



ARTICLE

A Numerically Aware GPT-Lite Method for Aluminum Alloy Composition Design

Xingci Hou^{1,2}, Yingli Liu^{1,2,*}, Jiancheng Yin^{3,*} and Haihe Zhou^{1,2}

¹Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming, China

²Yunnan Key Laboratory of Computer Technologies Application, Kunming University of Science and Technology, Kunming, China

³Faculty of Materials Science and Engineering, Kunming University of Science and Technology, Kunming, China

*Corresponding Authors: Yingli Liu. Email: lyl@kust.edu.cn; Jiancheng Yin. Email: yjc@kust.edu.cn

Received: 15 July 2026; Accepted: 08 September 2026; Published: 22 October 2026

ABSTRACT: AI for Science is extending materials modeling from property prediction toward inverse design. This study proposes Generative Pre-trained Transformer-lite (GPT-lite), a numerically aware framework for target-property-driven aluminum alloy composition design. Using a processed dataset containing 3553 composition–property records, 25 elemental variables, and three mechanical properties—elongation, tensile strength, and yield strength—the framework models forward property prediction and inverse composition generation as structured sequence tasks. EnhancedTokenizer is introduced to reduce numerical fragmentation, while a prefix-aware regression head provides continuous-property supervision to the lightweight Transformer backbone. In inverse design, generated compositions are subjected to strict parsing, compositional clipping and normalization, and unified evaluation by an external multilayer perceptron (MLP) surrogate. Under fixed data splits, 100 held-out targets, and the same candidate budget, GPT-lite achieves a mean best maximum relative error of 0.029679 and success rates of 0.85, 0.97, and 0.98 at relative-error thresholds of 0.05, 0.08, and 0.10, respectively, outperforming Non-dominated Sorting Genetic Algorithm II (NSGA-II), conditional variational autoencoder (CVAE), and conditional Wasserstein generative adversarial network with gradient penalty (cWGAN-GP) under the unified comparison protocol. Independent zero-shot validation on 40 external aluminum-alloy compositions yields R^2 values of 0.420, 0.748, and 0.740 for elongation, tensile strength, and yield strength. These results support the feasibility of combining numerically aware sequence modeling, continuous-property supervision, and constraint-aware evaluation for lightweight aluminum alloy inverse design.

KEYWORDS: Aluminum alloy; composition design; inverse design; GPT-lite; numerically aware encoding; property prediction

1 Introduction

AI for Science is driving artificial intelligence methods to evolve from traditional data-fitting tools toward prediction and generation for scientific problems. For materials design tasks, the value of artificial intelligence methods lies not only in establishing property prediction models from existing experimental data [1–3], but also in assisting the generation of candidate compositions that satisfy target property requirements by learning complex patterns in structured data [4,5]. In recent years, large language models and Transformer architectures have shown strong capabilities in sequence modeling, contextual representation, and conditional generation. These capabilities have enabled their application to extend beyond natural language processing to scientific data modeling tasks, such as molecules [6], crystals [7], and material compositions [8]. Recent studies have also explored multimodal and generalist material-intelligence

systems [9,10], as well as integrated data-driven platforms for materials modeling [11]. For example, AI-driven approaches have been systematically reviewed for nanoscale heat conduction and radiation, while machine-learning-assisted design has also been applied to advanced polymeric materials [12]. Nevertheless, aluminum alloy composition–property datasets are often limited in size. Small materials datasets may increase the risk of overfitting and make model selection more sensitive [13]. Therefore, developing lightweight language models for small-scale structured alloy data is a critical issue for improving the feasibility and reproducibility of AI-assisted composition design.

In AI-assisted alloy design, a central computational challenge is to learn reliable mappings from limited composition–property data and further generate evaluable candidate compositions that meet target property requirements. Aluminum alloys are characterized by low density, high specific strength, and favorable mechanical properties, making them widely applicable in aerospace, transportation, and lightweight structural components [14,15]. However, aluminum alloy design is generally not a single-property maximization problem. Instead, it often involves a strength–ductility trade-off [16], while multiple mechanical properties, including elongation, tensile strength, and yield strength, need to be considered simultaneously. Accordingly, aluminum alloy composition design requires models not only to learn the forward mapping from compositions to properties, but also to generate valid candidate compositions that satisfy target property requirements within a high-dimensional, continuous, and sparse composition space.

In existing studies on AI-assisted alloy design, one line of research primarily focuses on property prediction, where material properties are predicted based on given alloy compositions or relevant descriptors. Chaudry et al. [17] used machine learning methods to assist in the design of high-performance aluminum alloys, demonstrating that machine learning can accelerate the evaluation of alloy properties. Bhat et al. [18] further employed a data-driven partitioned regression method to predict the yield strength, tensile strength, and elongation of aluminum alloys, and indicated that differences in data distribution among different aluminum alloy categories may affect model prediction performance. These forward prediction models can rapidly estimate the properties of candidate alloys, thereby providing an evaluation basis for subsequent screening and optimization. However, forward prediction models themselves can only answer the question of “what properties will be obtained from a given composition”, but cannot directly answer “what composition should be generated for given target properties”. Therefore, relying solely on forward prediction remains insufficient for establishing a complete target-property-driven composition design process.

