

Article

Text2MFN and MFPredictor: Text-Conditioned Mesh Face-Count Recommendation for Efficient 3D Generation

Bingyang Ji  and Shan Cui *

School of Information Science and Engineering, Linyi University, Linyi 276000, China; jibingyang@lyu.edu.cn

* Correspondence: cuishan@lyu.edu.cn

Featured Application

MFPredictor serves as a lightweight front-end module that converts a natural-language asset request into a triangular-face budget for controllable mesh generation, QEM-based simplification, or progressive level-of-detail refinement.

Abstract

Current text-to-3D methods primarily improve geometric fidelity, topology, and generation efficiency, but generally require mesh complexity to be determined indirectly by architecture, sequence length, user settings, or post-processing. We formulate text-conditioned mesh face-count recommendation as a regression task that predicts a practical triangular-face budget from the object category, silhouette, semantic parts, structural relationships, and geometry-related details expressed in a prompt. We construct Text2MFN, a synthetic dataset containing 1,000,000 text–face-count pairs, and propose MFPredictor, a lightweight dual-branch model that separates semantic context from geometry-related complexity cues and combines them through bidirectional calibration, gated fusion, a small mixture-of-experts module, and residual regression. All models are trained exclusively on Text2MFN and evaluated on its held-out test set and on an independent benchmark containing 200,000 QEM-derived multi-LOD records from 40,000 Objaverse meshes. MFPredictor obtains a MAPE of 4.38% and a $\pm 20\%$ accuracy of 95.45% on Text2MFN, and a MAPE of 7.96% and a $\pm 20\%$ accuracy of 86.41% on the Objaverse-based benchmark. With 481,275 parameters and a measured average regression latency of 2.09 ms, the model provides an efficient front-end prediction of mesh complexity. The real-mesh results demonstrate transfer under the evaluated Text2MFN-to-Objaverse/QEM protocol and define the validated scope of the reported generalization.

Keywords: text-to-3D; mesh face-count recommendation; mesh complexity prediction; Text2MFN; MFPredictor; synthetic-to-real transfer; regression reliability



Received: 6 July 2026

Revised: 31 July 2026

Accepted: 2 August 2026

Published: 3 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Recent advances in generative artificial intelligence have substantially improved the generation of 3D content from natural-language descriptions. The field has progressed from implicit fields and volumetric representations toward direct polygonal-mesh generation. Representative systems, including MeshGPT [1], MeshXL [2], Meta 3D AssetGen [3], MeshAnything and MeshAnything V2 [4,5], MeshLLM [6], TreeMeshGPT [7], and MeshArt [8], improve mesh tokenization, topology, structural organization, articulation, texture, and generation quality. These developments make text-driven meshes increasingly useful in games, virtual and augmented reality, animation, simulation, and digital-content production.

Despite this progress, geometric complexity is still usually controlled *after* the semantic request has been interpreted, rather than predicted as an explicit decision variable before generation. Native mesh generators commonly inherit a complexity range from their token budget, model capacity, decoding length, or user-specified resolution. Compression-oriented methods increase the number of representable faces within a sequence budget, progressive methods require a stopping rule, and simplification methods require both an existing mesh and a target complexity. These approaches improve how a mesh is represented, generated, or reduced, but do not directly answer the prior question of how many triangular faces are warranted by a particular textual description [5,9–11].

This missing decision is practically important because prompts within the same object category imply different geometric requirements. A stool with a flat seat and four thick legs is represented with a small budget, whereas a mechanical product with curved shells, exposed joints, repeated fasteners, thin components, and perforated panels requires substantially more triangles. An excessive budget increases generation and rendering time, GPU memory, storage, editing effort, and runtime cost. An insufficient budget damages the silhouette, removes functional parts, closes holes, merges thin structures, or fails to express the requested articulation.

The central question studied in this work is therefore:

What triangular-face budget is appropriate for the semantic and geometric requirements expressed in a natural-language description?

To assess practical demand for prompt-dependent mesh-complexity control, we conducted a cross-sectional online survey that began on 3 July 2026 and retained 1000 valid respondents. The questionnaire was hosted on Wenjuanxing, and recruitment used the Wenjuanxing online-survey platform (WJX), combining its sample-recruitment service with convenience and snowball distribution through Chinese-language WeChat and QQ communities related to 3D modeling, computer graphics, game development, digital-content production, and AI-based 3D generation. This was a non-probability convenience-and-snowball sample; consequently, the survey characterizes the experiences and preferences of the recruited respondents; the non-probability sampling design does not support population-prevalence inference.

User-study design and analysis.

Eligibility required being at least 18 years old, being able to read Chinese, providing electronic informed consent, and reporting practical or educational experience during the preceding 12 months with at least one of the following: 3D modeling, mesh editing, game-engine asset preparation, rendering, animation, or AI-based 3D generation. The landing page stated the study purpose, stated completion time, voluntary nature of participation, eligibility requirements, specified use of the responses, data-handling procedure, and contact details for the research team. Respondents were free to leave the survey before submission, and partial answers were excluded from the analysis. No names, telephone numbers, email addresses, precise locations, or other directly identifying information were requested. Platform-level anti-duplication metadata were used only to prevent repeated submissions and were not included in the analysis dataset.

Before the main survey, a pilot study with 30 eligible respondents tested wording, option clarity, mobile-device display, branching logic, and completion time. The 30-person pilot was a separate instrument test and was not part of the completed main survey of 1000 valid respondents. Pilot responses were not included in the final analysis. The main survey accepted at most one response per Wenjuanxing account and device. Responses were excluded according to the pre-specified rules: incomplete submission; absence of consent; failure to satisfy the age, language, or experience screen; failure of one instructed-

response attention check; a duplicate submission flagged by Wenjuanxing's one-response-per-account or one-response-per-device controls and confirmed by response-pattern review; or a completion time below both 30 s and one-third of the median completion time observed in the pilot study. Exclusions were recorded in a screening log, and recruitment ended after 1000 valid questionnaires had been retained.

The substantive questionnaire contained four closed-ended items: (i) frequency of encountering excessive face counts (Frequently, Occasionally, Rarely, or Never); (ii) experienced consequences (Rendering Speed, GPU Memory Usage, Editing Efficiency, Game Runtime Performance, Storage and Transfer, or No Significant Impact), for which multiple selections were permitted; (iii) perceived need for automatic face-count recommendation (Highly Needed, Needed, Neutral, Slightly Needed, or Not Needed); and (iv) willingness to use such a tool (Very Willing, Willing, Uncertain, Unwilling, or Very Unwilling). In Wenjuanxing, "No Significant Impact" was mutually exclusive with the five specific consequence options. One separate instructed-response attention check was used for data-quality screening but was not treated as a substantive survey item.

The analysis reports option-level counts and percentages using the number of valid respondents as the denominator. For the multiple-response consequences item, each option was divided by the total number of valid respondents. The reported percentages sum to 227.0% because respondents selected multiple effects. The three pre-specified combined summaries were frequent-or-occasional exposure, needed-or-highly needed responses, and willing-or-very-willing responses. Their 95% Wilson confidence intervals were calculated following Brown et al. [12]. Because recruitment was non-probability based, the results are interpreted descriptively and are not used for causal or population-wide inference. Reporting follows the relevant transparency items of the CHERRIES guidance for internet surveys [13].

Survey results.

Figure 1 summarizes the responses. A total of 650 respondents (65.0%; 95% Wilson CI, 62.0–67.9%) reported frequently or occasionally encountering excessive face counts. Automatic face-count recommendation was rated as needed or highly needed by 729 respondents (72.9%; 95% Wilson CI, 70.1–75.6%), and 710 respondents (71.0%; 95% Wilson CI, 68.1–73.7%) were willing or very willing to use such a tool. The most frequently reported effects of excessive face counts were reduced rendering speed (61.1%), increased GPU-memory usage (57.2%), reduced editing efficiency (52.2%), and lower game-runtime performance (42.5%). Storage and transfer effects were selected by 3.3% of respondents, while 10.7% reported no significant impact. These findings support practical demand for prompt-dependent mesh-complexity control within the recruited sample.

We formulate this problem as *text-conditioned mesh face-count recommendation*. Given a free-form description, the objective is to predict a practical triangular-face budget from the object category, global silhouette, semantic components, structural relationships, curvature, holes, repetition, articulation, thin structures, and essential local details. The target is neither the original face count of an existing asset nor a mathematically unique optimum. It is a protocol-dependent recommendation that serves as an input constraint for mesh generation, a target for simplification, or a stopping condition for progressive refinement. Its appropriate value varies with the generator, topology, rendering resolution, viewing distance, deformation requirements, hardware, and application.

The task presents three main methodological challenges. First, existing 3D–text resources such as Objaverse, Objaverse-XL, and Cap3D [14–16] do not provide recommendation labels. Their original face counts reflect heterogeneous authoring, reconstruction, subdivision, remeshing, and export processes and are therefore confounded by factors that are not expressed in the captions. Second, the language–complexity relationship is

nonlinear and heterogeneous: two objects in the same category require different budgets when their part count, curvature, topology, repetition, articulation, or viewing conditions differ. Third, text length is not a reliable proxy for geometry: color and material clauses lengthen a description without increasing geometric demand, while concise articulation clauses increase geometric demand.

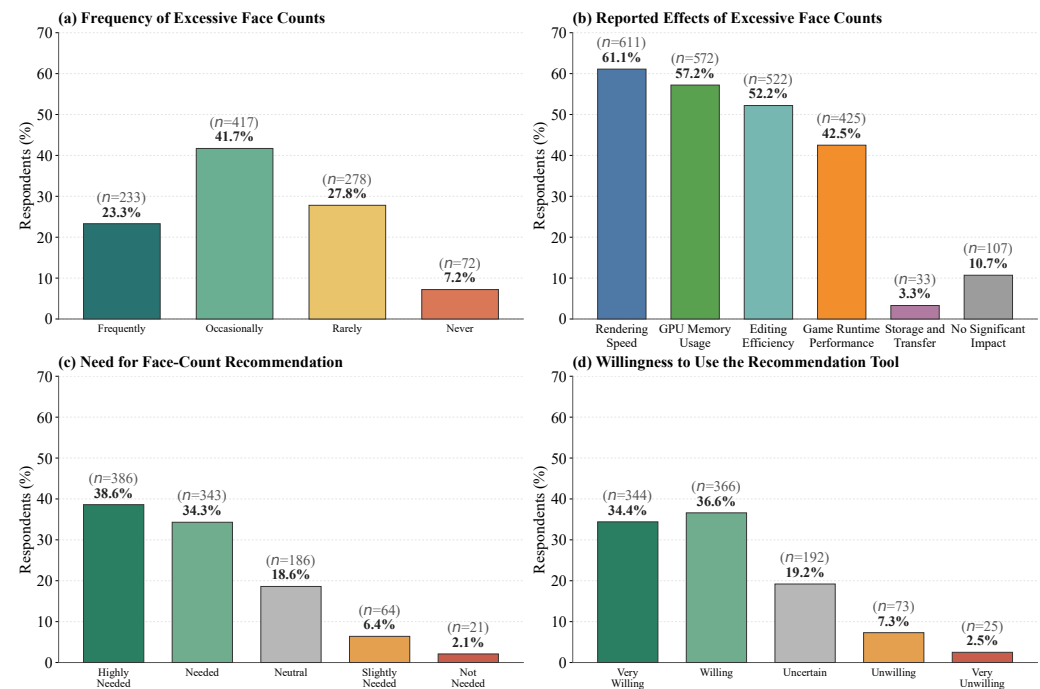


Figure 1. Survey results from 1000 valid respondents on practical demand for mesh-complexity control. (a) Frequency of encountering excessive face counts; (b) reported effects of excessive face counts; (c) perceived need for automatic face-count recommendation; and (d) willingness to use the recommendation tool. Multiple responses were allowed in panel (b), so its percentages do not sum to 100%.

To provide task-specific supervision, we construct **Text2MFN**, a synthetic dataset containing 1,000,000 text–face-count pairs. Its closed-loop pipeline combines diversity-controlled generation, duplicate detection, contradiction filtering, distribution checking, and prompt refinement. Because synthetic-only evaluation does not establish transfer [17–20], we additionally construct an independent benchmark from 40,000 Objaverse meshes. Each mesh is triangulated, validated, and simplified to five QEM levels, producing 200,000 text–face-count records whose targets are measured from the corresponding LOD meshes.

We also propose **MFPredictor**, a lightweight dual-branch regression model for frozen 384-dimensional sentence embeddings. A semantic branch captures category, part composition, and structural context, while a geometry-aware bottleneck focuses the representation on compact complexity cues. Bidirectional calibration and gated fusion model context-dependent interactions between the two branches, and a small mixture of experts captures heterogeneous mappings associated with different geometric regimes. The model predicts $\log_2(m + 1)$ to reduce the influence of the right-skewed face-count distribution.

All models are trained exclusively on Text2MFN. On its held-out test set, MFPredictor obtains a MAPE of 4.38% and an Acc₂₀ of 95.45%. On the independent Objaverse-based QEM benchmark, it obtains a MAPE of 7.96% and an Acc₂₀ of 86.41%. These results provide evidence that the learned text–complexity relationship transfers to the evaluated real-mesh protocol without real-data training or fine-tuning. They do not establish universal

generalization to other 3D collections, mesh generators, representations, languages, or deployment settings.

The main contributions are as follows:

1. We formulate text-conditioned mesh face-count recommendation as an explicit pre-generation regression problem and distinguish it from mesh generation, simplification, and LOD control.
2. We construct Text2MFN with 1,000,000 synthetic text–face-count pairs and an independent real-mesh benchmark with 200,000 multi-LOD records derived from 40,000 Objaverse objects.
3. We propose MFPredictor, a dual-branch model with fewer than 0.5 million trainable parameters, whose inductive biases are designed for the heterogeneous relationship between semantic structure and geometric complexity.
4. We evaluate the task with classical, tree-based, neural, neighborhood-based, and Transformer-based regression baselines; component and objective ablations; repeated-run statistics; clustered bootstrap confidence intervals; and reliability analysis of prediction intervals.

The remainder of this paper critically reviews related research, defines the task and datasets, presents the model and its design rationale, details the reproducible evaluation protocol, analyzes the results and statistical uncertainty, and discusses the scope and limitations of the conclusions.

2. Related Work

2.1. Text-to-3D and Native Mesh Generation

Direct mesh generation is attractive because polygonal assets support rendering, editing, animation, simulation, and deployment without conversion from an implicit representation. MeshGPT [1] and MeshXL [2] generate irregular triangular geometry through learned token or coordinate representations. Meta 3D AssetGen [3] extends text-to-mesh generation to high-quality geometry, textures, and physically based materials. MeshAnything and MeshAnything V2 [4,5], TreeMeshGPT [7], and compressive tokenization [9] improve adjacency modeling, sequence organization, and token efficiency. MeshLLM [6] and MeshArt [8] further emphasize structural reasoning, primitive decomposition, and articulated parts.

