this feature capture something meaningful and important for the brand?
- **Clarity**: Is the feature description precise and unambiguous?
- **Actionability**: Can a practitioner concretely apply this feature to evaluate new brand content?

K.3 Results

Table 11: Domain expert ratings of 15 FEST-discovered features per brand (1–5 Likert scale). Features were presented as a single blinded pool interleaving refined expert features and newly FEST-discovered features. All scores exceed 3.5, confirming features are relevant, clear, and actionable to practitioners.

Brand	Relevance	Clarity	Actionability
Zomato (images)	4.20	4.33	4.13
Adobe (images)	4.04	3.91	3.80

All scores exceed 3.5 across both brands and all three dimensions (range 3.80–4.33), confirming that FEST features are relevant, clear, and actionable to domain practitioners. The blinded design ensures that these ratings validate FEST's complete output, including both refined expert features and newly discovered data-specific features, without biasing the expert toward either category. The two brands span distinct industry sectors (enterprise creative software: Adobe; B2C food delivery: Zomato), providing broader coverage than a within-sector study.

K.4 Limitations

The study does not report inter-rater agreement because brand-evaluation expertise is scarce by design: a large panel of non-specialist annotators would produce low-signal ratings, and recruiting multiple independent practitioners per brand is infeasible within standard research constraints. Our design prioritizes depth over breadth, one highly qualified practitioner per brand which is the appropriate methodology when the task requires domain knowledge that cannot be distributed across a crowd.

L Runtime and Cost Analysis

Table 12: Runtime and cost comparison between FEST and Felix on brand text classification (mean $\pm$ std over 5 brands), using GPT-4o-mini (OpenAI, 2024) as the LLM backbone. FEST is 86$\times$ cheaper, 4.4$\times$ faster, and consumes 91$\times$ fewer tokens. FEST's efficiency stems from two complementary mechanisms: semantic consolidation reduces the feature space early via clustering, and iterative pruning eliminates low-discriminability features so they are never re-encoded in subsequent iterations.

Metric	FEST (Ours)	Felix
Cost per run (USD)	**\$0.10 $\pm$ 0.03**	\$8.62 $\pm$ 4.64
Runtime (minutes)	**15.9 $\pm$ 3.2**	69.4 $\pm$ 35.9
Total tokens	**75K $\pm$ 20K**	6.8M $\pm$ 3.5M
Cost ratio	**86$\times$ cheaper**	
Speed ratio	**4.4$\times$ faster**	
Token ratio	**91$\times$ fewer**	

Felix's one-shot generation produces a large feature set that must be fully encoded for every sample, leading to substantially higher token consumption. FEST's runtime is a one-time offline training cost; at inference, only a single LLM pass against 10–15 retained features is needed.

M VARIANCE ANALYSIS

We run 3 independent seeds on brand text classification (5 brands $\times$ 5 classifiers = 25 combinations).

Table 13: FEST stability: mean $\pm$ std accuracy (%) over 3 independent seeds across all tasks. For brand text classification, the maximum std is 3.03pp (MLP/Emirates, XGB/Porsche), with the majority below 2pp. Brand image classification shows higher variance (max 6.2pp, XGB/Porsche), reflecting greater stochasticity in visual feature generation. Content authenticity and stress detection also exhibit larger variance for LR and MLP classifiers (up to 6.9pp), likely due to smaller dataset sizes amplifying seed sensitivity. K-means deduplication and tree-based pruning stabilize stochastic LLM generation into consistent feature banks; residual variance stems from classifier sensitivity rather than feature bank instability.