To further realize inverse design from target properties to candidate compositions, some studies have adopted an artificial-intelligence-based search paradigm of “predictor + optimizer”. Bhat et al. [19] proposed an inverse design method for aluminum alloys based on multi-objective regression, attempting to infer alloy compositions from target properties. Lee et al. [20] combined machine learning with a genetic algorithm to design novel aluminum alloys that can alleviate the strength–ductility trade-off. NSGA-II, proposed by Deb et al. [21], can handle multi-objective optimization problems and is therefore commonly used for multi-property candidate search. Bayesian optimization methods summarized by Shahriari et al. [22] are suitable for expensive black-box function optimization. Tian et al. [23] further applied target-oriented Bayesian optimization to materials design to reduce the cost of candidate evaluation.

In addition to optimization-based search methods, generative artificial intelligence methods have also been applied to inverse materials design [24,25]. The advantage of generative models is that they can directly learn sample distributions and produce new candidate compositions, rather than repeatedly iterating within a fixed search space. Noh et al. [26,27] summarized and demonstrated machine-learning-driven inverse design strategies for inorganic solid-state materials, in which generative models are regarded as an important route from target properties to candidate structures or compositions. Lu et al. [28] also pointed out that deep generative models are becoming an important approach for achieving property-to-structure design in

materials discovery. MatGAN, proposed by Dan et al. [29] shows that generative adversarial networks can learn the compositional space of inorganic materials and be used for inverse design sampling. Türk et al. [30] evaluated multiple types of deep generative models from the perspective of chemical composition space, indicating that the effectiveness of generative models depends not only on generative capability, but also on validity, coverage, and diversity. These studies provide an important foundation for generative modeling in materials design. However, for continuous numerical composition data such as aluminum alloys, generative models still face challenges in numerical representation and candidate validity. On the one hand, continuous decimals and trace elements may lead to excessively long sequence representations or over-segmentation of numerical information. On the other hand, generated candidates may suffer from problems such as parsing failure, out-of-range compositions, or unreasonable compositional sums. Therefore, a unified protocol for candidate correction and evaluation is still required.

Recent studies have begun to use Transformer-based language modeling for materials generation and design. The Crystal Composition Transformer uses a blank-filling language model to learn compositional patterns for generative and tinkering design of inorganic materials [8]. Dong et al. developed the Material Transformer Generator (MTG), in which Transformer-based language models generate candidate two-dimensional material compositions that are subsequently subjected to structure prediction and DFT-based validation [31]. More recently, Ni et al. proposed AlloyGPT, which represents alloy data as structured text and uses an autoregressive GPT model to perform both forward prediction and inverse alloy design [32]. Together, these studies demonstrate that language-model-based representations can provide a unified way of learning and generating structured materials data.

However, directly transferring existing material language-model designs to small-scale aluminum alloy composition–property data presents several practical challenges. First, a large autoregressive backbone may be unnecessarily expensive when only several thousand composition–property samples are available. Second, aluminum alloy data contain continuous decimal values and sparse trace elements, making numerical representation particularly important because excessive token fragmentation increases sequence length and may obscure fine compositional differences. Third, autoregressive next-token prediction alone does not provide explicit continuous supervision for mechanical properties. Finally, inverse-generated compositions must satisfy numerical and compositional constraints before they can be meaningfully evaluated. Against this background, GPT-lite is designed as a task-specific lightweight framework rather than as a simple reduction of an existing material LLM. It combines a compact GPT backbone, a numerically aware tokenizer, a prefix-aware continuous regression branch, and a constraint-aware inverse-design evaluation workflow. These components are intended to jointly address numerical representation, property-aware learning, computational compactness, and candidate usability in small-scale aluminum alloy design.

Based on these considerations, this study investigates target-property-driven composition design using small-scale aluminum alloy composition–property data. The processed modeling dataset contains 3553 composition–property records, 25 elemental variables, and three mechanical properties: elongation, tensile strength, and yield strength. GPT-lite is developed to integrate property-aware representation learning and target-conditioned composition generation within a compact autoregressive framework. Rather than pursuing model scale, the proposed method focuses on adapting language modeling to the numerical characteristics and practical constraints of alloy composition design.

Specifically, the framework contains four closely connected design components. First, the forward task Pred (composition $\rightarrow$ property) and the inverse task Gene (target property $\rightarrow$ composition) are represented as structured sequences and learned by a shared GPT backbone. Second, EnhancedTokenizer is designed for elemental symbols, field boundaries, sparse compositions, and continuous decimal values to improve the compactness of numerical sequence representation. Third, a prefix-aware regression head introduces

continuous property supervision into the shared representation space while being restricted from accessing ground-truth property tokens. Finally, inverse-generated compositions undergo strict parsing and ON-base soft constraint correction before being evaluated and ranked by an independent external MLP surrogate. In this way, generation and evaluation remain functionally separated while following the same candidate constraints and evaluation protocol.