These methods solve the difficult problem of *how to generate or represent a mesh*, but their complexity is generally determined by model capacity, token limits, decoding settings, training distributions, or manually chosen quality controls. A larger token capacity allows more geometry to be generated, but it does not determine whether that geometry is warranted by a given prompt. Consequently, native generation and face-count recommendation address different stages of the pipeline: the former synthesizes geometry, while the latter predicts an appropriate budget before synthesis.

VertexRegen [10] introduces continuous LOD generation by reversing simplification trajectories. This improves resolution control, but a requested LOD or stopping point is still required. MFPredictor is complementary: its numerical prediction supplies the stopping condition from text.

2.2. 3D–Text Datasets and Synthetic Supervision

Objaverse [14] and Objaverse-XL [15] provide large and diverse 3D collections, while Cap3D [16] produces object-level captions from multi-view renderings and pretrained vision–language models. Later work shows that caption reliability depends strongly on view selection and quality control [21]. These resources support text-conditioned generation and 3D understanding, but their original polygon counts are not reliable recommendation

labels. Two semantically similar objects have different counts when one was scanned densely, subdivided, reconstructed, or exported with different settings. Direct regression on those counts conflates authoring artifacts with text-conditioned geometric need.

Synthetic supervision is a practical alternative when task-specific human labels are expensive or intrinsically ambiguous. Surveys of LLM-generated data emphasize generation, filtering, diversity, and downstream validation [18]. Magpie [17] demonstrates large-scale instruction synthesis from limited seed information, whereas Kim et al. [19] show that usefulness depends on accuracy, diversity, complexity, formatting, and task fit. At the same time, synthetic data contain risks of repeated templates, distributional shortcuts, annotation artifacts, and recursive-generation bias [20].

Text2MFN addresses these risks through a closed-loop protocol rather than an unfiltered one-pass generation. Its labels are protocol-dependent. Independent blinded audits and inter-rater reliability measures are therefore important complements to automatic filtering when the target is subjective or interval-valued [22,23]. For this reason, we do not treat performance on its synthetic test split as sufficient evidence of real-world transfer and evaluate all models on a separately constructed Objaverse-based multi-LOD benchmark.

2.3. Mesh Simplification and Level-of-Detail Modeling

Mesh simplification reduces geometric complexity while attempting to preserve silhouettes, boundaries, curvature, and important structures. The quadric error metric of Garland and Heckbert [11] remains a foundational approach because it assigns a geometric cost to contractions and efficiently produces discrete LODs. Robust low-poly meshing extends this objective to difficult inputs containing disconnected parts, non-manifold regions, intersections, and irregular tessellation [24].

Conventional LOD systems select resolution from approximation error, screen-space coverage, camera distance, hardware constraints, or predefined quality levels. Their limitation for the present task is not simplification quality, but the need for an existing mesh and an externally specified target. Progressive generators similarly require a resolution request or stopping criterion. Text-conditioned face-count recommendation supplies that missing target from the semantic and geometric requirements of the prompt; it neither replaces QEM nor competes with a mesh generator.

2.4. Regression Models, Predictive Uncertainty, and Reliability

The proposed task is fundamentally continuous regression over a dense learned text representation. Classical ensembles such as Random Forests [25] and gradient-boosted trees such as XGBoost [26] provide strong nonlinear baselines and model feature interactions without a task-specific neural architecture. For learned tabular or embedding features, plain multilayer perceptrons are important controls. FT-Transformer and ResNet-style tabular models demonstrate that carefully tuned generic neural baselines are highly competitive, while also showing that no single family consistently dominates gradient-boosted trees [27]. TabM [28], ModernNCA [29], and xRFM [30] represent parameter-efficient ensembling, learned neighborhood regression, and scalable nonlinear feature learning, respectively.

This literature motivates two requirements for our evaluation. First, MFPredictor must be compared not only with recent deep regressors but also with linear, tree-based, plain-MLP, and Transformer-based alternatives. Second, its task-specific branches are evaluated as task-specific inductive biases rather than as a theoretically optimal architecture. Their value must be established through controlled baselines and ablations.

A single numerical prediction does not communicate its reliability. Deep ensembles expose variation across independently trained predictors [31], calibration methods align stated regression uncertainty with empirical coverage [32], and conformal methods

construct finite-sample prediction intervals under exchangeability [33]. These ideas are particularly relevant here because a face-count label is protocol-dependent and multiple budgets are valid for the same description. We therefore supplement point metrics with repeated-run variation, bootstrap confidence intervals, statistical comparison, and split-conformal reliability analysis. We also use controlled minimal-pair tests inspired by behavioral evaluation methodology [34] to determine whether predictions respond to fine-grained geometric cues while the object category is held fixed.

Table 1 summarizes the distinctions among the method families related to mesh-complexity control.

Table 1. Critical comparison of method families related to mesh-complexity control.

Method Family	Primary Input	Primary Output	Existing Mesh?	Manual Target?	Text-to-Budget?	Limitation Relative to the Present Task
Native mesh generation	Text or 3D condition	Polygonal mesh	No	Often indirect	No	Complexity is inherited from token capacity, decoding settings, or the training distribution rather than explicitly inferred from the prompt. Expands representational capacity but does not estimate how much capacity is required by a specific prompt. Preserves geometry after a target is provided but cannot infer that target from language. Supports flexible resolution but still requires an externally specified quality level or stopping criterion.
Compressive tokenization	Mesh tokens or features	Compact mesh sequence	No	Usually indirect	No	
Mesh simplification	Mesh and target or error	Reduced mesh	Yes	Yes	No	
Progressive/LOD generation	Mesh or generation condition	Multi-resolution mesh	Depends on method	Yes	No	
MFPredictor	Natural-language description	Recommended face count	No	No	Yes	Infers a protocol-dependent budget from text; it does not generate or simplify meshes and does not guarantee a universally optimal face count.

3. Task Formulation and Dataset Construction

3.1. Task Definition

Given a natural-language description x of a 3D object, the proposed task predicts a recommended triangular-face budget $\hat{m}$:

$$\hat{m} = f_{\theta}(x), \quad (1)$$

where f_{θ} denotes a learnable regression model and $\hat{m} \in \mathbb{N}^+$ is the predicted number of triangular faces.

Input descriptions contain combinations of object category, global silhouette, semantic parts, structural relationships, curvature, holes, repeated components, articulation, thin structures, local details, modeling style, application, and viewing conditions. These factors jointly determine the geometric complexity assigned to the described object under the evaluation protocol.

In this work, mesh complexity is quantified using triangular face count. The predicted value is a practical recommendation rather than a mathematically unique optimum. The appropriate face count varies with the mesh generator, topology, rendering resolution, viewing distance, deformation requirements, target hardware, and application.

The predicted budget serves as an input constraint for controllable mesh generation, a target for mesh simplification, or a stopping condition for progressive mesh refinement.

Because the face-count distribution covers a wide numerical range and is strongly right-skewed, the regression target is transformed as

$$y = \log_2(m + 1), \quad \hat{m} = \max\left(1, \text{round}\left(2^{\hat{y}} - 1\right)\right), \quad (2)$$

where m is the reference face count and $\hat{y}$ is the model prediction. This transformation reduces the dominance of high-complexity samples and emphasizes relative prediction accuracy.

3.2. Closed-Loop Construction of Text2MFN

Existing resources such as Objaverse [14], Objaverse-XL [15], and Cap3D [16] provide large collections of 3D objects and captions. However, the original face count of an asset is affected by authoring software, subdivision settings, reconstruction procedures, remeshing operations, and export configurations. It therefore cannot be directly treated as a face-count recommendation derived from the associated text.

Figure 2 summarizes the closed-loop construction and quality-control pipeline used to build Text2MFN.

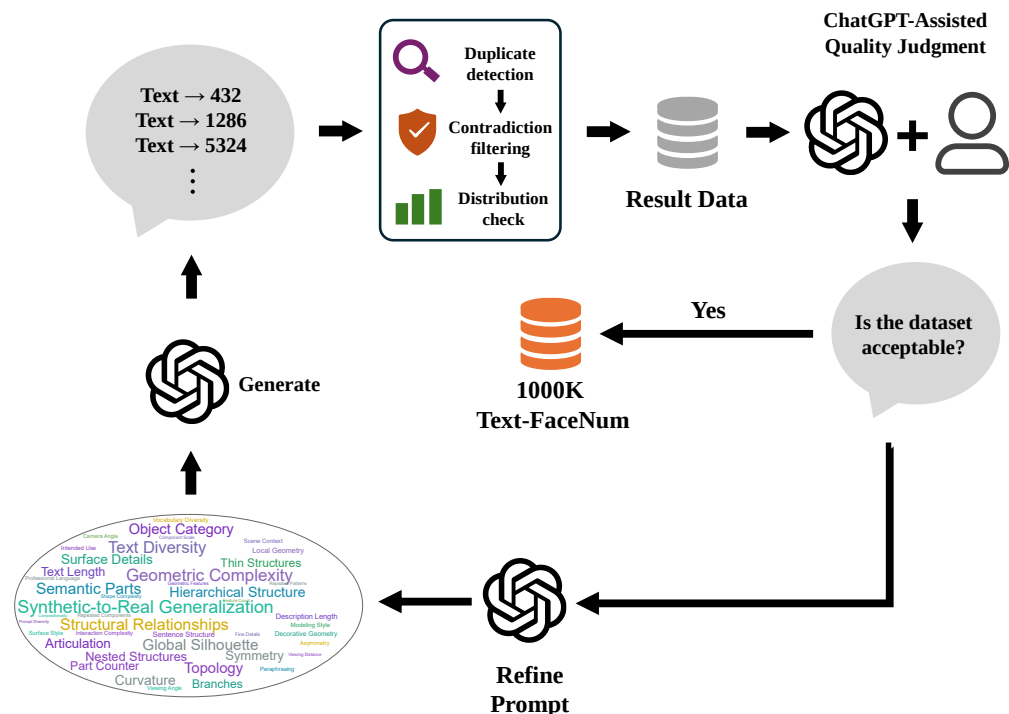


Figure 2. Closed-loop construction pipeline of Text2MFN. Candidate samples are generated from diversity-controlled prompts, evaluated through duplicate, contradiction, and distribution checks, and either accepted or regenerated after prompt refinement.

To provide task-specific supervision, we construct **Text2MFN**, a synthetic dataset containing 1,000,000 text–face-count pairs:

$$\mathcal{D}_{\text{Text2MFN}} = \{(x_i, m_i)\}_{i=1}^{1,000,000}, \quad (3)$$

where x_i is a natural-language description and m_i is the corresponding recommended triangular-face count. During construction, each candidate was stored internally in a structured record containing the description, face count, object category, geometry-factor tags, modeling style, intended application, viewing condition, generation-batch identifier, and quality-control status. The public Hugging Face release does not expose these auxiliary record-level fields. It is distributed as a single table containing only text and recommended_faces, which are the two fields used directly for regression modeling. The

public-table state used to define the S2 reference split was accessed on 28 July 2026 and is described by a dated snapshot manifest in the supporting information. The auxiliary metadata are summarized at protocol level in Supplementary File S1 and are not released at record level.

Generation protocol and prompt control.

Text2MFN was generated with OpenAI ChatGPT 5.5 in independent, stateless batches from a fixed prompt family. The prompt family explicitly varied object category, part composition, spatial relationships, silhouette complexity, curvature, holes, repeated components, articulation, thin structures, local detail, modeling style, application scenario, viewing condition, and linguistic form. It instructed ChatGPT 5.5 to produce both a natural-language description and a recommended triangular-face count in the closed interval [256, 16,384], with the count determined by the geometric requirements rather than by text length or category alone. Supplementary File S1 documents the dataset-construction and quality-control protocol, including the controlled generation factors, output constraints, normalization procedure, duplicate control, independent consistency review, and release-level checks.

Deterministic post-processing and duplicate control.

Candidate descriptions were normalized with Unicode NFKC normalization, lowercasing, whitespace collapse, and standardized punctuation before comparison. Exact normalized duplicates were removed. Near duplicates were detected with cosine similarity between frozen all-MiniLM-L6-v2 sentence embeddings using a fixed threshold and a pre-specified rule for retaining one representative from each connected duplicate component. Supplementary File S1 summarizes this duplicate-control procedure. Duplicate filtering was performed before data splitting, and all members of a detected duplicate component were assigned to a single partition to prevent cross-split leakage.

Contradiction and range-consistency filtering.

A second, independently instantiated review pass was given the description without the original generation rationale and returned (i) a lower and upper acceptable face-count bound, (ii) a three-level verdict (consistent, uncertain, or inconsistent), and (iii) machine-readable geometry-factor tags. A candidate was accepted only when its label lay inside the returned interval and the verdict was consistent. Uncertain, out-of-range, malformed, or internally contradictory records were rejected rather than manually relabeled. Rule-based checks additionally rejected non-integer counts, counts outside the allowed interval, missing required fields, incompatible statements about parts or topology, and descriptions whose declared complexity tags were inconsistent with their text. Supplementary File S1 summarizes the independent-review and rejection procedure.

Distribution and coverage control.

After each generation round, the accepted pool was summarized by face-count bins, object category, description-length bin, number of semantic parts, and the major geometry factors used by the prompt family. These checks were designed to detect numerical gaps, severe category concentration, absent geometry factors, abnormal values, and recurring linguistic templates; they did not force a uniform face-count distribution. The pre-specified coverage rules and acceptance thresholds were fixed before the final release-level audit. If a round failed a rule, only the deficient prompt strata were regenerated. Batch-level generation, rejection, retention, and cumulative coverage totals were monitored during construction.

Human oversight and release freeze.

The authors inspected all records flagged by rule conflicts and a stratified sample from every generation round. Manual review was restricted to record rejection and never altered a face-count label. Candidate generation, filtering, and targeted regeneration continued until 1,000,000 accepted records were obtained and all release-level checks passed. The final internal dataset, experimental split assignments, generation prompts, and filtering configuration were then frozen before model training. The independent blinded audit in Section 3.4 was conducted after this freeze and was not used to modify the released labels.

This protocol improves semantic, geometric, linguistic, and numerical diversity while reducing fixed-template, category-shortcut, contradiction, and split-leakage risks. It does not make the labels mathematically unique; their agreement with independent expert ranges is evaluated separately.

3.3. Use of GenAI Tools

OpenAI ChatGPT (version 5.5) was used as an auxiliary tool during the construction and iterative refinement of Text2MFN. Its roles were limited to prompt design, diversity-factor expansion, candidate text and label generation, and a separate machine-readable consistency-review pass. Supplementary File S1 documents the role of ChatGPT 5.5 and the distinction between the generation and independent-review stages. Generated records were not accepted directly: they passed deterministic schema, range, duplicate, contradiction, and distribution checks, together with author oversight and the independent audit described below.