Task	Dataset	DT	LR	RF	MLP	XGB
Brand Classification (Text)	*Average*	79.9	81.6	84.6	83.5	84.9
	Adobe	87.3$\pm$1.1	86.0$\pm$0.9	88.7$\pm$0.9	86.7$\pm$0.8	89.1$\pm$0.4
	Emirates	73.6$\pm$2.1	73.7$\pm$2.0	80.7$\pm$3.0	78.5$\pm$3.0	81.1$\pm$1.9
	Porsche	76.0$\pm$1.4	76.3$\pm$1.2	78.4$\pm$0.9	76.4$\pm$1.4	79.5$\pm$3.0
	Louis Vuitton	84.7$\pm$2.5	87.6$\pm$0.8	89.9$\pm$0.2	89.9$\pm$0.8	90.5$\pm$0.5
	Pizza Hut	78.1$\pm$1.1	84.5$\pm$0.9	85.4$\pm$0.9	86.2$\pm$1.9	84.4$\pm$1.3
Brand Classification (Images)	*Average*	78.9	79.1	84.6	78.6	84.6
	Adobe	81.9$\pm$0.4	83.7$\pm$1.9	87.6$\pm$0.5	81.1$\pm$1.9	86.5$\pm$1.5
	Emirates	77.9$\pm$1.2	77.6$\pm$1.4	83.7$\pm$1.5	77.6$\pm$0.3	83.5$\pm$1.1
	Porsche	72.7$\pm$2.2	73.0$\pm$3.0	77.0$\pm$4.8	70.8$\pm$4.4	79.3$\pm$6.2
	Louis Vuitton	73.0$\pm$2.4	73.3$\pm$1.0	81.6$\pm$2.5	74.5$\pm$2.4	81.6$\pm$3.3
	Pizza Hut	88.8$\pm$3.5	88.0$\pm$4.8	92.9$\pm$2.5	89.0$\pm$1.3	91.9$\pm$3.3
Content Authenticity		90.13$\pm$0.7	79.47$\pm$6.4	95.2$\pm$1.8	80.1$\pm$6.9	94.67$\pm$1.0
Stress Detection		77.6$\pm$0.3	74.1$\pm$4.6	82.1$\pm$1.8	73.6$\pm$5.6	80.5$\pm$0.7

N SYNTHETIC OFF-BRAND ROBUSTNESS

A natural concern is that FEST might learn trivial brand-specific entities or slogans rather than genuine voice/style patterns. We address this through both architectural design and empirical validation.

Architectural defense. Exclusion of trivial brand identifiers is a core design choice. It is explicitly enforced at all three LLM call sites in the FEST pipeline:

- **Stage 1 (Generation):** "Do NOT mention obvious identifiers like brand names, specific products, hashtags, URLs, logos, or location-specific references."
- **Stage 2 (Cluster summarization):** "Avoids any references to superficial brand identifiers (names, products, hashtags)."
- **Stage 3 (Feature inference):** "Ignore superficial identifiers: Brand names, Product names, Hashtags, URLs, Logos."

These constraints are enforced consistently at every LLM call, making the system structurally incapable of generating or inference on features like "mentions Porsche" or "includes product URL."

Empirical validation. We further validate by generating synthetic off-brand content using GPT-4o-mini that matches each brand's topics but uses generic writing style (no brand-specific voice). The negative class is topically identical to the brand's real posts; a shortcut learner relying on trivial brand markers would fail.

Table 14: FEST accuracy on synthetic off-brand content. Topic-matched but style-generic content is generated per brand using GPT-4o-mini. High accuracy confirms FEST captures voice/style, not brand name shortcuts.

Brand	Accuracy (%)	Top discovered features
Adobe	84.4	instructional tone, practical guidance
LG	79.8	vivid imagery, sensory language
Porsche	91.2	emotional storytelling, sentence length variance

FEST maintains 79.8–91.2% accuracy even when the negative class contains the same topics, brand names, and product keywords. The top features are purely stylistic: "instructional tone focused on practical applications" (Adobe SE), "vivid imagery and sensory language to forge emotional connections" (LG SE), "sentence length variance" (DE). No brand name or product mention features appear in the retained feature bank.

O CONTAMINATION DISCUSSION

Since GPT-4o-mini's training data is not publicly documented, we cannot fully rule out data contamination. However, several observations mitigate this concern:

1. **Same-LLM baselines**: Zero-Shot and Few-Shot baselines use the identical LLM on the same content. Any memorization benefit applies equally to all methods, yet FEST consistently outperforms these baselines, indicating methodological gains.
2. **Contamination-implausible datasets**: FEST achieves strong results on Dreaddit (Reddit posts about stress) and GPT-generated content detection. Reddit posts are unlikely to be memorized in their task-specific labels, and GPT-generated content detection requires distinguishing model outputs from human writing, not recalling training data.
3. **Feature interpretability**: FEST's features are expressed as natural-language descriptions or short executable functions and can be inspected for face validity. The discovered features (e.g., "uses emotional storytelling," "sentence length variance") are domain-meaningful, not artifacts of memorization.