In summary, to address the aforementioned challenges, the specific research contributions of this paper are explicitly outlined as follows:

- A lightweight GPT-based architecture is developed for small-scale aluminum alloy composition–property modeling. The final model uses a 256-dimensional hidden representation, four Transformer layers, and four attention heads, resulting in 3.406M trainable parameters while supporting both forward property-aware learning and inverse composition generation.
- A numerically aware EnhancedTokenizer is designed for structured alloy sequences containing elemental fields, sparse non-zero compositions, and continuous decimal values, with the objective of reducing unnecessary numerical fragmentation and improving sequence compactness.
- A prefix-aware regression branch is integrated with the shared autoregressive backbone to provide explicit continuous-property supervision. The regression head operates only on the composition prefix and does not access ground-truth property tokens, enabling property-aware representation learning without direct label exposure.
- A unified inverse-design evaluation workflow is established by combining target-conditioned generation, strict parsing, ON-base compositional correction, and independent surrogate-based ranking. Fixed data splits, held-out targets, candidate budgets, and evaluators are used to improve experimental comparability and reproducibility.

2 Dataset

2.1 Dataset Source and Field Composition

This study uses a processed aluminum-alloy composition–property table derived from the non-dominated candidate alloys reported by Li and Birbilis [33]. The source study developed the NSGAN framework using the publicly available aluminum-alloy supervised-learning dataset released by Bhat et al. [34] and reported computationally generated non-dominated aluminum-alloy candidates. In the present study, these candidate results were reorganized according to the requirements of the composition–property modeling task. After field selection and basic numerical consistency screening, 3553 composition–property records were retained for subsequent modeling. The detailed generation and optimization settings of the original NSGAN framework are available in the source study and are therefore not repeated here.

For the present task, only the 25 elemental composition variables and three target mechanical properties—elongation, tensile strength, and yield strength—were retained. Processing-condition fields and other non-target information were excluded. It should be emphasized that the 3553 internal records are derived from computationally generated candidates reported in the source study rather than from 3553 newly measured physical alloys. Accordingly, the internal experiments primarily evaluate GPT-lite within this processed computational composition–property domain, while independent cross-dataset transfer is examined separately in [Section 4.4.2](#) using the external aluminum-alloy dataset reported by Pfeiffer et al. [35].

2.2 Data Preprocessing and Splitting Strategy

First, the three mechanical property columns were fixed as Elongation, Tensile, and Yield. Samples with missing values in any of these property columns were removed. Second, elemental composition columns were identified according to the format of chemical element symbols, namely field names consisting of one or two English letters. All elemental columns were converted into numerical values following standard materials data-mining procedures [36]. Missing values were filled with 0, and negative values were clipped to 0. These preprocessing steps were mainly used to ensure the trainability of the input matrix and target matrix, which is consistent with the basic requirements for data integrity [37,38] and careful workflow design in materials machine learning [39].

This study retains both the original composition space and the standardized modeling space. The original 25-dimensional composition vectors are used for candidate composition parsing, validity checking, composition-sum constraint checking, and final candidate evaluation. The standardized numerical representations are used for model training and target-conditioned input. For the property prediction task, both input compositions and output properties were standardized according to the statistics of the training set to improve model training stability. For the inverse design task, the target properties were first transformed into the standardized space and used as conditional inputs for the Gene prompt. The candidate compositions generated by the model were finally converted back to the original 25-dimensional composition space, where strict parsing, constraint repair, and surrogate-based post-evaluation were performed.

To avoid the unstable influence of clearly out-of-range property values on inverse design target selection, standardization transformation, and relative error calculation, property range constraints were defined in the inverse design experiments. Specifically, the ranges were set as 1–50 for Elongation, 150–900 MPa for Tensile strength, and 100–800 MPa for Yield strength. This setting was mainly used to ensure that the held-out target properties were within an evaluable range, rather than to artificially modify the composition–property mapping required for model training.

The dataset was split using a fixed random seed of 42. In the main experiments, the dataset was divided into training, validation, and test sets at a ratio of 7:1:2, corresponding to 2487, 355, and 711 samples, respectively. The property standardizer was fitted only on the training set and then applied to the validation and test sets to reduce the risk of test information leakage. For the inverse composition design experiments, 100 fixed held-out target property vectors were further selected from the test set as design targets. All comparison methods used the same data split, target set, candidate budget, and evaluator, thereby ensuring consistency in the comparison among NSGA-II, GPT-lite, CVAE, and cWGAN-GP. In the raw validity check, candidate compositions were required to be non-negative, each elemental content should not exceed 1, and the sum of elemental contents should be close to 1.0. In the ON-base soft constraint, candidate compositions were normalized to $\sum x_i = 1.0$ after boundary clipping. The main data-splitting and inverse-design settings are summarized in Table 1.

2.3 Composition Serialization Representation and Task Construction

Since GPT-lite is based on an autoregressive language model, the tabular composition–property data need to be converted into structured sequences. Existing studies have shown that material properties can be learned directly from elemental composition or stoichiometric information using composition-only neural representations [40], with the key lying in the choice of an appropriate representation method and model architecture. Meanwhile, Wei et al. [8] converted material compositions into structured sequences for generative design and composition completion tasks, indicating that serialized representations can serve as the input basis for material language models. Based on this idea, this study represents each aluminum alloy sample as a text sequence with an explicit task identifier, field boundaries, and a fixed elemental order.

Table 1: Data split and inverse design target settings.