ChatGPT was also used for language polishing and clarity improvement during manuscript preparation. It was not used to implement the final model, execute experiments, calculate the reported statistics, select favorable results, or make independent scientific conclusions. All experimental design, code verification, data-quality decisions, result analysis, and final manuscript decisions were made by the authors. The authors reviewed and edited all AI-assisted outputs and take full responsibility for the content of this publication.

3.4. Independent Annotation Reliability Audit Protocol

Because a recommended face count is not mathematically unique, annotation reliability was assessed with a blinded, interval-based audit rather than by requiring exact agreement with a single integer. The audit included 2000 Text2MFN records selected by stratified sampling across face-count deciles, object categories, description-length bins, and the major geometric factors used during data construction. Three raters with practical experience in 3D modeling or mesh processing independently reviewed each description without seeing the generated label. Each rater provided (i) a lower and upper acceptable face-count bound and (ii) an ordinal judgment of whether the hidden generated label was too low, appropriate, or too high after it was revealed in a second stage.

The primary reliability statistic was the proportion of generated labels contained in the intersection-free majority interval formed by the median lower and upper bounds across raters. Ordinal agreement for the post-reveal judgments was summarized with Krippendorff's α , treating the three ordered categories as ordinal values. These two statistics correspond directly to the quantities collected in the audit instrument: interval bounds and ordinal judgments. No point-valued expert recommendation, rank correlation, point-estimate Log-MAE, or numerical ICC is reported. The ordinal agreement coefficient is interpreted as evidence of protocol-level consistency rather than as a stand-alone proof of label validity [22]. Disagreements were additionally reviewed by face-count range and geometric factor to identify systematic failure modes in the generation prompt.

3.5. Dataset Splitting

After quality control, Text2MFN is divided into training, validation, and test subsets using a ratio of 7 : 2 : 1:

$$|\mathcal{D}_{\text{train}}| = 700,000, \quad |\mathcal{D}_{\text{val}}| = 200,000, \quad |\mathcal{D}_{\text{test}}| = 100,000. \quad (4)$$

The training subset is used only for parameter optimization. The validation subset is further partitioned, using a fixed stratified split, into 150,000 samples for hyperparameter selection and checkpoint selection and 50,000 samples for post hoc uncertainty calibration. The calibration partition is not used to update model parameters or choose hyperparameters. The held-out test subset is used only for final in-distribution evaluation. Normalized duplicate descriptions and all members of a detected near-duplicate component are not allowed to appear across partitions. The split was generated once with a recorded random seed and was treated as immutable throughout model development and final evaluation. The public Hugging Face release is provided as one split named `train` and does not contain per-record labels for the internal experimental partition. The exact internal assignment used to produce the reported tables is not redistributed. To provide a fixed and directly reusable partition for future comparisons, Supplementary File S2 releases a deterministic public reference split over the zero-based row indices of the current one-million-row public table. The reference split is generated with NumPy 2.3.5 using the PCG64 generator and seed 2026, and contains exactly 700,000 training, 150,000 model-selection, 50,000 calibration, and 100,000 held-out test indices. The combined assignment file, four partition-specific index files, generation script, and SHA-256 manifest are included in Supplementary File S2. Because the public repository uses the moving revision selector `main`, the row-index assignment is explicitly defined for the repository state accessed on 28 July 2026: one `train` split with 1,000,000 rows, fields `text` and `recommended_faces`, and a displayed total file size of 382 MB. Supplementary File S2 records this dated snapshot description, a SHA-256 metadata fingerprint, and a verification script. If the public table is subsequently replaced or reordered, the supplied row indices must be applied to an archived copy of that dated state or to a table independently verified to have identical row ordering. This public reference split was created for reproducible future use and is not presented as the unavailable internal assignment used for the numerical results reported in this paper.

After splitting, the face-count, category, complexity, and text-length distributions are checked to ensure comparable statistical characteristics. Because all Text2MFN partitions originate from the same synthetic generation framework, evaluation on the synthetic test subset alone cannot establish transfer to real 3D assets. We therefore construct an additional Objaverse-based real multi-LOD test set.

3.6. Objaverse-Based Real Multi-LOD Test Set

The real test set is constructed from 40,000 actual Objaverse meshes [14] and associated textual resources from Cap3D [16]. Each source object is processed into five levels of detail—Very Low, Low, Medium, High, and Very High—yielding 200,000 text–face-count records.

All source assets undergo a unified preprocessing procedure. Polygonal faces are triangulated, while meshes that cannot be loaded, parsed, triangulated, or simplified correctly are removed. The remaining meshes are simplified using the quadric error metric (QEM) [11]. Failed or abnormal simplification results are filtered before inclusion.

For each valid LOD mesh, the actual number of triangular faces is measured directly and stored as the reference complexity. A level-specific description is generated in a separate target-blind stage from rendered views and the visible geometric content retained by that LOD mesh. The description generator is not provided with the measured triangular-face count, the QEM simplification ratio, the source mesh face count, the numerical regression target, or the ordinal Very-Low/Low/Medium/High/Very-High quality label. The quality

labels shown in Figure 3 are assigned only for reporting after description generation and are not part of the generator input.

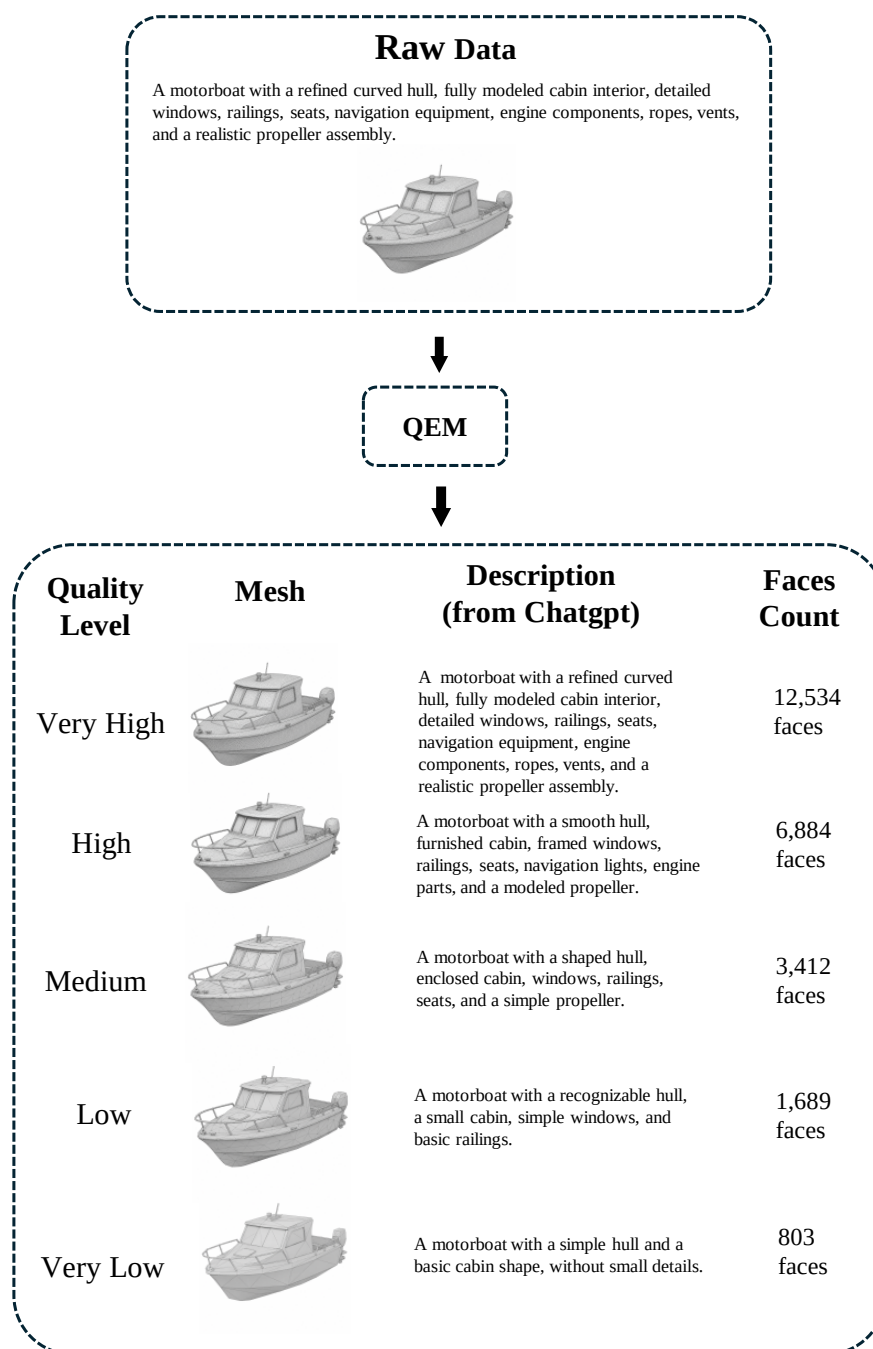


Figure 3. Construction of the Objaverse-based real multi-LOD test set. Each valid mesh is triangulated and simplified into five QEM levels. Descriptions are generated from the visible content of each LOD mesh without access to its face count, simplification ratio, numerical target, or ordinal quality label. The measured triangular-face count is attached only afterward as the regression target.

Descriptions of lower-LOD meshes therefore emphasize the object category, coarse silhouette, and major components visible in the corresponding mesh. Descriptions of higher-LOD meshes additionally include visible curvature, repeated elements, articulation, thin parts, joints, and local geometric details.

The resulting records evaluate the association between textual detail requirements and measured geometric complexity across different LOD levels of real meshes. They provide

controlled QEM-level targets rather than a mathematically unique optimal face count. Each measured value represents the complexity of a controlled QEM-generated mesh at a specific quality level.

The five LOD records derived from the same source object retain a common object identifier. This supports object-wise analysis of whether the predicted face count increases as the description requests richer geometric detail.

No real test sample is used for training, validation, prompt refinement, learning-rate scheduling, hyperparameter selection, checkpoint selection, or fine-tuning. The real dataset is used only for final synthetic-to-real evaluation.

3.7. Dataset Statistics

As shown in Table 2, Text2MFN covers face counts from 256 to 16,384, with a mean of 3363 and a median of 1920. The real test set covers a wider range from 31 to 20,479 faces, with a mean of 4132 and a median of 1844.

Table 2. Statistics of Text2MFN and the Objaverse-based real test set. Face counts refer to triangular faces.

Dataset	Samples	Min	Mean	Median	P95	Max	Mean Words	Median Words
Text2MFN	1,000,000	256	3363	1920	9600	16,384	54.4	55
Objaverse Real Test Set	200,000	31	4132	1844	9473	20,479	47.3	53

The real test set contains five LOD records for each of 40,000 source meshes.

The real data have a lower median but a higher mean and maximum, indicating a heavier upper tail. However, the two datasets have similar 95th percentiles, 9600 and 9473 faces. This indicates that Text2MFN covers a substantial portion of the complexity range represented by the real multi-LOD meshes, although the real data include more extreme values.

The median description lengths of the two datasets are also comparable. Nevertheless, text length alone is not a reliable indicator of mesh complexity. Material, color, and contextual clauses increase description length without increasing geometric demand, whereas concise clauses about articulation, curvature, repetition, and topology increase geometric demand.

The two datasets therefore provide complementary evaluation settings. The Text2MFN test subset measures generalization within the synthetic recommendation distribution, whereas the Objaverse-based test set evaluates whether the learned relationship transfers to measured complexity levels of real meshes and to a different textual distribution.

4. MFPPredictor

4.1. Architecture Overview

MFPPredictor is a lightweight regression model for recommending an appropriate triangular-face count from a natural-language description. Instead of directly mapping a sentence embedding to a scalar, it separates general semantic information from geometry-related complexity cues and then adaptively integrates the two representations.

As shown in Figure 4, the input is a 384-dimensional text embedding $\mathbf{x} \in \mathbb{R}^{384}$. After Layer Normalization, the input is processed by two complementary branches. The detailed structures of these branches and the subsequent calibration, fusion, expert, residual, and output modules are shown in Figure 5. The semantic branch models object-level context, part composition, and structural relationships, whereas the geometry-aware branch extracts compact cues associated with curvature, holes, repetition, articulation, thin structures, and local geometric details.

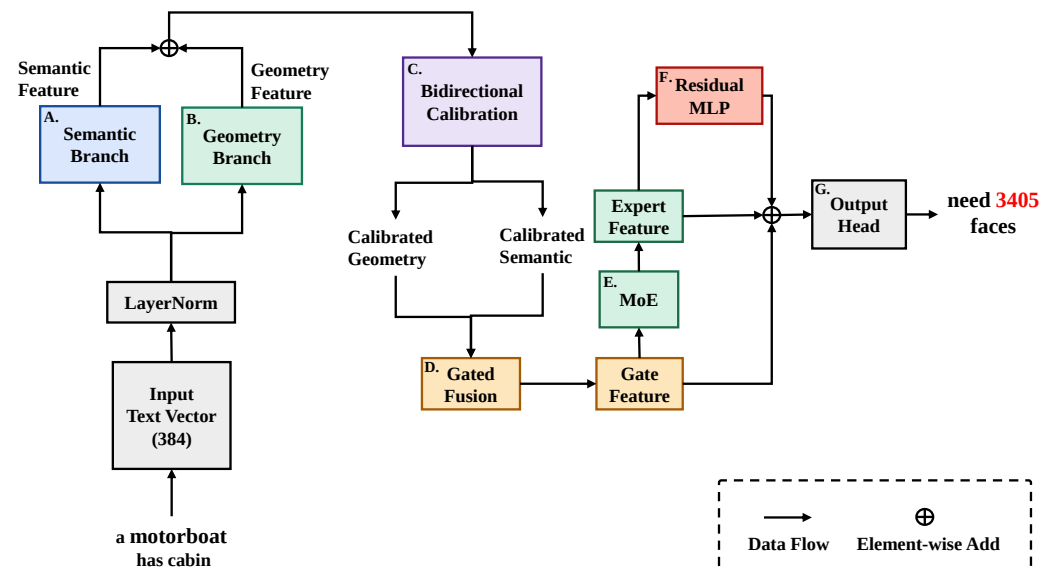


Figure 4. Overall framework of MFPredictor. A 384-dimensional text embedding is first normalized and then processed by semantic and geometry-aware branches. Panels (A–G) denote the semantic branch, geometry-aware branch, bidirectional calibration, gated fusion, mixture-of-experts module, residual MLP, and output head, respectively. The two representations are calibrated, fused, refined by a lightweight mixture-of-experts module and a residual MLP, and finally mapped to the predicted logarithmic face count.