We acknowledge this as a limitation inherent to all closed-source LLM evaluations and encourage future work with open-source models where training data provenance is verifiable.

P LIMITATIONS

While FEST demonstrates effective feature discovery and refinement capabilities, several limitations warrant discussion:

Binary classification scope: FEST currently addresses binary classification tasks. Extension to multi-class classification and regression requires architectural modifications, particularly in pairwise comparison formulation and decision tree feedback mechanisms.

Incomplete expert feature coverage: FEST achieves 60–80% coverage of expert-designed features across brand classification tasks (measured via LLM-as-judge evaluation). An audit of uncovered guidelines reveals they are structurally unobservable from post text (organizational policies, abstract brand philosophy, platform meta-guidelines). Achieving higher coverage may require multi-modal inputs or interactive expert feedback during feature generation.

LLM-dependent feature quality: Feature quality is bounded by the underlying LLM's capabilities and biases. While semantic deduplication and tree-guided pruning mitigate spurious features, the framework inherits limitations of the base LLM. On tasks where semantic features suffice (e.g., stress detection), simpler approaches may achieve comparable performance.

Computational overhead: Iterative feature generation, semantic clustering, and model retraining impose computational costs exceeding single-pass methods. For brand classification, FEST requires approximately 16 minutes per task on average ($0.10 per run), compared to Felix's 69 minutes ($8.62 per run). While 86$\times$ cheaper than the nearest baseline, these costs are amortized over deployment and may limit real-time applications. Inference requires only a single LLM pass against 10–15 retained features.

Correlation versus causation: Discovered features represent predictive patterns, not causal relationships. While decision tree paths provide interpretable rules, practitioners must validate features against domain knowledge before deployment in high-stakes contexts. Integration with causal discovery methods remains future work.

Limited domain validation: Evaluation focuses on marketing, content moderation, and psychological assessment tasks. Validation in critical domains (healthcare diagnosis, financial fraud detection,

legal decision-making) requiring stricter safety and regulatory compliance remains necessary before broader adoption.

Overall, these limitations point to directions for future work, including bias mitigation in LLM feature generation, integration with causal discovery methods and optimization for large-scale deployment.

Q FUTURE WORK

Beyond improving the scalability and causal grounding of FEST, we envision several exciting directions where the framework could extend its impact.

Content optimization through actionable feedback: Beyond classification, FEST generates explicit, interpretable features that form the decision path for each prediction. This opens the door to applications such as optimizing headlines, tweets, or advertisements. For example, when distinguishing between engaging and non-engaging content, FEST can surface the exact linguistic or structural attributes that influence predicted engagement. Practitioners can then receive concrete feedback such as “headline length too short" or “absence of emotional keywords", allowing them to modify content in ways directly aligned with model logic. This shifts predictive modeling from passive forecasting toward active guidance.

Post-hoc explainability of black-box models: Another intriguing direction is to repurpose FEST as an explanation layer for opaque models. Suppose a neural network achieves state-of-the-art accuracy on a classification task. By labeling data with the network's predictions and then running FEST on top, one can extract interpretable features and decision rules that approximate the network's learned representations. This would combine the high performance of black-box models with FEST's ability to articulate insights in natural language, offering practitioners a window into otherwise inscrutable models.

Scalability and causal discovery: On the methodological side, future work should push FEST toward more efficient large-scale deployment and explore causal discovery. Enhancing the efficiency of the generate, deduplicate, validate loop will make FEST practical for massive datasets and near real-time applications. Integrating causal reasoning into the feature refinement process could help distinguish predictive correlations from genuine drivers of outcomes, a particularly critical need in scientific and policy domains.

Multiclass extension: Future work can extend FEST through one-versus-all decomposition: for k classes, train k FEST classifiers, one per class-versus-rest split, and aggregate predictions at inference. Each split requires contrastive pairs comparing class i against the remaining classes. Negative samples can be pooled uniformly across those classes or drawn preferentially from classes the current classifier confuses most, forming a hard-negative mining protocol.

Protected expert features: FEST prunes non-discriminative guidelines for the classification objective. When practitioners require retaining such guidelines, FEST can initialize expert seeds with protected retention weights, exempting them from Gini-importance pruning while still refining their feature definitions.