Item	Setting
Number of valid samples	3553
Elemental composition dimension	25
Whether processing fields are included	No
Property targets	Elongation, Tensile strength, Yield strength
Data split ratio	Train/Validation/Test = 7:1:2
Number of samples in each split	2487/355/711
Random seed	42
Number of inverse design targets	100 fixed held-out targets
Composition-sum constraint	Sum $\approx$ 1.0
Property ranges for inverse design	E: 1–50; T: 150–900 MPa; Y: 100–800 MPa

For the property prediction task, the task identifier is defined as Pred:

Pred: COMP $\rightarrow$ PROP

where COMP denotes the 25-dimensional composition sequence organized according to a fixed elemental order, and PROP denotes the three-dimensional property vector consisting of Elongation, Tensile strength, and Yield strength. The corresponding structured text format can be expressed as follows:

{Task}e4 COMP: [Ag = x_1 , Al = x_2 , ..., Zr = x_{25}] $\rightarrow$ PROP: [E, T, Y]

Gene: PROP $\rightarrow$ COMP

where the input is the target property vector and the output is the 25-dimensional candidate composition vector arranged according to the fixed elemental order. The corresponding structured text format can be expressed as follows:

{Task} PROP: [E, T, Y] $\rightarrow$ COMP: [Ag = x_1 , Al = x_2 , ..., Zr = x_{25}]

The purpose of adopting a fixed elemental order is to ensure that the outputs of different models are located in the same 25-dimensional composition space, thereby enabling unified post-evaluation of GPT-lite, NSGA-II, CVAE, and cWGAN-GP under the same external surrogate.

It should be noted that the fixed elemental order is a serialization convention rather than a permutation-invariant representation of chemical composition. From a chemical perspective, an alloy composition is determined by the identities and fractions of its constituent elements and should not depend on the arbitrary order in which these element–fraction pairs are listed. In contrast, an autoregressive Transformer processes an ordered token sequence and can therefore learn position-dependent relationships that do not correspond to an intrinsic ordering of the chemical mixture. The fixed ordering adopted in this study removes representation ambiguity across samples and enables consistent training and evaluation, but it does not mathematically guarantee permutation invariance. This limitation is further discussed in [Section 5.2](#).

3 GPT-Lite Method

3.1 Overall Framework

The proposed GPT-lite is a generative framework for aluminum alloy composition design. The core idea of this framework is to transform aluminum alloy compositions and mechanical properties into structured sequences with explicit field boundaries, enabling the autoregressive language model to learn both composition–property mappings and composition generation conditioned on target properties. The

objective of GPT-lite is to establish a complete design workflow under a unified sequence modeling framework, including property-aware representation learning, candidate composition generation, constraint repair, and surrogate-based back-evaluation. The overall framework of GPT-lite is shown in Fig. 1.

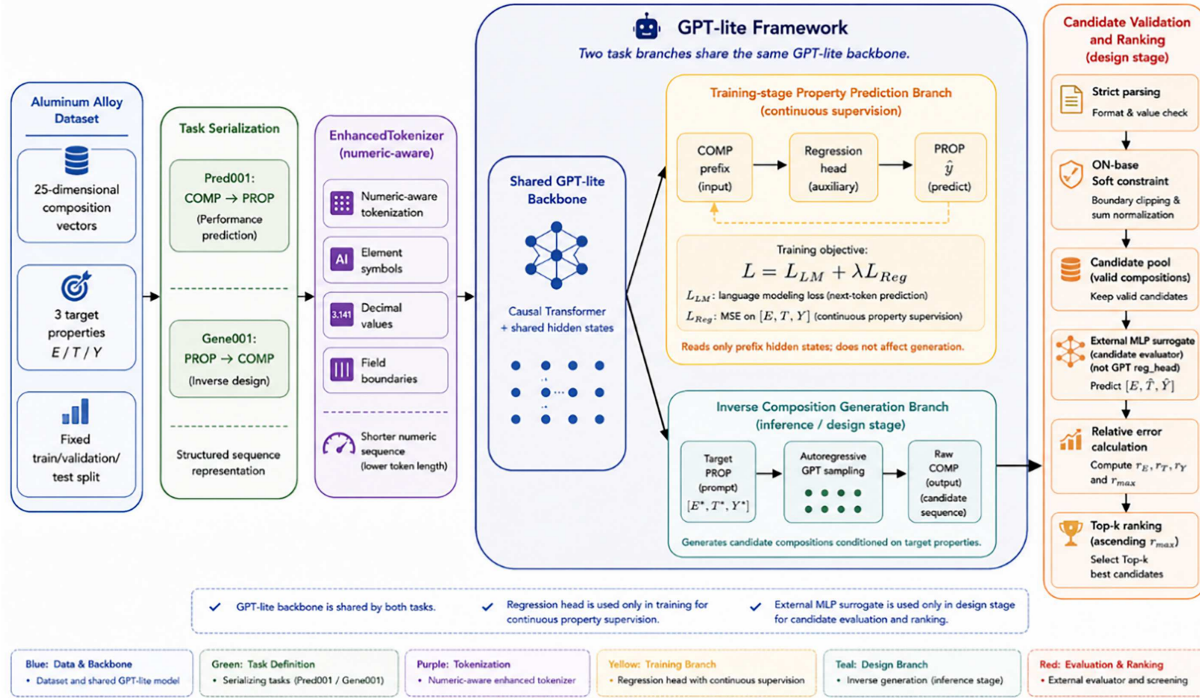


Figure 1: Overall framework of GPT-lite.