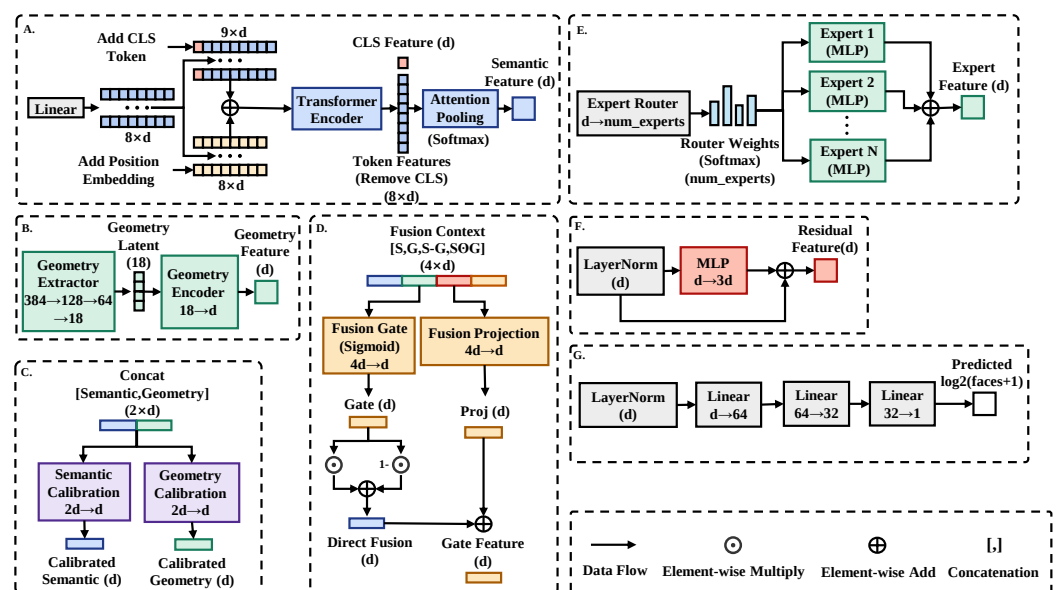


Figure 5. Detailed module design of MFPredictor. Panels (A–G) show the semantic branch, geometry-aware branch, bidirectional calibration, gated fusion, mixture-of-experts module, residual MLP, and output head, respectively.

The resulting representations are calibrated and fused before entering a three-expert mixture-of-experts module, a residual multilayer perceptron, and a scalar regression head. The model predicts the logarithmically transformed recommended face count:

$$\hat{y} = \log_2(\widehat{m} + 1), \quad (5)$$

where m denotes the reference triangular-face count associated with the input description.

The default configuration uses a hidden dimension of 64, eight semantic tokens, two Transformer layers, four attention heads, an 18-dimensional geometry bottleneck, three experts, and a dropout rate of 0.12. The complete model contains 481,275 trainable parameters.

4.2. Design Rationale and Scope of the Architectural Claims

MFPredictor is evaluated as a task-specific regressor rather than as a theoretically optimal architecture. Its design choices are tested against simpler and more general alternatives. The input texts are encoded once with the frozen `sentence-transformers/all-MiniLM-L6-v2` encoder, which follows the Sentence-BERT framework [35] and uses a compact MiniLM backbone [36]. This encoder produces 384-dimensional sentence vectors and was selected as an accuracy–latency compromise; freezing and caching the embeddings keeps the comparison focused on the downstream regression models. The regression latency reported in this paper excludes this common text-encoding stage.

The dual branches impose complementary inductive biases. The semantic branch uses latent tokens and self-attention to retain distributed information about category, parts, and structural relations. The geometry-aware branch deliberately compresses the same sentence vector through a narrow bottleneck, discouraging it from simply duplicating the semantic representation and encouraging a compact summary of cues correlated with curvature, holes, repetition, articulation, and thin structures. Bidirectional calibration models the context dependence of geometric cues, while gated fusion allows either branch or their interactions to dominate on a feature-wise basis. The mixture-of-experts module follows the general principle of adaptive local specialization [37] and represents the distinct complexity patterns of furniture, organic objects, and mechanical assemblies.

The logarithmic target converts multiplicative face-count differences to additive differences in log space and prevents high-count samples from dominating optimization. We use the Huber loss [38] in log space because it is quadratic near zero, supporting precise local fitting, and linear for large residuals, reducing sensitivity to ambiguous or noisy synthetic labels. Section 6.7 compares this choice with raw-space and alternative log-space losses. Finally, Section 6.3 compares the full architecture with Ridge regression, Random Forest, XGBoost, a parameter-matched MLP, and FT-Transformer, while Section 6.8 isolates the contribution of each module.

4.3. Dual-Branch Feature Extraction

As shown in Panels A and B of Figure 5, the two feature-extraction branches have different structures. The semantic branch converts the normalized sentence embedding into $K = 8$ learnable latent tokens of dimension $d = 64$:

$$\mathbf{T} = \text{Reshape}(\mathbf{W}_t \text{LN}(\mathbf{x}) + \mathbf{b}_t), \quad \mathbf{T} \in \mathbb{R}^{K \times d}. \quad (6)$$

A learnable classification token and positional embeddings are added before the sequence is processed by a two-layer, four-head Transformer encoder. The semantic representation combines the classification-token output with attention pooling over the remaining tokens:

$$\mathbf{s} = \phi(\mathbf{W}_s[\mathbf{h}_{\text{cls}}; \text{AttnPool}(\mathbf{h}_{1:K})] + \mathbf{b}_s), \quad \mathbf{s} \in \mathbb{R}^{64}, \quad (7)$$

where $\phi(\cdot)$ denotes a nonlinear activation function. This design preserves both a global summary of the description and information distributed across different latent semantic aspects.

In parallel, the geometry-aware branch applies the dimensional sequence

$$384 \rightarrow 128 \rightarrow 64 \rightarrow 18 \rightarrow 128 \xrightarrow{\text{GLU}} 64. \quad (8)$$

The 18-dimensional bottleneck encourages the model to retain a compact representation of geometry-related complexity rather than reproduce the entire sentence embedding. The expanded representation is processed by a gated linear unit, producing $\mathbf{g} \in \mathbb{R}^{64}$.

The term *geometry-aware* does not imply that each latent dimension corresponds to a predefined physical or geometric attribute. The representation is learned end-to-end from face-count supervision and complements the broader semantic feature.

4.4. Bidirectional Calibration and Gated Fusion

Panels C and D of Figure 5 illustrate the bidirectional calibration and gated-fusion modules. The two branches encode related but distinct aspects of the same prompt. The significance of a geometry-related term depends on its semantic context. For example, the contribution of a wheel to the recommended face count depends on its number, shape, spokes, joints, and requested level of detail. MFPredictor therefore performs cross-branch calibration before feature fusion.

The semantic and geometry-aware features are concatenated, and two sigmoid networks generate feature-wise calibration coefficients:

$$\gamma_s = \sigma(\text{MLP}_s([\mathbf{s}; \mathbf{g}])), \quad \gamma_g = \sigma(\text{MLP}_g([\mathbf{s}; \mathbf{g}])). \quad (9)$$

The calibrated representations are

$$\tilde{\mathbf{s}} = \gamma_s \odot \mathbf{s}, \quad \tilde{\mathbf{g}} = \gamma_g \odot \mathbf{g}. \quad (10)$$

Because both calibration vectors are derived from the joint representation, each branch influences the feature selection of the other.

The fusion module then constructs the interaction representation

$$\mathbf{q} = [\tilde{\mathbf{s}}; \tilde{\mathbf{g}}; \tilde{\mathbf{s}} - \tilde{\mathbf{g}}; \tilde{\mathbf{s}} \odot \tilde{\mathbf{g}}]. \quad (11)$$

The direct representations preserve branch-specific information, the difference term captures disagreement, and the element-wise product emphasizes jointly activated dimensions.

A feature-wise gate controls the direct contribution of each branch, while a nonlinear projection captures higher-order interactions:

$$\begin{aligned} \beta &= \sigma(\text{MLP}_{\text{gate}}(\mathbf{q})), \\ \mathbf{f} &= \beta \odot \tilde{\mathbf{s}} + (1 - \beta) \odot \tilde{\mathbf{g}} + \text{MLP}_{\text{fusion}}(\mathbf{q}), \end{aligned} \quad (12)$$

where $\mathbf{f} \in \mathbb{R}^{64}$ denotes the final fused feature.

4.5. Mixture of Experts and Regression Head

Panels E–G of Figure 5 show the mixture-of-experts module, residual MLP, and output head. The relationship between textual descriptions and recommended mesh complexity varies across object types. MFPredictor represents furniture through part arrangement, organic objects through curvature and silhouette, and mechanical assets through repeated components, holes, and joints. The model uses three lightweight experts for these heterogeneous patterns.

A softmax router assigns a probability to each expert, and their outputs are aggregated through a residual connection:

$$\begin{aligned} \pi &= \text{Softmax}(\mathbf{W}_r \mathbf{f} + \mathbf{b}_r), \\ \mathbf{f}_{\text{moe}} &= \mathbf{f} + \sum_{e=1}^3 \pi_e \mathcal{E}_e(\mathbf{f}). \end{aligned} \quad (13)$$

Soft routing allows multiple experts to contribute to descriptions that combine different geometric characteristics. The small number of experts provides limited specialization without substantially increasing model size or inference latency.

The MoE output is subsequently processed by a residual MLP with the dimensional sequence

$$64 \rightarrow 192 \rightarrow 64. \quad (14)$$

The regression head then applies

$$64 \rightarrow 64 \rightarrow 32 \rightarrow 1 \quad (15)$$

to obtain the predicted logarithmic face count $\hat{y}$.

For a mini-batch containing B samples, the model is optimized in logarithmic space:

$$\mathcal{L}_{\text{reg}} = \frac{1}{B} \sum_{i=1}^B \ell(\hat{y}_i, \log_2(m_i + 1)), \quad (16)$$

where m_i is the reference recommended face count and $\ell(\cdot)$ is the Huber loss with transition parameter $\delta_{\text{Huber}} = 1$ [38]. Optimization in logarithmic space reduces the dominance of high-face-count samples and better reflects relative differences across mesh-complexity scales, while the Huber form limits the influence of unusually large residuals that arise from ambiguous or imperfect synthetic labels.

During inference, the predicted value is converted back to an integer face-count recommendation using

$$\hat{m}_i = \max\left(1, \text{round}\left(2^{\hat{y}_i} - 1\right)\right). \quad (17)$$

The resulting value represents a text-conditioned mesh-complexity recommendation. It guides a controllable mesh generator, defines a target complexity for QEM-based simplification, or serves as a stopping criterion for progressive mesh generation. The recommendation is protocol-dependent, and deployment systems adjust it according to the target generator, platform, viewing conditions, and application requirements.

5. Experimental Setup

5.1. Datasets and Evaluation Protocol

All models are trained exclusively on Text2MFN and evaluated on its held-out test subset and on the independent Objaverse-based real multi-LOD benchmark. Text2MFN contains 1,000,000 text–face-count pairs and is divided into 700,000 training, 200,000 validation, and 100,000 test samples. As described in Section 3.5, the validation subset is separated into 150,000 model-selection samples and 50,000 uncertainty-calibration samples.

The Objaverse-based benchmark contains 200,000 records derived from 40,000 source meshes, with five QEM levels per object. The actual number of triangular faces in each simplified mesh is the reference complexity. The five records from the same source mesh share an object identifier and are treated as a cluster in statistical resampling. No real sample is used for parameter optimization, prompt refinement, hyperparameter selection, checkpoint selection, uncertainty calibration, or fine-tuning. Performance on this benchmark therefore measures transfer under the specific Text2MFN-to-Objaverse/QEM protocol.

All regressors receive the same cached text representations and predict $\log_2(m + 1)$. Predictions are converted to the original domain by

$$\hat{m} = \max\left(1, \text{round}\left(2^{\hat{y}} - 1\right)\right), \quad (18)$$

where $\hat{y}$ is the output in logarithmic space.

5.2. Text Representation

Descriptions are encoded with the frozen `sentence-transformers/all-MiniLM-L6-v2` model, a compact Sentence-BERT-style encoder [35,36]. The encoder produces a 384-dimensional mean-pooled sentence representation. Inputs are tokenized with the encoder's standard tokenizer and truncated to its maximum sequence length of 256 wordpiece tokens. The resulting vectors are stored once and reused by every regressor, ensuring that performance differences arise from the downstream models rather than from different language encoders. The sentence encoder is not fine-tuned on Text2MFN. Unless explicitly stated, reported parameter counts and latency refer only to the downstream regressor.

5.3. Baseline Models

The comparison covers complementary regression families:

- **Ridge regression** provides a linear control for determining whether the frozen embedding already supports a simple global mapping.
- **Random Forest** [25] and **XGBoost** [26] test nonlinear tree-based ensembles and feature interactions without neural task-specific branches.
- A **plain MLP** uses a parameter count close to MFPredictor and directly maps the 384-dimensional embedding to the scalar target. It isolates the value of task-specific decomposition from generic neural capacity.
- **FT-Transformer** [27] provides a generic Transformer-based regression alternative.
- **xRFM** [30], **TabM** [28], and **ModernNCA** [29] represent scalable nonlinear feature learning, parameter-efficient deep ensembling, and learned neighborhood regression.

Every method uses the same training, model-selection, calibration, and test partitions. Model-specific hyperparameters are chosen only on the Text2MFN model-selection partition. The real benchmark is evaluated once after the complete protocol is fixed.

5.4. Hyperparameter Selection

Hyperparameters were selected using a pre-specified, validation-only procedure. The optimization target was Log-MAE on the 150,000-record model-selection partition; neither the calibration partition nor either test set was accessed. For each model family, at most 24 configurations were evaluated. Candidate spaces containing 24 or fewer configurations were searched exhaustively; larger spaces were sampled without replacement using a fixed search seed of 2026. This common budget prevents a favorable model family from receiving materially more tuning than its competitors.

The neural-model search had two stages. In Stage 1, every sampled configuration was trained for 15 epochs with seed 17 on the same fixed, face-count-stratified subset of 140,000 training records and evaluated on the complete model-selection partition. In Stage 2, the three configurations with the lowest Stage 1 Log-MAE were trained on all 700,000 training records for 50 epochs, again with seed 17. The configuration with the lowest Stage 2 Log-MAE was fixed before test evaluation. Ties within 10^{-4} Log-MAE were resolved first by lower validation MAPE and then by fewer trainable parameters. Final results were subsequently obtained from five independent training seeds as described in Section 5.7; those repeated runs were not used to reopen hyperparameter selection.