Feature-grounded supervision: FEST's decision-tree paths offer a concrete mechanism for training LLMs into stronger, grounded reasoners. A path represents a sequence of verified feature evaluations, such as checking a semantic predicate followed by an executable threshold. These paths can provide step-by-step supervision for process reward models, while the corresponding trajectories can be converted into structured chain-of-thought fine-tuning data grounded in auditable criteria rather than final labels alone.

Taken together, these directions suggest that FEST is not only a framework for automating feature engineering but also a step toward rethinking the role of models in human decision-making: from opaque predictors to transparent copilots that explain, advise and guide.

R IMPACT STATEMENT

FEST's automation of feature engineering for high-stakes domains carries both benefits and mitigated risks. By democratizing access to expert-validated feature discovery, FEST can accelerate

ML deployment in domains like marketing, healthcare, and legal decision-making where manual feature engineering currently limits scalability. The framework's interpretability through practitioner-inspectable features and transparent decision tree paths supports accountability and enables domain experts to verify automated discoveries before deployment.

Importantly, FEST's demonstrated ability to discover features relevant to domain experts (as measured by 60–80% coverage of brand voice characteristics) provides empirical evidence that mitigates risks typically associated with fully automated methods. The expert refinement capability further aligns discovered features with domain knowledge, reducing concerns about arbitrary or opaque feature generation. However, practitioners should maintain human oversight to prevent perpetuation of biases present in training data or expert guidelines. The framework should augment rather than replace domain expertise, particularly in contexts where erroneous predictions carry significant consequences. We recommend validation of FEST-discovered features by domain experts before production deployment in sensitive domains.

S HYPERPARAMETERS

S.1 TEMPERATURE CONFIGURATION

Table 15: Per-component LLM temperature settings. Higher temperatures encourage diversity for feature generation and exploration stages, while near-zero temperatures ensure deterministic encoding during feature inference.

Pipeline Stage	**Temperature**	**Rationale**
SE feature generation	0.5	Encourage diverse hypotheses
Feature inference	0.01	Near-deterministic encoding
Cluster summarization	0.2	Balanced precision
DE feature ideation	0.7	Creative exploration
DE code generation	0.1	Precise implementation

S.2 OTHER HYPERPARAMETERS

- Batch size K: 50 pairs per iteration
- K-means clusters: $k = 30$ (for SE feature deduplication)
- Convergence threshold $\tau_{accuracy}$: 0.95
- Importance pruning threshold $\tau_{importance}$: 0.04
- Prompt templates per pair (M): 3
- Typical feature counts: $\sim$300 raw SE candidates per iteration $\rightarrow$ 30 after K-means $\rightarrow$ 15 after DT pruning across iterations; DE features per iteration: 5–10
- Embedding model: Qwen3-Embedding-4B for conditional embeddings
- Similarity threshold for semantic clustering of features: 0.8

T PROMPT TEMPLATES

Below we provide representative prompt templates for each FEST pipeline stage. These templates capture the core structure, input/output format, and key constraints; the full production prompts include additional task-specific instructions and formatting details. Placeholders are shown in `{braces}`.

P.1 SE Feature Proposal

System: You are a {domain} expert specializing in {task_description}.
Input:
Positive sample (class +): {positive_sample}
Negative sample (class −): {negative_sample}
Task: Identify {num_features} specific, concrete, and verifiable features (15–20 words each) that explain why the positive sample belongs to class +. Focus on observable, measurable characteristics. {task_constraints}
Output: JSON {"features": ["feature 1", "feature 2", ...]}

In practice, FEST issues both a positive-discriminator prompt (as above) and a symmetric negative-discriminator prompt (identifying why the negative sample fails to match class +). Both variants use the same template with reversed sample order.

P.2 Cluster Summarization

System: You are a {domain} consultant specializing in defining guidelines.
Input: A cluster of related features: {cluster_features}
Task: Create a single representative feature that captures the core characteristic shared by most features in the cluster. Exclude outliers. Keep concise ($\leq$20 words). {task_constraints}
Output: Plain text (single feature summary)

P.3 SE Feature Inference

System: You evaluate whether features are present in content samples related to {task_description}. {task_constraints}
Input:
Sample: {sample}
Features: {feature_list}
Task: For each feature, indicate whether the sample exhibits it.
Output: JSON {0: "1", 1: "0", 2: "1", ...}

Token log-probabilities for "1"/"0" are extracted and normalized to obtain continuous confidence scores (§3.2.3).