As shown in Fig. 1, the GPT-lite framework first converts the 25-dimensional composition vectors and three mechanical properties in the aluminum alloy dataset into structured task sequences. Specifically, Pred denotes the forward property prediction task, namely COMP → PROP, whereas Gene denotes the inverse composition generation task, namely PROP → COMP. Subsequently, the EnhancedTokenizer performs numerically aware encoding of elemental symbols, decimal values, and field boundaries, and the resulting token sequences are fed into the shared GPT-lite backbone to learn composition–property relationships and conditional generation representations.

Fig. 1 summarizes the two task branches of GPT-lite. The Pred branch uses the shared GPT-lite backbone and a prefix-aware regression head for property-aware learning, whereas the Gene branch generates candidate compositions conditioned on target properties. Generated candidates are subsequently parsed, corrected using the ON-base constraint, evaluated by the external MLP surrogate, and ranked according to r_{max} . The candidate with the minimum r_{max} for each target is used to calculate best_max_rel and SR@ ϵ .

The GPT regression head provides auxiliary continuous-property supervision during GPT-lite training, whereas the external MLP surrogate is used only for unified candidate evaluation in inverse design. This separation avoids evaluating GPT-lite-generated candidates using its own regression head. To clarify the distinction from existing materials language models, Table 2 summarizes the main architectural differences and practical advantages of GPT-lite.

Table 2: Comparison with representative transformer- and LLM-based materials design methods.

Method	Main Architecture/Design	Key Architectural Feature	Practical Focus
Transformer-based materials generators [8,31]	Transformer language models for composition generation or controllable materials design	Primarily designed for generative composition modeling; GPT-lite additionally introduces explicit continuous-property supervision through a prefix-aware regression branch	Candidate generation and downstream materials discovery
AlloyGPT [32]	Large autoregressive GPT for forward prediction and inverse alloy design	Uses a substantially larger GPT backbone and character-based numerical representation, whereas GPT-lite introduces a lightweight backbone with a separate prefix-aware regression branch for continuous-property supervision	Large-capacity unified alloy language modeling
GPT-lite (this work)	Lightweight GPT + EnhancedTokenizer + prefix-aware regression head + constraint-aware evaluation	256-dimensional, 4-layer, 4-head backbone with 3.406M parameters and explicit continuous-property supervision	Compact numerical representation and reduced model-scale requirements for small-scale alloy data

Compared with existing approaches, GPT-lite emphasizes the combination of lightweight architecture, numerically aware representation, explicit continuous-property supervision, and constraint-aware candidate evaluation for small-scale alloy design.

3.2 Numerically Aware Tokenizer: EnhancedTokenizer

Aluminum alloy sequences differ from ordinary natural language text, as their key information is mainly concentrated in elemental symbols, continuous numerical values, decimal points, positive and negative signs, delimiters, and field boundaries. If a conventional character-level or byte-level representation is adopted, continuous compositional values are likely to be split into long token sequences [41,42]. This increases the input length and weakens the model's ability to represent trace elements and fine-grained compositional variations. Existing studies on byte-level language models have shown that although byte-level representations provide strong coverage, they usually lead to longer sequences and higher computational costs [43]. In materials generation and composition representation tasks, the choice of representation also affects the model's ability to learn the composition space and control generation.

Based on these characteristics, this paper designs EnhancedTokenizer to process structured aluminum alloy sequences. The tokenizer preserves task identifiers, field symbols, elemental symbols, and numerical characters, and incorporates decimal points, sign symbols, and commonly used delimiters into the basic vocabulary to support the representation of continuous numerical compositions. Unlike a general byte-level tokenizer, EnhancedTokenizer is designed for the structured alloy grammar used in this study. Its objective is not to enlarge the vocabulary size, but to reduce meaningless segmentation, enabling the model to capture more complete composition and property information within a fixed context window.

However, the purpose of EnhancedTokenizer is to reduce numerical fragmentation rather than to convert the discrete language-model representation into a continuous numerical representation. The resulting token embeddings therefore do not explicitly guarantee that distances between numerical values are preserved in embedding space [44,45].

For an aluminum alloy sample, the structured sequence can be expressed as follows:

COMP: [Ag = 0.0000, Al = 0.9917, Fe = 0.0017, ..., Zr = 0.0000] → PROP: [E = 32.3265, T = 100.1978, Y = 35.2727]

The tokenization efficiency is evaluated using the following metrics:

$$Fertility = \frac{1}{N} \sum i \frac{L_i}{C_i} \quad (1)$$

$$Compression = \frac{1}{N} \sum i \frac{C_i}{L_i} \quad (2)$$

$$Attention \ cost \ proxy = \frac{1}{N} \sum i L_i^2 \quad (3)$$

$$Num - fertility = \frac{1}{N} \sum i \frac{L_{i,num}}{C_{i,num}} \quad (4)$$

where Fertility denotes the number of tokens per character. A lower value indicates a more compact sequence representation. Compression denotes the average number of characters carried by each token. A higher value indicates better compression efficiency. The attention cost proxy approximates the attention computation cost using (L_i^2) . Num-fertility is calculated only on numerical fragments and is used to measure the tokenizer's representation efficiency for continuous numerical values.