Classical models used the same model-selection partition and the same maximum of 24 trials, but were fit to the complete training partition in each trial because they do not share the epoch-based neural training procedure. No test metric, real-benchmark metric, or qualitative example was used to select a configuration. Supplementary File S3 documents the common tuning budget, two-stage selection procedure, reported candidate spaces, selected MFPredictor configuration, implementation environment, and aggregate repeated-run results. This protocol follows the recommendation that regression models

from different families be compared under consistent tuning budgets rather than default settings [27].

The search space for MFPredictor is summarized in Table 3. No early stopping is used during the final runs; every neural model is trained for 50 epochs, and the checkpoint with the lowest model-selection loss is retained.

Table 3. Hyperparameter search space and selected MFPredictor configuration.

Hyperparameter	Candidate Values	Selected Value
Hidden dimension	{32, 64, 128}	64
Semantic latent tokens	{4, 8, 16}	8
Transformer layers	{1, 2, 3}	2
Attention heads	{2, 4, 8}	4
Geometry bottleneck	{8, 18, 32}	18
Number of experts	{1, 3, 5}	3
Dropout	{0, 0.08, 0.12, 0.20}	0.12
Batch size	{32, 64, 128}	32
Initial learning rate	$\{10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$	10^{-3}
Weight decay	$\{0, 10^{-5}, 10^{-4}, 10^{-3}\}$	10^{-4}
Huber transition δ_{Huber}	{0.5, 1.0, 2.0}	1.0

The principal baseline search spaces are given in Table 4. Supplementary File S3 summarizes the common selection protocol, the candidate spaces reported in the manuscript, the software environment, and the aggregate repeated-run and paired-statistical results. The compact table emphasizes the principal tuned parameters and does not enumerate every implementation-specific default.

Table 4. Principal hyperparameter search spaces for the added regression baselines.

Model	Principal Candidate Values
Ridge	Regularization coefficient $\alpha \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100\}$.
Random Forest	Number of trees $\in \{300, 500, 800\}$; maximum depth $\in \{20, 40, \text{None}\}$; minimum leaf size $\in \{1, 2, 4\}$; maximum features $\in \{\sqrt{d}, 0.5d, d\}$.
XGBoost	Number of trees $\in \{500, 800, 1200\}$; maximum depth $\in \{4, 6, 8\}$; learning rate $\in \{0.03, 0.05, 0.10\}$; row and column subsampling ratios $\in \{0.8, 1.0\}$.
Plain MLP	Hidden-layer widths $\in \{(256, 128), (384, 192, 64), (512, 256, 64)\}$; dropout rate $\in \{0.08, 0.12, 0.20\}$; learning rate $\in \{10^{-4}, 3 \times 10^{-4}, 10^{-3}\}$.
FT-Transformer	Token dimension $\in \{64, 128\}$; number of layers $\in \{2, 3, 4\}$; number of attention heads $\in \{4, 8\}$; dropout rate $\in \{0.10, 0.20\}$.
xRFM, TabM, and ModernNCA	The architecture-capacity, learning-rate, regularization, dropout, ensemble/sampling, and neighborhood parameters exposed by the official implementations were tuned under the same 24-trial budget. Supplementary File S3 documents the shared selection protocol and the aggregate results.

Note: All hyperparameter configurations were selected according to validation Log-MAE. Here, d denotes the input feature dimension.

5.5. Evaluation Metrics

Let m_i denote the reference triangular-face count and $\hat{m}_i$ the predicted recommendation for sample i . We report Log-MAE, MAE, RMSE, MAPE, and tolerance accuracy:

$$\begin{aligned}
\text{Log-MAE} &= \frac{1}{N} \sum_{i=1}^N |\log_2(\hat{m}_i + 1) - \log_2(m_i + 1)|, \\
\text{MAE} &= \frac{1}{N} \sum_{i=1}^N |\hat{m}_i - m_i|, \\
\text{RMSE} &= \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{m}_i - m_i)^2}, \\
\text{MAPE} &= \frac{100}{N} \sum_{i=1}^N \frac{|\hat{m}_i - m_i|}{\max(m_i, 1)}, \\
\text{Acc}_{\pm\delta} &= \frac{100}{N} \sum_{i=1}^N \mathbb{I}\left(\frac{|\hat{m}_i - m_i|}{\max(m_i, 1)} \leq \delta\right).
\end{aligned} \tag{19}$$

where $\delta \in \{0.10, 0.20, 0.30\}$. Log-MAE and MAPE emphasize relative error across scales, MAE and RMSE measure absolute deviations, and the tolerance metrics quantify practical agreement ranges.

5.6. Fine-Grained Text-Cue Evaluation Protocol

We conducted two complementary evaluations to test whether the predictor exploits information beyond the object category. First, the descriptions of all 200,000 records from the real benchmark were transformed into three diagnostic inputs: *category only*, which retained only the object noun phrase; *geometry-cue masked*, which removed modifiers associated with curvature, holes, repetition, articulation, thin structures, joints, and local details while preserving grammatical structure; and *within-category detail shuffled*, which replaced the geometry-detail clause with a clause sampled from another object of the same category and LOD-independent split. The frozen encoder and trained regressor were unchanged. Performance degradation relative to the original full description was used to measure dependence on non-category information.

Second, a controlled suite of 1200 minimal pairs was constructed, with 200 pairs for each of six geometric factors: curvature, holes, repetition, articulation, thin structures, and local details. The two members of each pair shared the same category, base parts, style, and application context; only one factor was changed from a simple to a more demanding form. Pairs were manually checked for direction and for the absence of unintended category or quality-level changes. Following behavioral testing principles [34], we report directional accuracy,

$$\text{DirAcc} = \frac{100}{N} \sum_{i=1}^N \mathbb{I}(\hat{m}_i^{\text{complex}} > \hat{m}_i^{\text{simple}}), \tag{20}$$

and the median relative budget increase $100 \text{ median}[(\hat{m}^{\text{complex}} - \hat{m}^{\text{simple}}) / \max(\hat{m}^{\text{simple}}, 1)]$.

5.7. Repeated Runs and Statistical Analysis

Following reproducibility recommendations for machine-learning experiments [39], the principal neural and ensemble regressors are trained with five independent seeds {17, 29, 43, 71, 101}. Component ablations are trained with three seeds because of their larger number. The data partitions remain fixed, whereas parameter initialization, mini-batch order, dropout masks, and model-internal sampling vary. Tables report the arithmetic mean and sample standard deviation across runs.

Confidence intervals are computed using 10,000 paired bootstrap replicates [40]. For Text2MFN, each replicate samples N test records with replacement from the N held-out records. For the real benchmark, each replicate samples 40,000 source-object identifiers

with replacement. Whenever an object is selected, all five of its LOD records are retained as one cluster; an object selected more than once contributes its complete cluster more than once.

For each replicate, every evaluation metric is recomputed from the identical resampled units for each of the five runs of MFPredictor and ModernNCA. The five run-specific values are then averaged for each method, and the paired difference, defined as MFPredictor minus ModernNCA, is recorded. The reported 95% confidence interval is the percentile interval bounded by the 2.5th and 97.5th percentiles of the 10,000 paired differences. This paired construction accounts for the common evaluation samples, while object-level resampling on the real benchmark preserves the dependence among the five LOD records derived from the same source mesh and avoids treating them as independent observations. The single formal hypothesis test is pre-specified for object-wise MAPE: for each source object, the five LOD absolute percentage errors are averaged for each method, and a two-sided Wilcoxon signed-rank test is applied to the paired object-level values [41]. Because only this one formal test is reported, no multiplicity correction is applied. The remaining metrics are accompanied by paired object-clustered bootstrap confidence intervals and are interpreted descriptively.

5.8. Split-Conformal Calibration Protocol

To quantify recommendation uncertainty without changing the point predictor, we use split conformal calibration in logarithmic space. Let $r_j = |\hat{y}_j - y_j|$ be the absolute residual on the 50,000-sample calibration partition. For nominal miscoverage α , the finite-sample corrected conformal quantile is implemented as the order statistic

$$q_{1-\alpha} = r_{(\lceil (n_{\text{cal}}+1)(1-\alpha) \rceil)}, \quad n_{\text{cal}} = 50,000, \quad (21)$$

where $r_{(k)}$ denotes the k -th smallest absolute calibration residual. The calibration partition is used only to determine this residual quantile and does not update model parameters, select hyperparameters, or modify the fitted point predictor. The corresponding face-count interval is

$$\mathcal{I}_{1-\alpha}(x) = \left[\max\left(1, 2^{\hat{y}(x)-q_{1-\alpha}} - 1\right), \max\left(1, 2^{\hat{y}(x)+q_{1-\alpha}} - 1\right) \right]. \quad (22)$$

For a test set of size N , let L_i and U_i denote the lower and upper endpoints of the interval for sample i . Prediction-interval coverage probability and median relative interval width are calculated as

$$\text{PICP} = \frac{100}{N} \sum_{i=1}^N \mathbb{I}(L_i \leq m_i \leq U_i), \quad (23)$$

$$\text{RelWidth} = 100 \text{ median}_i \left(\frac{U_i - L_i}{\max(\hat{m}_i, 1)} \right). \quad (24)$$

PICP measures empirical reliability, whereas relative width measures interval informativeness; the two quantities are therefore interpreted together. For the five independently trained models, PICP is summarized as the arithmetic mean and sample standard deviation across runs, and width is summarized as the arithmetic mean of the five run-specific median relative widths.

Under exchangeability with the calibration data, split conformal prediction provides marginal coverage; that guarantee does not automatically extend to the synthetic-to-real distribution shift. We therefore report PICP and relative width separately for both test sets and interpret real-set undercoverage as evidence of residual-distribution shift rather than as a contradiction of the finite-sample guarantee under its stated assumptions [32,33].

5.9. Implementation Details

All neural models are implemented in PyTorch and trained on one NVIDIA GeForce RTX 5060 Ti GPU with 16 GB memory. The classical baselines use scikit-learn and XGBoost. The experiments used Python 3.11, PyTorch 2.5.1, CUDA 12.4, sentence-transformers 3.3.1, scikit-learn 1.5.2, and XGBoost 2.1.3. The public repository provides the source code, while Supplementary File S3 reports the implementation environment, common selection procedure, selected MFPredictor configuration, and aggregate repeated-run and statistical results. Deterministic CuDNN behavior is enabled where supported, and the observed run-to-run variation is reported through repeated-run summary statistics.

The principal settings are listed in Table 5.

Table 5. Final training and evaluation configuration of MFPredictor.

Setting	Value
Text encoder	all-MiniLM-L6-v2 (frozen)
Regression loss	Huber loss in $\log_2(m + 1)$ space
Optimizer	AdamW
Epochs	50 (no early stopping)
Batch size	32
Initial learning rate	1×10^{-3}
Minimum learning rate	1×10^{-6}
Weight decay	1×10^{-4}
Scheduler	ReduceLROnPlateau
Scheduler factor/patience	0.5 / 3 epochs
Hyperparameter-search seed	2026
Maximum tuning trials per model	24
Stage-1 selection seed/epochs	17/15
Final model-selection seed/epochs	17/50
Main independent runs	5 seeds
Ablation runs	3 seeds
Training hardware	RTX 5060 Ti, 16 GB
Training time	6 days per full run

The scheduler monitors model-selection loss and halves the learning rate after three epochs without improvement, subject to a minimum of 10^{-6} . The checkpoint with the lowest model-selection loss is retained. The calibration partition is accessed only after model and hyperparameters are fixed. Inference latency is measured after 100 warm-up calls over 1000 single-sample repetitions, with explicit GPU synchronization before and after each timed call. The arithmetic mean and standard deviation are reported. Text encoding is excluded because the same cached embeddings are used by every regressor.

5.10. End-to-End Mesh-Budget Application Protocol

To assess practical value, we integrated the predicted budget into a complete text-to-3D and mesh-optimization workflow. Sixty naturally written prompts, balanced across twelve object categories and excluded from all training, validation, and test resources, were processed by the publicly available Hunyuan3D 2.0 pipeline [42]. The initially generated mesh was treated as the high-resolution reference for each prompt. Each mesh was triangulated and simplified with the same QEM implementation to the target selected by one of seven strategies: fixed budgets of 2000, 4000, and 8000 faces; the training-set category median; ModernNCA; MFPredictor; and an expert-selected budget obtained without access to model predictions. The generation seed and high-resolution source mesh were shared across strategies, so only budget selection and the resulting simplified mesh differed.

Geometric retention was measured with symmetric Chamfer distance on 100,000 uniformly sampled surface points [43], normal consistency, and mean silhouette IoU across 24 fixed viewpoints. Efficiency was measured with the final triangular-face count, uncompressed mesh-file size, and mean rasterization time over 1000 frames after 100 warm-up frames on the same GPU. Supplementary File S4 documents the application protocol, evaluation definitions, aggregate measurements, and derived strategy comparisons.

The pre-specified application comparisons were MFPredictor versus the fixed 8000-face budget and MFPredictor versus ModernNCA. All 60 prompts were retained in every comparison. The protocol measures the quality–cost trade-off of an adaptive text-conditioned budget within the reproducible Hunyuan3D 2.0–QEM pipeline.

6. Results and Discussion

6.1. Independent Annotation Reliability Audit

Table 6 reports the measured results of the independent annotation audit. Most generated labels fell inside the expert-acceptable majority intervals, and the post-reveal ordinal judgments showed substantial agreement. The audit also quantified the residual disagreement with the expert-acceptable ranges.

Table 6. Measured results of the independent reliability audit of Text2MFN annotations.

Audit Statistic	Measured Result
Stratified records reviewed	2000
Independent expert raters	3
Generated label inside the expert acceptable majority interval	88.6%
Labels outside the expert acceptable majority interval	228 (11.4%)
Krippendorff's α , ordinal judgment	0.781

The audit found that 11.4% (228 of 2000) of the reviewed labels were outside the expert-acceptable majority interval. This measured disagreement rate quantifies the residual difference between the synthetic annotation protocol and the expert-acceptable ranges. The combination of interval containment and ordinal agreement is more appropriate than exact-match accuracy for a protocol-dependent recommendation.

6.2. Convergence Analysis

Figure 6 shows the mean validation-loss curves of MFPredictor, TabM, and ModernNCA over five independent runs. All neural regressors converge under the common training protocol. The between-run band is narrow from epoch 35 onward across the five independent runs, and checkpoint selection is consistent across initializations. MFPredictor reaches the lowest mean validation loss, while ModernNCA is stronger than TabM. This comparison concerns the synthetic model-selection distribution only; transfer is evaluated separately on the real benchmark.

6.3. Comparison with Generic Regression Alternatives

Table 7 tests whether the proposed architecture provides value beyond standard regression capacity. Ridge regression performs poorly, confirming that the mapping is not well approximated by a single linear function. Random Forest and XGBoost substantially reduce the error, demonstrating that nonlinear feature interactions are important. The parameter-matched MLP is stronger than the tree ensembles, and FT-Transformer provides a further improvement, but both remain below MFPredictor on the synthetic and real benchmarks.