P.4 DE Feature Ideation

System: You are a data scientist specializing in feature engineering for {task_description}. Propose features that are: (1) objectively computable in Python, (2) deterministic, (3) CPU-efficient.
Input:
Positive samples: {positive_samples}
Negative samples: {negative_samples}
Existing features (do NOT duplicate): {existing_features}
Task: Propose up to {max_features} new computable features. For each: name (snake_case), description, data_type (numeric|boolean), rationale.
Output: JSON {"features": [{"name": ..., "description": ..., "data_type": ..., "rationale": ...}, ...]}

P.5 DE Code Generation

System: You are an expert Python developer specializing in feature extraction.
Input: Feature name: {name}, Description: {description}, Data type: {data_type}, Input modality: {input_modality}
Task: Write a Python function extract_feature({signature}) that returns a single value. Handle edge cases gracefully. Allowed libraries: {allowed_imports}. No ML models or external APIs.
Output: Python code only (no markdown fences).

Generated code is sandbox-validated (compiled, executed on sample data, output-type checked) before inclusion in the feature bank.

U BrandGuide Dataset Details

This appendix provides comprehensive details on the BrandGuide dataset, including our collection methodology, quality assurance procedures, extended statistics, and representative examples.

U.1 Collection Pipeline

Our multi-stage pipeline combines automated extraction with rigorous quality control to ensure dataset integrity:

1. **Data Acquisition**: We systematically collected brand guidelines from the web, extracting rich metadata including publication year, geographic region, language, and sector tags. Initial collection yielded 3,466 candidate entries spanning 1963–2025.
2. **Temporal Filtering**: To ensure contemporary relevance and consistency in design conventions, we filtered entries to the 2014–2025 timeframe, yielding 2,736 brands that reflect modern digital-first brand systems.
3. **Guideline Extraction**: Each document undergoes structured parsing to extract design specifications as text including color codes (HEX, RGB, CMYK, Pantone), typography hierarchies (primary/secondary typefaces, weights, sizes), logo clearance rules (minimum sizes, spacing requirements), and usage constraints (approved/prohibited applications).
4. **Visual Asset Retrieval**: For each brand, we retrieve imagery through web search using brand name and relevant keywords. We collect real-world logo applications, color implementations, marketing collateral, and brand touchpoints. This process yielded approximately **1M** brand images and textual descriptions across all entries.
5. **Manual Verification**: Each stage incorporates human review to ensure annotation accuracy, filter malformed entries, and validate guideline-image alignment. Annotators verified that extracted specifications match source documents and that retrieved images accurately represent the corresponding brand.

U.2 Quality Assurance

The metadata extraction pipeline processed 2,736 PDF guidelines and removed 53 non-aligned documents, leaving 2,683 documents in the released corpus.

To measure dataset reliability, we conducted a human verification study on a 10% stratified random sample ($n = 300$ brands, approximately 300 human-annotator hours). Two annotators independently evaluated each record, and a third senior annotator adjudicated disagreements. Table 16 reports exact-match agreement for sector classification and official website URL.

Table 16: BrandGuide metadata verification on a stratified random sample of 300 brands.

Field	Model–Human	Human–Human
Sector classification	269/300 (89.7%)	283/300 (94.3%)
Official website URL	278/300 (92.7%)	298/300 (99.3%)

Model–human agreement falls only modestly below human–human ceiling agreement, with residual errors concentrated in multi-sector brands and regional domain variants. We also implemented the following quality-control procedures:

- **Specification Validation**: Extracted color codes were validated against standard formats; typography specifications were checked for completeness.
- **Image Filtering**: Retrieved images underwent automated filtering for resolution, followed by manual review for ambiguous cases.
- **Duplicate Detection**: We removed duplicate brands and near-duplicate guideline versions, retaining the most recent edition for each brand.

- **Metadata Verification**: Publication years, regions, and sector tags were cross-referenced with source documents and corrected where inconsistencies were detected.

U.3 Extended Statistics

Geographic Distribution. The dataset exhibits strong international coverage with representation from 103 regions. While USA (35.0%), UK (9.8%), and France (6.7%) comprise the largest segments, substantial coverage spans Europe (Germany, Spain, Italy, Netherlands, Switzerland), Asia (Japan, India, Indonesia, China), and Latin America (Brazil, Colombia, Mexico). This diversity enables cross-cultural analysis of design conventions and regional branding patterns.