3.3 Property Prediction Branch and Joint Optimization with the Regression Head

The property prediction branch of GPT-lite corresponds to the Pred task, where the input is the alloy composition and the output is the three standardized mechanical properties. This task can be expressed as follows:

Pred: COMP → PROP

where COMP denotes the composition sequence constructed according to a fixed elemental order, and PROP denotes the three-dimensional property vector consisting of Elongation, Tensile strength, and Yield strength. The purpose of this branch is not to train an independent optimal property predictor, but to enable the GPT-lite backbone to acquire property-aware modeling capability while learning the distribution of alloy sequences.

Given the (i) -th Pred sequence (x_i) , the autoregressive backbone outputs a sequence of hidden states:

$$H_i = GPT_{\theta}(x_i) = [h_1, h_2 \dots h_T] \quad (5)$$

where θ denotes the parameters of the GPT-lite backbone, and (T) denotes the sequence length. To prevent the regression head from directly accessing ground-truth property tokens and causing label leakage, the regression input is restricted to the prefix part before the PROP field. Let (t_i) denote the position of the last valid token in the prefix. The last hidden state can then be represented as (h_i, t_i) , and the mean hidden state over the prefix interval can be expressed as:

$$\bar{h}_i = \frac{1}{t_i} \sum_{j=1}^{t_i} h_{i,j} \quad (6)$$

This paper adopts last-mean pooling, where the last prefix hidden state is concatenated with the mean prefix hidden state:

$$Z_i = [h_{i,t_i}, \bar{h}_i] \quad (7)$$

The regression head then outputs the predicted values of the three mechanical properties based on (z_i):

$$\hat{y}_i = f_{\varphi} z_i \quad (8)$$

In the main GPT-lite training protocol, the prefix-aware regression head is jointly optimized with the language-modeling objective during Stage 1. For the dedicated property-prediction comparison in Section 4.4, an additional prefix-only regression fine-tuning stage is applied to make the evaluation condition closer to the prefix-only prediction scenario. The structure of the property prediction branch is illustrated in Fig. 2.

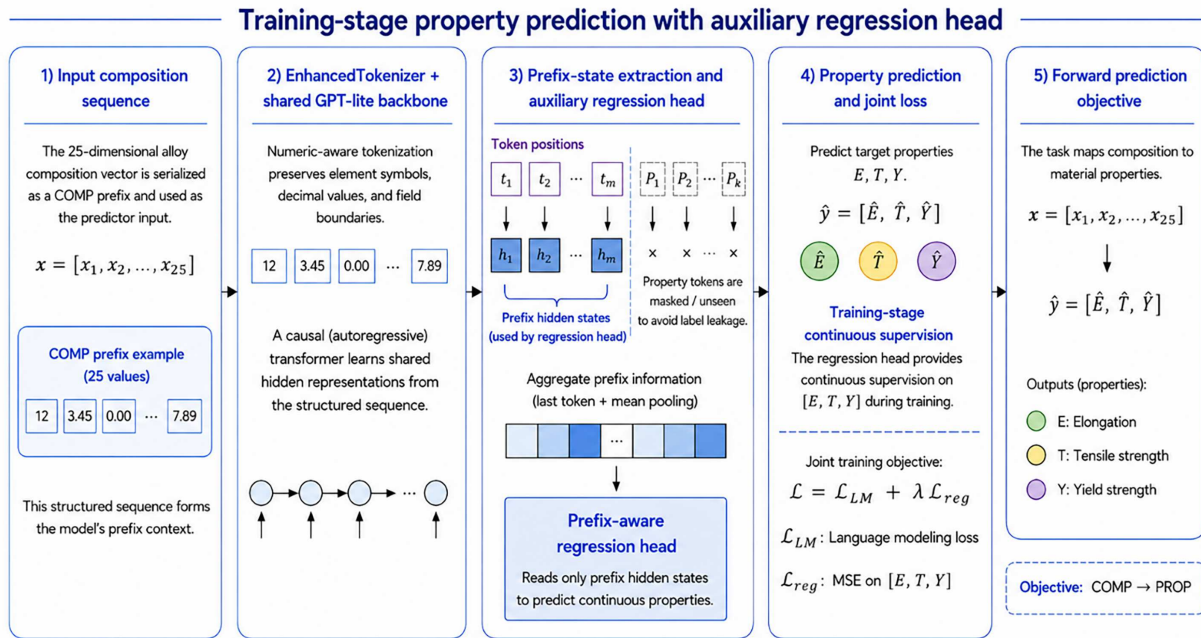


Figure 2: Structure of the property prediction branch.

Fig. 2 summarizes the prefix-aware property-prediction branch. Only hidden states before the PROP field are pooled and passed to the regression head, preventing direct access to ground-truth property tokens while providing continuous-property supervision to the shared GPT-lite backbone.

The joint training objective of GPT-lite consists of the language modeling loss and the regression loss:

$$\mathcal{L} = \mathcal{L}_{LM} + \lambda \mathcal{L}_{reg} \quad (9)$$

where $\mathcal{L}_{LM}$ denotes the autoregressive language-modeling loss, $\mathcal{L}_{reg}$ denotes the regression loss for the three target properties, and λ denotes the regression-loss weight.

$$\mathcal{L}_{reg} = \frac{1}{N} \sum_i \|\hat{y}_i - y_i\|^2 \quad (10)$$

When $\lambda = 0$, the model degenerates into a pure language model and only learns sequence reconstruction. When $\lambda > 0$, the property supervision signal is back-propagated to the shared representation space through the regression head, enabling the backbone model to learn representations related to target properties. In Section 4.3, a dedicated λ -sensitivity experiment further evaluates how the regression-loss weight affects

not only forward property-aware learning and language modeling, but also inverse target-conditioned composition generation directly.