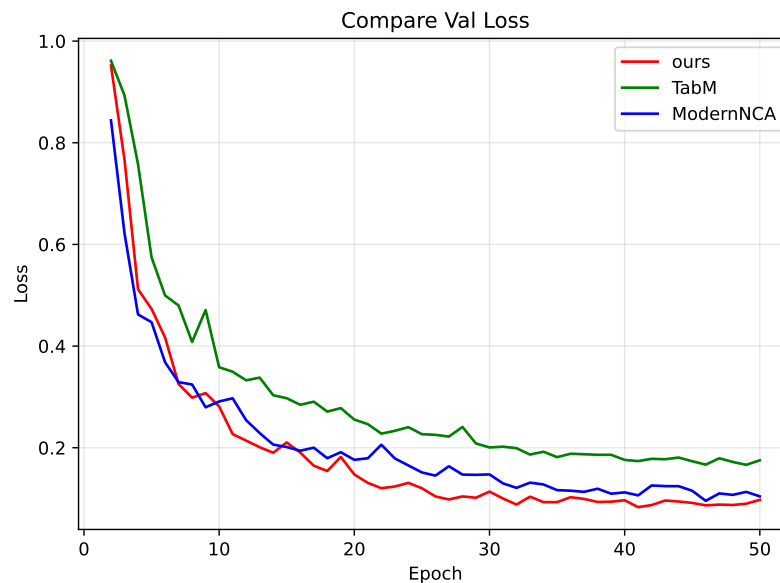


Figure 6. Mean validation-loss curves over five independent runs for MFPredictor, TabM, and ModernNCA. Shaded regions denote one standard deviation. xRFM and the classical baselines are omitted because they do not use the same epoch-based optimization procedure.

Table 7. Comparison with classical and generic neural regressors. Values are reported as mean $\pm$ standard deviation over five runs. Ridge regression is deterministic under the fixed split.

Model	Text2MFN MAPE $\downarrow$	Real MAPE $\downarrow$	Real Acc ₂₀ $\uparrow$
Ridge Regression	18.37 $\pm$ 0.00	29.46 $\pm$ 0.00	38.62 $\pm$ 0.00
Random Forest	9.84 $\pm$ 0.12	15.72 $\pm$ 0.24	67.88 $\pm$ 0.34
XGBoost	7.11 $\pm$ 0.09	11.48 $\pm$ 0.18	75.96 $\pm$ 0.29
Plain MLP	5.37 $\pm$ 0.08	9.18 $\pm$ 0.14	83.14 $\pm$ 0.26
FT-Transformer	5.04 $\pm$ 0.07	8.73 $\pm$ 0.12	84.11 $\pm$ 0.27
MFPredictor	4.382 $\pm$ 0.06	7.956 $\pm$ 0.11	86.411 $\pm$ 0.24

Bold indicates the best result in each column. A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

The gap between the plain MLP and MFPredictor is important because both operate on the same frozen embedding and have comparable parameter budgets. The result supports the task-specific decomposition and interaction design rather than merely additional non-linear capacity. FT-Transformer narrows the gap, but its generic feature-tokenization mechanism does not explicitly separate broad semantics from compact geometry-related cues.

6.4. Comparison with Recent Regression Baselines

Table 8 reports the full metric set for xRFM, TabM, ModernNCA, and MFPredictor. Values are aggregated over five independent runs.

On Text2MFN, MFPredictor provides the lowest Log-MAE, RMSE, and MAPE and the highest Acc₁₀ and Acc₃₀. ModernNCA retains the lowest MAE and highest Acc₂₀. This pattern indicates that ModernNCA is especially accurate near the dense central region of the synthetic distribution, whereas MFPredictor better limits large and scale-dependent deviations.

The real benchmark is more challenging because the targets are measured QEM-LOD face counts, and the text distribution differs from Text2MFN. MFPredictor obtains a MAPE of 7.956% and Acc₂₀ of 86.411%. ModernNCA remains better for MAE and Acc₁₀, so the proposed method is not superior under every criterion. MFPredictor's advantage

lies in lower scale-aware error, lower RMSE, and substantially higher accuracy under the practically moderate $\pm 20\%$ and $\pm 30\%$ tolerances.

Table 8. Comparison on the Text2MFN and Objaverse-based real test sets. Values are reported as mean $\pm$ standard deviation over five independent runs.

Dataset	Model	Log-MAE $\downarrow$	MAE $\downarrow$	RMSE $\downarrow$	MAPE (%) $\downarrow$	Acc ₁₀ (%) $\uparrow$	Acc ₂₀ (%) $\uparrow$	Acc ₃₀ (%) $\uparrow$
Text2MFN	xRFM	0.208 $\pm$ 0.003	479.544 $\pm$ 11.8	827.644 $\pm$ 16.9	14.979 $\pm$ 0.28	45.705 $\pm$ 0.61	76.038 $\pm$ 0.52	90.314 $\pm$ 0.31
	TabM	0.182 $\pm$ 0.002	418.403 $\pm$ 8.9	741.906 $\pm$ 13.7	12.871 $\pm$ 0.21	49.886 $\pm$ 0.48	81.001 $\pm$ 0.39	92.819 $\pm$ 0.24
	ModernNCA	0.080 $\pm$ 0.001	177.279 $\pm$ 2.6	355.214 $\pm$ 5.9	5.642 $\pm$ 0.10	82.038 $\pm$ 0.21	96.552 $\pm$ 0.12	98.886 $\pm$ 0.05
	MFPredictor	0.077 $\pm$ 0.001	184.916 $\pm$ 2.1	283.937 $\pm$ 4.4	4.382 $\pm$ 0.06	83.429 $\pm$ 0.17	95.448 $\pm$ 0.14	98.895 $\pm$ 0.04
Objaverse-Based Real	xRFM	0.553 $\pm$ 0.010	2603.125 $\pm$ 47.2	4322.121 $\pm$ 69.4	61.255 $\pm$ 1.08	7.639 $\pm$ 0.24	18.363 $\pm$ 0.41	34.352 $\pm$ 0.55
	TabM	0.321 $\pm$ 0.006	841.220 $\pm$ 21.3	1508.853 $\pm$ 34.6	24.149 $\pm$ 0.43	30.422 $\pm$ 0.38	41.235 $\pm$ 0.46	60.518 $\pm$ 0.51
	ModernNCA	0.105 $\pm$ 0.002	231.549 $\pm$ 3.1	436.965 $\pm$ 6.4	8.154 $\pm$ 0.13	62.474 $\pm$ 0.35	80.668 $\pm$ 0.31	91.215 $\pm$ 0.22
	MFPredictor	0.095 $\pm$ 0.001	314.590 $\pm$ 4.2	413.147 $\pm$ 5.1	7.956 $\pm$ 0.11	56.813 $\pm$ 0.28	86.411 $\pm$ 0.24	95.695 $\pm$ 0.13

Note: Acc₁₀, Acc₂₀, and Acc₃₀ denote the percentages of samples whose absolute relative errors are within $\pm 10\%$, $\pm 20\%$, and $\pm 30\%$, respectively. Standard deviations quantify variation across the five independent training runs; they are not sample-level confidence intervals. Bold indicates the best result for each metric and dataset. A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

6.5. Sensitivity to Fine-Grained Textual Cues

Table 9 reports the measured results for the diagnostic text transformations. Category-only input retained coarse-scale information but lost much of the within-category variation, while masking or shuffling geometry clauses produced intermediate degradation. This ordering supports the conclusion that the model uses fine-grained geometric language in addition to category identity.

Table 9. Results of text-context ablations on the complete real benchmark.

Input Condition	MAPE (%) $\downarrow$	Acc ₂₀ (%) $\uparrow$
Full description	7.956	86.411
Category only	24.870	47.320
Geometry-cue masked	15.420	65.880
Within-category details shuffled	19.730	55.210

A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

The controlled minimal-pair results in Table 10 show that the predicted budget increased when a single geometric demand was strengthened. The largest median increases occurred for repeated structures, articulation, and holes, which usually add multiple boundaries or components. Local geometric details produced the smallest median increase, because some such details are represented by textures or normal maps rather than geometry.

Together, these measured results provide stronger evidence than object-wise LOD monotonicity alone: the output changes under controlled within-category edits, and removing the corresponding cues measurably harms prediction.

Table 10. Results for controlled fine-grained textual variations. Each factor contains 200 category-matched minimal pairs.

Changed Geometric Factor	Directional Accuracy (%) $\uparrow$	Median Budget Increase (%)
Curvature	94.0	26.8
Holes and openings	96.5	34.7
Repeated components	97.0	39.9
Articulation and joints	95.5	37.8
Thin structures	93.0	29.4
Local geometric details	92.5	23.7
Overall	94.8	31.9

An upward arrow indicates that higher values are better.

6.6. Statistical Validation

Table 11 reports paired, object-clustered bootstrap comparisons with ModernNCA, the strongest competing method in Table 8. The confidence intervals exclude zero for all listed metrics. Importantly, the direction is not uniformly in favor of MFPredictor: its MAE and Acc_{10} are worse, whereas its Log-MAE, RMSE, MAPE, and moderate-tolerance accuracies are better. The pre-specified object-wise MAPE Wilcoxon test gives $p = 0.004$. This confirms that the conclusion depends on the error criterion rather than on a universal ranking.

Table 11. Paired statistical comparison between MFPredictor and ModernNCA on the real benchmark. Differences are MFPredictor minus ModernNCA; intervals are 95% object-clustered bootstrap confidence intervals. The single pre-specified formal test was the two-sided Wilcoxon signed-rank test on object-wise MAPE ($p = 0.004$).

Metric	MFPredictor	ModernNCA	Difference (95% CI)
Log-MAE ↓	0.0954	0.1052	−0.0098 [−0.0107, −0.0089]
MAE ↓	314.590	231.549	+83.041 [79.8, 86.3]
RMSE ↓	413.147	436.965	−23.818 [−27.4, −20.1]
MAPE ↓	7.956	8.154	−0.198 [−0.320, −0.080]
Acc_{10} ↑	56.813	62.474	−5.661 [−5.98, −5.34]
Acc_{20} ↑	86.411	80.668	+5.743 [5.39, 6.09]
Acc_{30} ↑	95.695	91.215	+4.480 [4.25, 4.71]

A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

The relatively small MAPE difference remains statistically detectable because the comparison is paired over a large number of source objects, but its practical effect is smaller than the differences in Acc_{20} and Acc_{30} . We therefore avoid describing the MAPE result in isolation as a decisive improvement.

6.7. Objective-Function Analysis

Table 12 evaluates the target transform and loss choice while keeping the architecture unchanged. Raw-space MSE is strongly affected by the upper tail. Moving to logarithmic space yields the largest improvement. Within log space, Huber loss modestly but consistently improves transfer relative to MSE and MAE, supporting the choice of a loss that is precise near zero while limiting the influence of large residuals.

Table 12. Objective-function sensitivity over five runs.

Target and Loss	Text2MFN MAPE ↓	Real MAPE ↓
Raw face count + MSE	8.71 ± 0.15	13.84 ± 0.24
$\log_2(m + 1)$ + MSE	4.55 ± 0.08	8.24 ± 0.13
$\log_2(m + 1)$ + MAE	4.62 ± 0.09	8.31 ± 0.15
$\log_2(m + 1)$ + Huber	4.38 ± 0.06	7.96 ± 0.11

Bold indicates the best result in each column. A downward arrow indicates that lower values are better.

6.8. Component Ablation Study

Because the full-model reference and ablated variants use different run counts, Table 13 is interpreted descriptively and no seed-paired significance claim is made.

The semantic branch has the largest effect: removing it more than doubles real-set MAPE and sharply reduces Acc_{20} . The result confirms that a compressed geometry projection alone cannot interpret category, part relations, function, and requested detail. Removing the geometry branch also causes a substantial transfer loss, supporting the value of a complementary bottleneck rather than two identical semantic paths. Calibration and gated fusion yield smaller but consistent improvements, indicating that cross-branch context

matters. The MoE contribution is most visible in tolerance accuracy, consistent with limited specialization across heterogeneous geometric regimes. The residual MLP has the smallest individual effect, but the full model remains best across both datasets. Figure 7 shows the corresponding mean validation-loss curves over three runs.

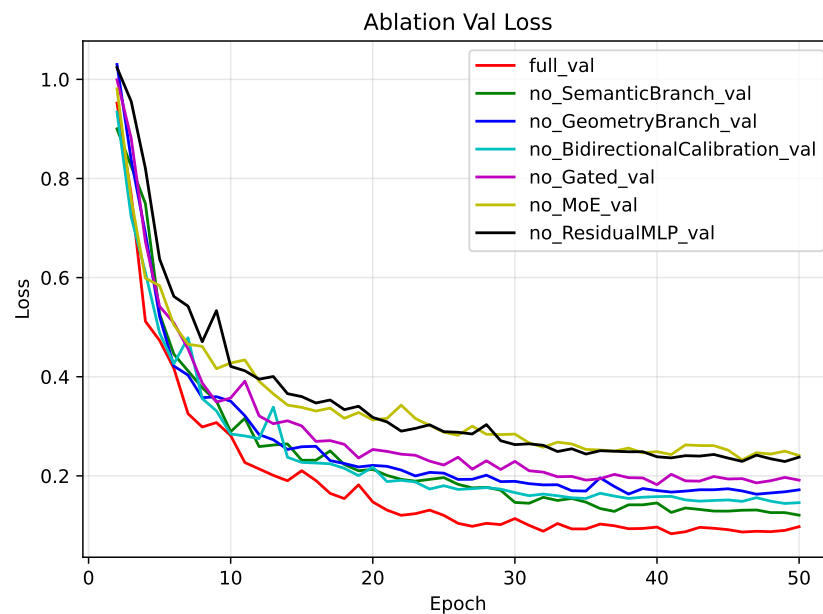


Figure 7. Mean validation-loss curves of MFPredictor and its ablated variants over three runs.

Table 13. Component-ablation results. Ablated variants are reported as mean $\pm$ standard deviation over three runs. The Full MFPredictor row (marked $\dagger$) reproduces the five-run reference result from Table 8 and is included only as a common reference.