Sector Diversity. Guidelines span 80 sectors including Education (385), Sport (230), Technology (144), Software (134), Food & Beverage (117), and Financial Services (104). The sector distribution reflects real-world brand guideline availability, with educational institutions and sports organizations particularly well-represented due to their public communication requirements. This diversity enables domain-specific analysis of design conventions across organizational types.

Temporal Coverage. With guidelines spanning 2014–2025, BrandGuide captures contemporary design trends during the era of digital-first brand systems, responsive identity design, and the rise of design systems. The distribution peaks in 2019 (375 brands) and shows consistent coverage across years, enabling longitudinal studies of evolving design practices.

Language Distribution. English dominates (79.4%), reflecting global business practices, but the dataset includes substantial multilingual coverage: French (153), Spanish (108), Portuguese (47), German (37), and Arabic (27), among 28 total languages. This enables research on language-specific design conventions and cross-lingual brand communication.

Table 17: BrandGuide dataset overview: 2,683 brand guidelines across 80 sectors, 103 regions, and 28 languages (2014–2025).

(a) Summary Statistics

Statistic	Value	Statistic	Value	Statistic	Value
Total brands	2,683	Geographic regions	103	Year span	2014–2025
Sectors	80	Languages	28	Total images	∼1M

(b) Distribution by Sector, Region, and Language

Sector	#	Region	#	Language	#
Education	385	USA	940	English	2,129
Sport	230	United Kingdom	262	French	153
Regional	202	France	179	Spanish	108
Corporate	186	International	134	Portuguese	47
Technology	144	Canada	126	German	37
Software	134	Australia	73	Arabic	27
Food & Beverage	117	Spain	67	Italian	26
Transport	109	Germany	64	Russian	20
Events	106	Italy	45	Chinese	20
Financial	104	India	40	Japanese	16
NGO	96	Japan	39	Indonesian	15
Tourism	87	Ireland	37	Catalan	13
Others (68)	783	Others (91)	677	Others (16)	72

(c) Temporal Distribution

2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025
144	203	251	311	321	375	295	267	173	148	133	62

U.4 Research Directions

BrandGuide supports multiple research directions: (1) *brand consistency verification*: automated compliance checking of design assets against guidelines; (2) *generative design*: training models to produce brand-coherent visual assets conditioned on textual specifications; (3) *design trend analysis*: studying temporal and geographic patterns in visual identity; and (4) *multimodal grounding*: learning alignments between natural language design descriptions and precise visual properties.

U.5 Dataset Examples

We provide representative examples demonstrating the structure and content of BrandGuide entries. Each example illustrates how expert-authored design specifications are paired with corresponding visual assets.

Example Structure. Each brand entry contains:

- *Brand metadata*: name, sector, region, language, publication year
- *Visual assets*: logo files, brand imagery

Representative Samples. We include two complete brand examples in the supplementary materials:

- `LG`: A global technology and electronics brand demonstrating comprehensive digital-first guidelines with detailed color systems, responsive logo variants, and extensive application rules across product categories.
- `Porsche`: A luxury automotive brand showcasing premium brand architecture with precise typography hierarchies, strict color specifications, and meticulous guidelines for maintaining brand prestige across touchpoints.

These examples illustrate the diversity of design approaches across sectors and the granularity of expert specifications captured in BrandGuide. Each folder contains the extracted guideline text, and corresponding visual assets.

U.6 Licensing and Access

BrandGuide will be released for non-commercial research purposes. The dataset provides structured access to brand guidelines and associated imagery; all underlying intellectual property rights remain with the respective brand owners and guideline authors. We do not claim ownership over any third-party brand assets included in the dataset. To ensure transparency and provenance, we will release: (i) a compiled attribution list identifying the authors of all brand guidelines, (ii) image URLs rather than raw image files, enabling independent provenance verification and allowing rights holders to request removal if needed. Users of BrandGuide are required to restrict usage to non-commercial academic research. Any other intended use must be communicated to the authors prior to deployment. By accessing the dataset, users acknowledge that compliance with the terms of the original copyright holders remains their responsibility.