3.4 Inverse Design and Candidate Evaluation Procedure

The inverse composition design branch of GPT-lite corresponds to the Gene task, where the input is the target property vector and the output is the candidate composition vector. This task can be expressed as follows:

Gene: PROP $\rightarrow$ COMP

The corresponding serialized form is:

{Task:Gene} PROP: [E, T, Y] $\rightarrow$ COMP: [$x_1, x_2, \dots, x_{25}$]

where (E), (T), and (Y) denote the standardized elongation, tensile strength, and yield strength, respectively, and (x_1) to (x_{25}) denote the 25-dimensional candidate composition arranged according to the fixed elemental order. The core objective of the inverse design stage is not to generate a single seemingly reasonable example, but to generate a set of candidate compositions conditioned on given target properties and then select the candidates closest to the target properties through a unified evaluation protocol.

In the generation stage, GPT-lite performs autoregressive sampling according to the target property prompt and obtains multiple candidate composition texts. The generated textual candidates are then passed to a strict parsing procedure. Strict parsing only accepts outputs that conform to the structure of COMP: [$x_1, x_2, \dots, x_{25}$] and requires that all 25 finite numerical values can be successfully parsed. If parsing fails, the dimensionality does not match, or non-finite values are present, the candidate is regarded as invalid. The parsed raw compositions are first subjected to a raw validity check, denoted as the OFF validity check in this study, which determines whether a candidate already satisfies the basic compositional constraints without constraint repair. On this basis, this study further introduces the ON-base soft constraint. Specifically, after a candidate has been successfully parsed, boundary clipping and normalization correction are applied to the raw numerical values:

$$\dot{c} = \text{clip}(c, 0, 1) \quad (11)$$

$$c_{on,i} = \frac{\dot{c}_i}{\sum_j \dot{c}_j} \quad (12)$$

The ON-base soft constraint clips parsed elemental fractions to [0, 1] and subsequently normalizes the composition to $\sum x_i = 1$. It is intended to repair minor boundary or composition-sum violations rather than generate new compositions. Candidates for which all elemental values become zero after clipping remain invalid. The complete inverse-design procedure is illustrated in Fig. 3.

Fig. 3 summarizes the complete inverse-design workflow from conditional generation to candidate ranking. Parsed candidates are corrected using the ON-base constraint, evaluated by the external MLP surrogate, and ranked according to their property errors. The surrogate is used only for external candidate evaluation and is independent of the auxiliary GPT regression head.

Let the target property vector be y^* , and let the surrogate-predicted property vector of a candidate composition be $\hat{y}$. The relative error of the k -th property is defined as:

$$r_k = \frac{|\hat{y}_k - y_k|}{\max(|y_k|, \epsilon)} \quad (13)$$

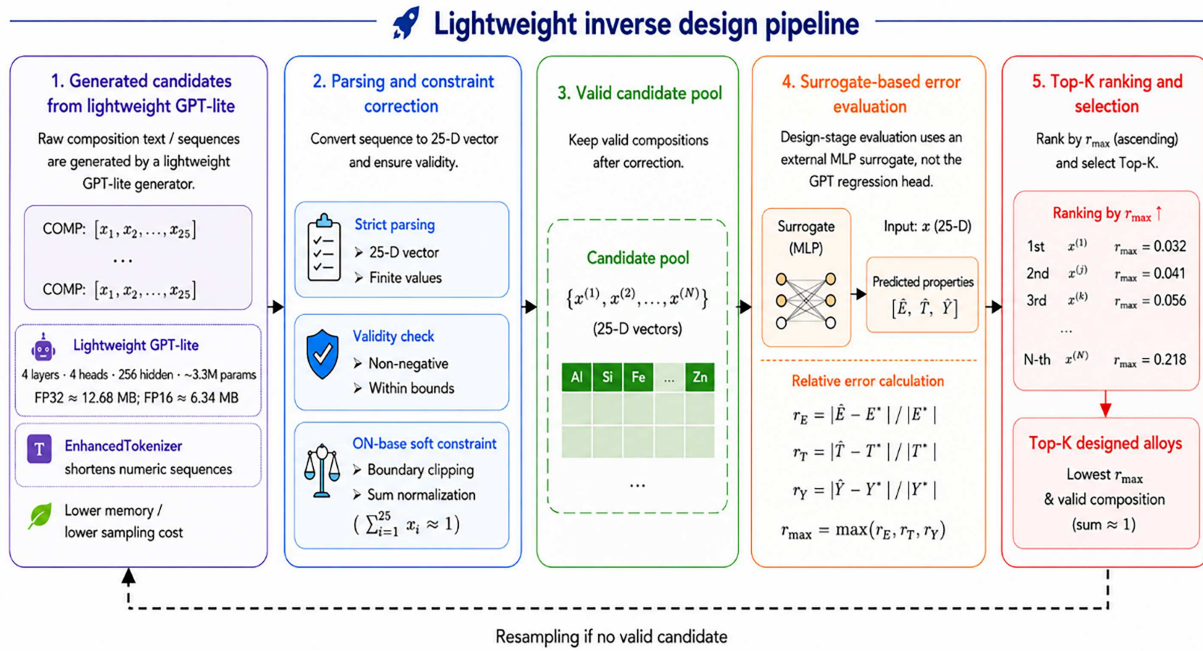


Figure 3: Inverse composition design pipeline of GPT-lite.