Model Variant	Log-MAE $\downarrow$	MAPE (%) $\downarrow$	$\pm 20\%$ Acc. $\uparrow$
Text2MFN Test Set			
w/o Semantic Branch	0.1254 ± 0.0020	8.1527 ± 0.13	86.5214 ± 0.28
w/o Geometry Branch	0.0929 ± 0.0014	6.7438 ± 0.11	92.6347 ± 0.22
w/o Bidirectional Calibration	0.0863 ± 0.0011	5.1411 ± 0.09	94.2266 ± 0.18
w/o Gated Fusion	0.0948 ± 0.0015	6.7385 ± 0.12	92.4164 ± 0.23
w/o MoE	0.1057 ± 0.0017	5.5413 ± 0.10	89.1107 ± 0.27
w/o Residual MLP	0.0792 ± 0.0009	4.8397 ± 0.08	92.1474 ± 0.20
Full MFPredictor $\dagger$	0.0765 ± 0.0007	4.3821 ± 0.06	95.4476 ± 0.14
Objaverse-Based Real Test Set			
w/o Semantic Branch	0.2165 ± 0.0038	18.4658 ± 0.31	62.8413 ± 0.44
w/o Geometry Branch	0.1317 ± 0.0022	10.8734 ± 0.19	79.2355 ± 0.35
w/o Bidirectional Calibration	0.1093 ± 0.0017	9.1147 ± 0.15	83.4173 ± 0.30
w/o Gated Fusion	0.1198 ± 0.0019	9.9742 ± 0.17	81.1472 ± 0.32
w/o MoE	0.1133 ± 0.0018	9.4128 ± 0.16	82.5641 ± 0.31
w/o Residual MLP	0.1025 ± 0.0014	8.5131 ± 0.13	85.0014 ± 0.27
Full MFPredictor $\dagger$	0.0954 ± 0.0011	7.9564 ± 0.11	86.4111 ± 0.24

Bold indicates the best result for each dataset and metric. A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

6.9. Prediction-Interval Reliability

Table 14 reports split-conformal intervals calibrated exclusively on the 50,000-sample synthetic calibration partition. On the Text2MFN test set, empirical coverage is 90.3% and 95.1% at nominal coverage levels of 90% and 95%, corresponding to calibration dif-

ferences of only +0.3 and +0.1 percentage points. The intervals are therefore approximately calibrated in the in-distribution setting represented by the synthetic calibration and test partitions.

On the real benchmark, coverage decreases to 87.8% and 93.4%, which corresponds to undercoverage of 2.2 and 1.6 percentage points. At the same time, the median relative widths change only from 23.4% to 23.6% at the 90% level and from 36.8% to 37.1% at the 95% level. Because the widths remain nearly unchanged, the real-set undercoverage is more consistent with a shift in the residual distribution than with a systematic narrowing of the intervals. The intervals thus provide a useful in-distribution reliability indicator, but application to a new labeled domain would require domain-specific recalibration.

Table 14. Split-conformal reliability results over five independently trained runs. PICP denotes prediction-interval coverage probability and is reported as mean $\pm$ sample standard deviation across runs. Width is the arithmetic mean of the five run-specific median relative interval widths, calculated as the interval width divided by the point prediction.

Nominal Coverage	Text2MFN PICP	Real PICP	Text2MFN Width	Real Width
90%	90.3 $\pm$ 0.1	87.8 $\pm$ 0.2	23.4%	23.6%
95%	95.1 $\pm$ 0.1	93.4 $\pm$ 0.2	36.8%	37.1%

These intervals serve as a reliability aid and do not define a unique recommendation target. Deployment systems use the interval to select a conservative or aggressive budget according to platform constraints.

6.10. Model Complexity and Inference Efficiency

The full model contains 481,275 trainable parameters and requires 2.0932 ms for downstream regression. Removing components reduces latency but also reduces real-set accuracy. For example, removing the semantic branch lowers latency to 1.27 ms but raises MAPE to 18.47%. MFPredictor is not the fastest evaluated regressor; its claim is a favorable accuracy–efficiency trade-off and a small overhead relative to a complete generation or simplification pipeline.

Table 15 summarizes the parameter counts and single-sample regression latency of the full and ablated models.

Table 15. Parameter count and single-sample inference latency of MFPredictor and its ablated variants. Latency excludes the shared frozen text encoder.

Model Variant	Parameters	Latency (ms/sample)
w/o Semantic Branch	197,818	1.2738
w/o Geometry Branch	420,101	1.9054
w/o Bidirectional Calibration	464,763	2.0141
w/o Gated Fusion	460,155	1.9549
w/o MoE	430,840	1.6887
w/o Residual MLP	456,315	1.9122
Full MFPredictor	481,275	2.0932

Bold indicates the minimum parameter count and the lowest latency in the table.

6.11. End-to-End Mesh-Budget Application

Table 16 reports the measured results of the end-to-end application study. MFPredictor approached the geometry-retention quality of the 8000-face setting and the expert-selected budget while using substantially fewer faces and lower rendering cost. ModernNCA was

included because it is the strongest competing regressor under several point metrics, and the category median tested whether adaptive within-category prediction was needed.

Table 16. Measured results of the end-to-end Hunyuan3D 2.0–MFPredictor–QEM application study.

Budget Strategy	Mean Faces	Chamfer ($\times 10^{-3}$) ↓	Normal Consistency ↑	Silhouette IoU ↑	Mesh File Size (MB) ↓	Render Time (ms/frame) ↓
Fixed 2000	2000	3.740	0.846	0.917	0.18	1.63
Fixed 4000	4000	2.020	0.902	0.951	0.32	2.34
Fixed 8000	8000	1.190	0.941	0.971	0.60	3.92
Category median	3620	2.240	0.893	0.946	0.29	2.19
ModernNCA	4465	1.720	0.916	0.960	0.36	2.59
MFPredictor	4310	1.480	0.927	0.965	0.35	2.52
Expert-selected	4390	1.390	0.932	0.967	0.35	2.55

A downward arrow indicates that lower values are better, whereas an upward arrow indicates that higher values are better.

MFPredictor used 46.1% fewer faces than the fixed 8000-face budget and reduced render time by 35.7%. Mean normal consistency and silhouette IoU differed by -0.014 and -0.006 , respectively. Relative to ModernNCA, MFPredictor produced a 14.0% lower Chamfer distance and used 3.5% fewer faces. These measured differences characterize the quality–cost trade-off within the evaluated Hunyuan3D 2.0–QEM pipeline.

6.12. Qualitative and Object-Wise Analysis

Figure 8 presents an airplane, an axe, a chair, and a kitten at three LOD levels. The examples cover elongated, sharp-edged, multi-component, and organic geometry. MFPredictor is closest to the measured face count in 9 of the 12 cases and predicts a strictly increasing budget from Low to Medium to High for all four objects. The three cases in which another method is closer are retained to avoid a favorable-only presentation.

A case figure alone cannot establish monotonic reliability. We therefore also evaluate the complete real benchmark object-wise. The object-wise analysis yields strict five-level monotonicity for 93.7% of source objects and no adjacent-level reversal for 97.9%. Thus, richer descriptions of the same source mesh received larger predicted budgets for 93.7% of the evaluated objects, and 97.9% had no adjacent-level reversal.

6.13. Discussion and Limitations

The experiments support four bounded conclusions. First, diversified synthetic supervision trained regressors that recover a strong relationship between textual requirements and face-count labels within the Text2MFN protocol. Second, transfer to the independent Objaverse/QEM benchmark is substantially stronger for MFPredictor than for most generic alternatives, without using real samples for training or selection. Third, the task-specific architecture contributes beyond a parameter-matched MLP and a generic Transformer regressor. Fourth, metric choice matters: ModernNCA has lower MAE and higher Acc_{10} , whereas MFPredictor is better under Log-MAE, RMSE, MAPE, Acc_{20} , and Acc_{30} .

The independent audit, controlled text-cue study, and end-to-end application extend the evidence in three ways. The audit quantifies annotation consistency against independent expert judgments; the controlled pairs demonstrate sensitivity to geometric cues beyond object-category shortcuts; and the application study connects regression accuracy to a measured mesh quality–cost trade-off.

Quality Level	Low	Medium	High
Mesh			
Description	A small single-engine high-wing airplane with an enclosed cabin, straight wings supported by struts, a vertical tail, a two-blade propeller, and three fixed landing wheels.	A light high-wing airplane with a shaped fuselage, enclosed cockpit windows, wing support struts, a modeled nose and propeller assembly, fixed landing gear, and defined horizontal and vertical tail surfaces.	A light utility airplane with a smoothly shaped fuselage, framed cockpit windows, refined wings and tail surfaces, detailed support struts, a modeled engine cowling, propeller hub, antenna, landing wheels, and wheel connections.
Results	(3001,1644,1847, <u>1574</u> ,1614)	(2846,3214,3511, <u>3444</u> ,3456)	(7412,6233,6741, <u>6749</u> ,6813)
Mesh			
Description	A simple axe with a recognizable metal blade, a defined head connection, and a slightly shaped wooden handle with a widened grip end.	A structured axe with a curved cutting edge, a modeled blade thickness, a reinforced head, and a gently curved handle with a defined grip.	A refined axe with a smoothly curved blade, a rounded metal head, a detailed top wedge, and an ergonomically shaped handle with softened edges.
Results	(517,214,310, <u>298</u> ,281)	(324,648,661, <u>664</u> ,667)	(1888,2010,1871, <u>1785</u> ,1609)
Mesh			
Description	A wooden chair with a defined square seat, four slightly tapered legs, horizontal support bars, and a broad rectangular back panel.	A structured wooden chair with a shaped seat frame, four tapered legs, reinforced side and front supports, and a framed rectangular backrest.	A refined wooden chair with slightly rounded seat edges, smoothly shaped legs, detailed support rails, clean structural joints, and a well-defined back panel.
Results	(712,423,407, <u>411</u> ,426)	(864,959, <u>950</u> ,988,953)	(6432,1983,2310, <u>2288</u> ,2204)
Mesh			
Description	A seated kitten with a rounded head, defined triangular ears, modeled eyes and nose, straight front legs, rounded hindquarters, separate paws, and a curved tail along the ground.	A seated kitten with a shaped face, defined muzzle and cheeks, pointed ears, rounded eyes, a smoothly connected neck and body, articulated front legs, hind legs, paws, and a curved tail.	A refined seated kitten with smooth facial contours, shaped eyelids, a defined nose and mouth, rounded cheeks, detailed ears, naturally curved legs and paws, a modeled chest, hindquarters, and a smoothly tapering tail.
Results	(1054,841,702, <u>718</u> ,729)	(4864,1540,1577, <u>1686</u> ,1657)	(7432,3983,3995, <u>3788</u> ,3810)
■ : xRFM ■ : TabM ■ : ModernNCA ■ : Ours ■ : GT — : Best			

Figure 8. Qualitative comparison of face-count predictions for representative real meshes at low, medium, and high quality levels. The colored numerical values correspond to the predictions of different methods: blue for xRFM, green for TabM, purple for ModernNCA, yellow for MFPredictor, and red for the ground-truth label. For each example, the underlined value indicates the prediction closest to the ground-truth face count.

The evidence does not show that MFPredictor solves mesh-complexity selection universally. Only one synthetic training dataset and one independent real benchmark are evaluated. The real benchmark is derived from Objaverse and QEM and does not cover CAD collections, scanned assets, production game meshes, non-triangular representations, or the outputs of other simplification algorithms. Its LOD-specific texts are also more explicit and structured than many naturally written user prompts.

The target itself is not uniquely identifiable from language. An appropriate count changes with topology, generator design, camera distance, screen resolution, deformation,

texture and normal-map use, hardware, and application. The conformal experiment shows that intervals calibrated on synthetic data under-cover on the shifted real benchmark; deployment in another labeled domain therefore requires domain-specific recalibration.

The filtering protocol does not eliminate the risks of synthetic templates, semantic shortcuts, or label bias. The independent audit quantifies these residual risks. The released dataset, source code, documented construction protocol, and aggregate evaluation results support verification of the reported workflow; exact regeneration of externally hosted generative-model responses is not assumed. The frozen English sentence encoder limits multilingual and domain-specific prompts. The end-to-end protocol covers one generation system and one simplifier and treats the initially generated mesh as a high-resolution reference rather than an absolute perceptual ground truth. The validated budget–quality relationship is therefore restricted to the evaluated generators, topology constraints, representations, platforms, deformation settings, and rendering pipeline.

7. Conclusions

This study formulated text-conditioned mesh face-count recommendation as a pre-generation regression task. Rather than generating or simplifying a mesh, the task predicts a practical triangular-face budget from the object category, silhouette, parts, structural relationships, and geometry-related details expressed in a natural-language description. The value is explicitly defined as a protocol-dependent recommendation, not as the original count of an asset or a universally unique optimum.

We constructed Text2MFN with 1,000,000 synthetic text–face-count pairs and an independent benchmark with 200,000 QEM-derived LOD records from 40,000 Objaverse meshes. No real sample is used for training, hyperparameter selection, checkpoint selection, or uncertainty calibration. We introduced MFPredictor, a lightweight dual-branch model that combines semantic context and compact geometry-related cues through bidirectional calibration, gated fusion, a small mixture of experts, and residual regression.

On the Text2MFN test set, MFPredictor obtains a MAPE of 4.38% and an Acc_{20} of 95.45%. On the Objaverse-based benchmark, it obtains a MAPE of 7.96% and Acc_{20} of 86.41%. The broader regression comparisons, module ablations, and repeated-run statistics indicate that the task-specific architecture improves scale-aware accuracy and moderate-tolerance reliability. ModernNCA is stronger under MAE and Acc_{10} , demonstrating that the proposed method is not uniformly superior under every error criterion. This study also reports an independent annotation audit, controlled within-category textual tests, and an end-to-end text-to-3D–simplification experiment. The measured results support annotation reliability, sensitivity to fine-grained geometric cues, and a practical quality–cost advantage within the evaluated pipeline.

The real-mesh evaluation demonstrates transfer within the specific Text2MFN-to-Objaverse/QEM setting. Comparable performance is not established for other object collections, mesh generators, simplification algorithms, representations, languages, or deployment platforms. The validated scope of this study is restricted to the reported datasets, English-language encoder, Hunyuan3D 2.0 generator, QEM simplifier, and evaluation conditions.

Supplementary Materials: The following supporting information can be downloaded at <https://www.mdpi.com/article/10.3390/app16157722/s1>, Supplementary File S1, the Text2MFN dataset-construction and quality-control protocol and the scope of the public dataset fields; Supplementary File S2, the internal partition definitions and permitted uses, integrity rules, and a deterministic public reference split consisting of a combined row-index assignment, four partition-specific index files, a generation script, and SHA-256 checksums; Supplementary File S3, the common hyperparameter-selection protocol, selected MFPredictor configuration, implementation environment, aggregate

repeated-run results, paired object-clustered bootstrap comparisons, and the pre-specified object-wise MAPE test; Supplementary File S4, the target-blind real-benchmark description-generation protocol and the end-to-end Hunyuan3D 2.0-QEM application protocol, aggregate measurements, and derived comparisons; and Supplementary File S5, the independent annotation-audit protocol, interval-containment result, and ordinal-agreement statistic. The public reference split in S2 is supplied for reproducible future comparisons and is not claimed to be the unavailable internal row assignment used to produce the reported tables. The supporting information does not include auxiliary record-level generation metadata, the internal record-level experimental assignment, individual hyperparameter-trial logs, per-seed predictions, prompt-level application measurements, or individual rater records.