V AI Use Statement

Generative AI tools were used as assistants during the preparation of this manuscript. Specifically, they were employed to: (1) polish and refine the language and clarity of written sections; (2) assist with formatting and organization of content; and (3) assist with writing and debugging code. All AI-assisted text and code were reviewed and verified by the authors. Generative AI tools were *not* involved in any aspect of the research ideation, methodology design, experimental design, data analysis or interpretation of results.

W Feature Examples

Table 18: Representative features discovered by FEST from brand social media post **text** (subset of the full feature bank, selected for qualitative inspection). SE = Semantic (LLM-scored); DE = Deterministic (executable Python function).

Feature	Type	Code
Louis Vuitton		
Authoritative, sophisticated tone celebrating cultural figures in high-fashion discourse.	SE	—
Formal vocabulary and complex sentence structures conveying luxury and exclusivity.	SE	—

Average sentence length (words per sentence).	DE	`def extract_feature(text: str):`
		`  if not text or not isinstance(text, str):`
		`  return 0.0`
		`  sentences = nltk.sent_tokenize(text)`
		`  if not sentences:`
		`  return 0.0`
		`  total_words = sum(len(nltk.word_tokenize(s)) for s in sentences)`
		`  return total_words / len(sentences)`

Average number of clauses per sentence (comma/semicolon delimited).	DE	`def extract_feature(text: str):`
		`  if not text or not isinstance(text, str):`
		`  return 0.0`
		`  sentences = nltk.sent_tokenize(text)`
		`  if not sentences:`
		`  return 0.0`
		`  clause_count = sum(len([c for c in re.split(r'[;,]', s) if c.strip()]) for s in sentences)`
		`  return clause_count / len(sentences)`

Emirates		
Clear, friendly short-form language for broad audience connection.	SE	—
Informative value propositions emphasizing travel experiences and service quality.	SE	—

Frequency of @-mentions of travel partners and destination accounts.	DE	`def extract_feature(text: str):`
		`  if not isinstance(text, str) or text is None:`
		`  return 0`
		`  mentions = re.findall(r'@\w+', text)`
		`  return len(mentions)`

Average character length of hashtags used in the post.	DE	`def extract_feature(text: str):`
		`  if not text or not isinstance(text, str):`
		`  return 0.0`
		`  hashtags = re.findall(r'#\w+', text)`
		`  if not hashtags:`
		`  return 0.0`
		`  return (sum(len(h) for h in hashtags) / len(hashtags))`

Pizza Hut		
Urgency-driven language with compelling calls-to-action for limited-time offers.	SE	—

Feature	Type	Code
Vivid sensory language and emotional storytelling evoking food cravings.	SE	—

Ratio of emoji characters to total post length.	DE	`def extract_feature(text: str):`
		`  if text is None or not isinstance(text, str):`
		`  return 0.0`
		`  total_chars = len(text)`
		`  if total_chars == 0:`
		`  return 0.0`
		`  emoji_count = sum(`
		`  1 for c in text`
		`  if unicodedata.category(c).startswith('So'))`
		`  return emoji_count / total_chars`

Ratio of exclamation marks to total punctuation marks.	DE	`def extract_feature(text: str):`
		`  if not text or not isinstance(text, str):`
		`  return 0.0`
		`  punc = sum(1 for c in text if c in string.punctuation)`
		`  if punc == 0:`
		`  return 0.0`
		`  return text.count('!') / punc`

Table 19: Representative features discovered by FEST from brand promotional **images** (subset of the full feature bank, selected for qualitative inspection). SE = Semantic (LLM-scored); DE = Deterministic (executable Python function).

Feature	Type	Code
Porsche		
Sleek dynamic shapes and bold colors showcasing automotive craftsmanship and precision.	SE	—
Dynamic angles and action-oriented compositions conveying speed and performance.	SE	—

Aspect ratio (width/height) capturing widescreen landscape framing for car photography.	DE	`def extract_feature(image_path: str):`
		`  if not image_path or not isinstance(image_path, str):`
		`  return None`
		`  try:`
		`  image = PIL.Image.open(image_path)`
		`  width, height = image.size`
		`  if height == 0:`
		`  return None`
		`  return width / height`
		`  except (FileNotFoundError, OSError):`
		`  return None`