Author Contributions: Conceptualization, B.J. and S.C.; methodology, B.J.; software, B.J.; validation, B.J. and S.C.; formal analysis, B.J.; investigation, B.J.; resources, B.J. and S.C.; data curation, B.J.; writing—original draft preparation, B.J.; writing—review and editing, B.J. and S.C.; visualization, B.J.; supervision, S.C.; project administration, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: The online survey protocol was determined to be exempt from full ethical review by the competent research ethics body at Linyi University. The exemption certificate was submitted with the manuscript. Recruitment began on 3 July 2026 after the exemption determination had been documented.

Informed Consent Statement: Electronic informed consent was obtained from every survey respondent before the eligibility questions were displayed. Participation was voluntary, and respondents were informed of the study purpose, stated completion time, data use, and their right to leave before submitting the questionnaire.

Data Availability Statement: The public Text2MFN release is available on Hugging Face at <https://huggingface.co/datasets/jibingyang111/Text2MFN> (accessed on 28 July 2026). It contains 1,000,000 records in one split named `train`, with the two released fields `text` and `recommended_faces`. Auxiliary construction metadata, including category, geometry-factor, application, batch, and quality-control fields, are not released at record level. The internal experimental partition used for the reported tables comprised 700,000 training records, 150,000 model-selection records, 50,000 calibration records, and 100,000 held-out test records, but its exact row assignment is not redistributed. Supplementary File S2 additionally releases a deterministic public reference split over the public-table row order, generated with NumPy 2.3.5 PCG64 and seed 2026. It contains a combined zero-based row-index assignment, four partition-specific index files, the generation script, and SHA-256 checksums. The reference split is defined for the Hugging Face repository state accessed on 28 July 2026 using revision selector `main`: one `train` split with 1,000,000 rows, fields `text` and `recommended_faces`, and a displayed total file size of 382 MB. S2 records this dated snapshot description and a SHA-256 metadata fingerprint; because `main` is mutable, later users must use an archived copy of that state or verify identical row ordering before applying the indices. This public reference split is intended for reproducible future comparisons and is not claimed to reproduce the unavailable internal assignment underlying the reported tables. Source code is publicly available at <https://github.com/lifei111122222-alt/Text2MeshNum> (accessed on 1 August 2026). The remaining supporting files contain protocol-level construction and quality-control information, hyperparameter-selection and implementation details, aggregate repeated-run results, paired object-clustered bootstrap comparisons and the pre-specified object-wise MAPE test, the target-blind real-benchmark description-generation protocol and the end-to-end application protocol and aggregate measurements, and the annotation-audit protocol with interval-containment and ordinal-agreement results. They do not contain auxiliary record-level generation metadata, individual training-run predictions, prompt-level application measurements, individual rater records, or respondent-level survey data. Objaverse assets remain subject to their original licenses and are not redistributed. Aggregate survey counts and percentages are reported in the manuscript.

Acknowledgments: During the preparation of this study and manuscript, the authors used OpenAI ChatGPT (version 5.5) for generation-prompt design, diversity-factor expansion, candidate text and label generation, a separate consistency-review pass, survey-protocol drafting, and language polishing. The role of ChatGPT 5.5 in the dataset-construction and quality-control protocol is documented in Supplementary File S1. ChatGPT was not used to implement the final model, execute experiments, calculate the reported statistics, select favorable results, or make independent scientific conclusions. The authors designed the study, verified the generated data, implemented the methods, conducted and checked the experiments, reviewed and edited all AI-assisted outputs, and take full responsibility for the content of this publication. The authors thank all survey participants for their time and contributions.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Siddiqui, Y.; Alliegro, A.; Artemov, A.; Tommasi, T.; Sirigatti, D.; Rosov, V.; Dai, A.; Nießner, M. MeshGPT: Generating Triangle Meshes with Decoder-Only Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: New York, NY, USA, 2024; pp. 19615–19625.
2. Chen, S.; Chen, X.; Pang, A.; Zeng, X.; Cheng, W.; Fu, Y.; Yin, F.; Wang, Z.; Yu, J.; Yu, G.; et al. MeshXL: Neural Coordinate Field for Generative 3D Foundation Models. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2024; Volume 37. <https://doi.org/10.52202/079017-3080>.
3. Siddiqui, Y.; Monnier, T.; Kokkinos, F.; Kariya, M.; Kleiman, Y.; Garreau, E.; Gafni, O.; Neverova, N.; Vedaldi, A.; Shapovalov, R.; et al. Meta 3D AssetGen: Text-to-Mesh Generation with High-Quality Geometry, Texture, and PBR Materials. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2024; Volume 37.
4. Chen, Y.; He, T.; Huang, D.; Ye, W.; Chen, S.; Tang, J.; Cai, Z.; Yang, L.; Yu, G.; Lin, G.; et al. MeshAnything: Artist-Created Mesh Generation with Autoregressive Transformers. In *Proceedings of the International Conference on Learning Representations*, Singapore, 24–28 April 2025.
5. Chen, Y.; Wang, Y.; Luo, Y.; Wang, Z.; Chen, Z.; Zhu, J.; Zhang, C.; Lin, G. MeshAnything V2: Artist-Created Mesh Generation with Adjacent Mesh Tokenization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*; IEEE: New York, NY, USA, 2025; pp. 13922–13931.
6. Fang, S.; Shen, I.C.; Wang, Y.; Tsai, Y.H.; Yang, Y.; Zhou, S.; Ding, W.; Igarashi, T.; Yang, M.H. MeshLLM: Empowering Large Language Models to Progressively Understand and Generate 3D Mesh. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*; IEEE: New York, NY, USA, 2025; pp. 14061–14072.
7. Lionar, S.; Liang, J.; Lee, G.H. TreeMeshGPT: Artistic Mesh Generation with Autoregressive Tree Sequencing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: New York, NY, USA, 2025; pp. 26608–26617.
8. Gao, D.; Siddiqui, Y.; Li, L.; Dai, A. MeshArt: Generating Articulated Meshes with Structure-Guided Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: New York, NY, USA, 2025; pp. 618–627.
9. Weng, H.; Zhao, Z.; Lei, B.; Yang, X.; Liu, J.; Lai, Z.; Chen, Z.; Liu, Y.; Jiang, J.; Guo, C.; et al. Scaling Mesh Generation via Compressive Tokenization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: New York, NY, USA, 2025; pp. 11093–11103.
10. Zhang, X.; Siddiqui, Y.; Avetisyan, A.; Xie, C.; Engel, J.; Howard-Jenkins, H. VertexRegen: Mesh Generation with Continuous Level of Detail. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*; IEEE: New York, NY, USA, 2025; pp. 12570–12580.
11. Garland, M.; Heckbert, P.S. Surface Simplification Using Quadric Error Metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*; ACM Press/Addison-Wesley Publishing Co.: New York, NY, USA, 1997; pp. 209–216. <https://doi.org/10.1145/258734.258849>.
12. Brown, L.D.; Cai, T.T.; DasGupta, A. Interval Estimation for a Binomial Proportion. *Stat. Sci.* **2001**, *16*, 101–133. <https://doi.org/10.1214/ss/1009213286>.
13. Eysenbach, G. Improving the Quality of Web Surveys: The Checklist for Reporting Results of Internet E-Surveys (CHERRIES). *J. Med. Internet Res.* **2004**, *6*, e34. <https://doi.org/10.2196/jmir.6.3.e34>.
14. Deitke, M.; Schwenk, D.; Salvador, J.; Weihs, L.; Michel, O.; VanderBilt, E.; Schmidt, L.; Ehsani, K.; Kembhavi, A.; Farhadi, A. Objaverse: A Universe of Annotated 3D Objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*; IEEE: New York, NY, USA, 2023; pp. 13142–13153.
15. Deitke, M.; Liu, R.; Wallingford, M.; Ngo, H.; Michel, O.; Kusupati, A.; Fan, A.; Laforte, C.; Voleti, V.; Gadre, S.Y.; et al. Objaverse-XL: A Universe of 10M+ 3D Objects. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; Volume 36.

16. Luo, T.; Rockwell, C.; Lee, H.; Johnson, J. Scalable 3D Captioning with Pretrained Models. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; Volume 36.
17. Xu, Z.; Jiang, F.; Niu, L.; Deng, Y.; Poovendran, R.; Choi, Y.; Lin, B.Y. Magpie: Alignment Data Synthesis from Scratch by Prompting Aligned LLMs with Nothing. In *Proceedings of the International Conference on Learning Representations*, Singapore, 24–28 April 2025.
18. Long, L.; Wang, R.; Xiao, R.; Zhao, J.; Ding, X.; Chen, G.; Wang, H. On LLMs-Driven Synthetic Data Generation, Curation, and Evaluation: A Survey. In *Proceedings of the Findings of the Association for Computational Linguistics: ACL 2024, Bangkok, Thailand*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2024; pp. 11065–11082. <https://doi.org/10.18653/v1/2024.findings-acl.658>.
19. Kim, S.; Suk, J.; Yue, X.; Viswanathan, V.; Lee, S.; Wang, Y.; Gashteovski, K.; Lawrence, C.; Welleck, S.; Neubig, G. Evaluating Language Models as Synthetic Data Generators. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vienna, Austria; Association for Computational Linguistics: Stroudsburg, PA, USA, 2025; pp. 6385–6403. <https://doi.org/10.18653/v1/2025.acl-long.320>.
20. Shumailov, I.; Shumaylov, Z.; Zhao, Y.; Papernot, N.; Anderson, R.; Gal, Y. AI Models Collapse When Trained on Recursively Generated Data. *Nature* **2024**, *631*, 755–759. <https://doi.org/10.1038/s41586-024-07566-y>.
21. Luo, T.; Johnson, J.; Lee, H. View Selection for 3D Captioning via Diffusion Ranking. In *Proceedings of the Computer Vision—ECCV 2024; Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2025; Volume 15089, pp. 180–197. https://doi.org/10.1007/978-3-031-72751-1_11.
22. Artstein, R.; Poesio, M. Inter-Coder Agreement for Computational Linguistics. *Comput. Linguist.* **2008**, *34*, 555–596. <https://doi.org/10.1162/coli.07-034-R2>.
23. Koo, T.K.; Li, M.Y. A Guideline of Selecting and Reporting Intraclass Correlation Coefficients for Reliability Research. *J. Chiropr. Med.* **2016**, *15*, 155–163. <https://doi.org/10.1016/j.jcm.2016.02.012>.
24. Chen, Z.; Pan, Z.; Wu, K.; Vouga, E.; Gao, X. Robust Low-Poly Meshing for General 3D Models. *ACM Trans. Graph.* **2023**, *42*, 1–20. <https://doi.org/10.1145/3592396>.
25. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. <https://doi.org/10.1023/A:1010933404324>.
26. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; ACM: New York, NY, USA, 2016; pp. 785–794. <https://doi.org/10.1145/2939672.2939785>.
27. Gorishniy, Y.; Rubachev, I.; Khrulkov, V.; Babenko, A. Revisiting Deep Learning Models for Tabular Data. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 18932–18943.
28. Gorishniy, Y.; Kotelnikov, A.; Babenko, A. TabM: Advancing Tabular Deep Learning with Parameter-Efficient Ensembling. In *Proceedings of the International Conference on Learning Representations*, Singapore, 24–28 April 2025.
29. Ye, H.J.; Yin, H.H.; Zhan, D.C.; Chao, W.L. Revisiting Nearest Neighbor for Tabular Data: A Deep Tabular Baseline Two Decades Later. In *Proceedings of the International Conference on Learning Representations*, Singapore, 24–28 April 2025.
30. Beaglehole, D.; Holzmüller, D.; Radhakrishnan, A.; Belkin, M. xRFM: Accurate, Scalable, and Interpretable Feature Learning Models for Tabular Data. In *Proceedings of the International Conference on Learning Representations*, Rio de Janeiro, Brazil, 23–27 April 2026.
31. Lakshminarayanan, B.; Pritzel, A.; Blundell, C. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
32. Kuleshov, V.; Fenner, N.; Ermon, S. Accurate Uncertainties for Deep Learning Using Calibrated Regression. In *Proceedings of the 35th International Conference on Machine Learning*; PMLR: New York, NY, USA, 2018; Volume 80, pp. 2796–2804.
33. Romano, Y.; Patterson, E.; Candès, E.J. Conformalized Quantile Regression. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.
34. Ribeiro, M.T.; Wu, T.; Guestrin, C.; Singh, S. Beyond Accuracy: Behavioral Testing of NLP Models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online, 5–10 July 2020; pp. 4902–4912. <https://doi.org/10.18653/v1/2020.acl-main.442>.
35. Reimers, N.; Gurevych, I. Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2019; pp. 3982–3992. <https://doi.org/10.18653/v1/D19-1410>.
36. Wang, W.; Wei, F.; Dong, L.; Bao, H.; Yang, N.; Zhou, M. MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2020; Volume 33, pp. 5776–5788.
37. Jacobs, R.A.; Jordan, M.I.; Nowlan, S.J.; Hinton, G.E. Adaptive Mixtures of Local Experts. *Neural Comput.* **1991**, *3*, 79–87. <https://doi.org/10.1162/neco.1991.3.1.79>.

38. Huber, P.J. Robust Estimation of a Location Parameter. *Ann. Math. Stat.* **1964**, *35*, 73–101. <https://doi.org/10.1214/aoms/1177703732>.
39. Pineau, J.; Vincent-Lamarre, P.; Sinha, K.; Larivière, V.; Beygelzimer, A.; d'Alché Buc, F.; Fox, E.; Larochelle, H. Improving Reproducibility in Machine Learning Research: A Report from the NeurIPS 2019 Reproducibility Program. *J. Mach. Learn. Res.* **2021**, *22*, 1–20.
40. Efron, B. Bootstrap Methods: Another Look at the Jackknife. *Ann. Stat.* **1979**, *7*, 1–26. <https://doi.org/10.1214/aos/1176344552>.
41. Wilcoxon, F. Individual Comparisons by Ranking Methods. *Biom. Bull.* **1945**, *1*, 80–83. <https://doi.org/10.2307/3001968>.
42. Zhao, Z.; Lai, Z.; Lin, Q.; Zhao, Y.; Liu, H.; Yang, S.; Feng, Y.; Yang, M.; Zhang, S.; Yang, X.; et al. Hunyuan3D 2.0: Scaling Diffusion Models for High-Resolution Textured 3D Asset Generation. *arXiv* **2025**, arXiv:2501.12202. <https://doi.org/10.48550/arXiv.2501.12202>.
43. Fan, H.; Su, H.; Guibas, L.J. A Point Set Generation Network for 3D Object Reconstruction from a Single Image. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; IEEE: New York, NY, USA, 2017; pp. 2463–2471. <https://doi.org/10.1109/CVPR.2017.264>.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.