Feature	Type	Code
Density of edges detected via Canny operator (sharp, precise automotive lines).	DE	`def extract_feature(image_path: str): if not image_path or not isinstance(image_path, str): return 0.0 if not os.path.isfile(image_path): return 0.0 try: img = cv2.imread(image_path, cv2.IMREAD_GRAYSCALE) if img is None: return 0.0 edges = cv2.Canny(img, 100, 200) return cv2.countNonZero(edges) / img.size except Exception: return 0.0`
Pizza Hut		
Vibrant warm color palette (red-dominant) evoking appetite and energy.	SE	—
Bold playful typography with festive visual elements and vibrant colors.	SE	—
Luminance-weighted color contrast ratio between dominant and background color.	DE	`def extract_feature(image_path: str): if not image_path or not isinstance(image_path, str): return 0.0 try: img = PIL.Image.open(image_path).convert("RGB") except Exception: return 0.0 pixels = list(img.getdata()) if not pixels: return 0.0 dominant = collections.Counter(pixels).most_common(1)[0][0] background = pixels[0] def lum(c): return 0.2126*c[0] + 0.7152*c[1] + 0.0722*c[2] L1, L2 = lum(dominant), lum(background) lo, hi = min(L1, L2), max(L1, L2) return (hi + 0.05) / (lo + 0.05)`

Feature	Type	Code

Count of distinct visual elements via HSV color segmentation (food variety and abundance).	DE	`def extract_feature(image_path: str):`
		`  if not image_path or not isinstance(image_path, str):`
		`  return 0`
		`  if not os.path.isfile(image_path):`
		`  return 0`
		`  try:`
		`  image = cv2.imread(image_path)`
		`  if image is None:`
		`  return 0`
		`  hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)`
		`  lo = np.array([0, 50, 50])`
		`  hi = np.array([180, 255, 255])`
		`  mask = cv2.inRange(hsv, lo, hi)`
		`  contours, _ = cv2.findContours(`
		`  mask, cv2.RETR_EXTERNAL,`
		`  cv2.CHAIN_APPROX_SIMPLE)`
		`  return len(contours)`
		`  except Exception:`
		`  return 0`

Louis Vuitton		
Warm sophisticated palette (browns, golds, creams) evoking timeless luxury and elegance.	SE	—
High-quality artistic fashion photography with elegant, sophisticated compositions.	SE	—

Horizontal symmetry score via grayscale histogram intersection of left/right halves.	DE	`def extract_feature(image_path: str):`
		`  if not image_path or not isinstance(image_path, str):`
		`  return 0.0`
		`  try:`
		`  image = PIL.Image.open(image_path).convert('L')`
		`  except (FileNotFoundError, IOError):`
		`  return 0.0`
		`  width, height = image.size`
		`  if width == 0 or height == 0:`
		`  return 0.0`
		`  left = image.crop((0, 0, width // 2, height))`
		`  right = image.crop(`
		`  (width // 2, 0, width, height)`
		`  ).transpose(PIL.Image.FLIP_LEFT_RIGHT)`
		`  lh = collections.Counter(left.getdata())`
		`  rh = collections.Counter(right.getdata())`
		`  score = sum(min(lh.get(p, 0), rh.get(p, 0))`
		`  for p in set(lh) | set(rh))`
		`  max_score = left.size[0] * left.size[1]`
		`  return score / max_score if max_score > 0 else 0.0`

Feature	Type	Code
Euclidean distance of bright-pixel centroid from image center (product subject offset).	DE	`def extract_feature(image_path: str): if not image_path or not isinstance(image_path, str): return 0.0 if not os.path.isfile(image_path): return 0.0 try: img = PIL.Image.open(image_path).convert("L") w, h = img.size px = np.array(img) fg = px > np.mean(px) ys, xs = np.where(fg) if len(xs) == 0: return 0.0 cx, cy = np.mean(xs), np.mean(ys) return math.sqrt((cx - w/2)**2 + (cy - h/2)**2) except Exception: return 0.0`

X LICENSING FOR EXISTING ASSETS

- **GPT-GC** (Zhou et al., 2024), used for content authenticity detection: MIT License.
- **Dreaddit** (Turcan & McKeown, 2019), used for stress detection: distributed by the authors for research purposes via the ACL Anthology (DOI: 10.18653/v1/D19-6213) and their institutional page at Columbia University. No explicit open-source license is assigned; we use it solely for non-commercial academic research, consistent with its intended distribution and community norms.
- **Engaging ImageNet** (Khurana et al., 2025): CC BY-NC-ND 4.0.