This study adopts the maximum relative error among the three properties as the candidate ranking criterion:

$$r_{\max} = \max(r_{\text{elongation}}, r_{\text{tensile}}, r_{\text{yield}}) \quad (14)$$

For each target property vector, the candidate with the smallest $r_{\max}$ is selected from the generated candidate set as the best design result under that target. This evaluation strategy avoids the situation in which a large error in one property is masked by the average error, and is therefore more consistent with the requirements of multi-property collaborative design.

It should be noted that the numerical validity and surrogate-based property evaluation used in the present workflow do not constitute direct verification of metallurgical feasibility or manufacturability. These aspects are further discussed in [Section 4.5.5](#).

4 Experimental Results and Discussion

Unless otherwise specified, the principal inverse-design performance results of GPT-lite are those obtained under the unified four-method comparison protocol reported in [Section 4.5.2](#). Several supplementary robustness and sensitivity experiments use dedicated matched generation seeds, nested candidate streams, or multiple independently trained models to isolate specific factors. Therefore, the absolute inverse metrics obtained in these controlled experiments are intended primarily for within-experiment comparison and should not be interpreted as replacements for the main four-method comparison results.

4.1 Analysis of the Lightweight Structural Complexity and Computational Efficiency of GPT-Lite

To evaluate the lightweight characteristics of GPT-lite, structural complexity and measured computational efficiency were examined jointly. [Table 3A](#) compares model dimensions, parameter count, FP32 model

size, and a structural computational proxy, whereas Table 3B reports hardware-level training and inference efficiency for the final GPT-lite implementation.

Table 3: Structural complexity and computational efficiency of GPT-lite. (A) Structural complexity; (B) Hardware-level computational efficiency.

(A)						
Model	n_embd	n_layer	n_head	Params (M)	FP32 Size (MB)	Compute Proxy
AlloyGPT-style large backbone	1024	36	16	453.859	1731.34	9.66×10^9
Proposed GPT-lite backbone only	256	4	4	3.258	12.43	6.71×10^7
Proposed GPT-lite + regression head	256	4	4	3.406	12.99	6.71×10^7
(B)						
Evaluation	Metric			Measured Value		
Training	Full 8-epoch wall-clock time			2.150 min		
Training	Peak GPU memory			466.65 MB allocated/1610.81 MB device-used		
Training	GPU energy consumption			4.143 Wh		
Forward inference	Latency, batch = 1			5.551 ms/sample		
Forward inference	Latency, batch = 32			0.817 ms/sample		
Forward inference	Throughput, batch = 32			1223.9 samples/s		
Inverse generation	Latency, B = 300			6.275 s/target		
Inverse generation	Latency per candidate			20.915 ms/candidate		
Inverse generation	Peak GPU memory			160.33 MB		

Note: PyTorch memory denotes peak tensor allocation, whereas NVML device memory additionally includes CUDA context, caching, and other device-level usage. Runtime and energy measurements are hardware- and implementation-dependent. The 453.859M AlloyGPT-style model in Table 3A is used only as a structural reference and is distinct from the 151.747M resource-scaled AlloyGPT-style baseline evaluated separately in Section 4.5.4.

The hardware benchmark was conducted on a single NVIDIA GeForce RTX 2060 GPU using CUDA 12.6 and PyTorch 2.12.1. The final GPT-lite configuration used $\lambda = 1$, a hidden dimension of 256, four Transformer layers, four attention heads, a maximum sequence length of 256, and eight training epochs. Training time, peak GPU memory, and GPU energy consumption were measured using a fresh training run, while forward and inverse inference latency were measured after GPU warm-up and synchronization.

For structural comparison, the compute proxy is defined as $n_{\text{layer}} \times \max_len \times n_{\text{embd}}^2$ and is used only as an approximate relative indicator rather than measured FLOPs.

Table 3A shows that the complete GPT-lite model contains 3.406M trainable parameters, compared with 453.859M for the AlloyGPT-style structural reference. GPT-lite therefore uses approximately 99.25% fewer parameters, while the structural computational proxy is reduced from 9.66×10^9 to 6.71×10^7 . The regression head adds only approximately 0.148M parameters to the 3.258M backbone, indicating limited additional structural cost.

Table 3B provides complementary hardware-level evidence. A fresh eight-epoch training run required 2.150 min on the RTX 2060, with 466.65 MB peak PyTorch-allocated memory and 4.143 Wh GPU energy consumption. Forward inference required 5.551 ms/sample at batch size 1 and 0.817 ms/sample at batch size 32, while inverse generation with B = 300 required 6.275 s/target. These measurements support the computational compactness of the current implementation, although runtime and memory remain hardware- and implementation-dependent.

4.2 Ablation Experiment on the Numerically Aware Tokenizer

To verify the representation efficiency of EnhancedTokenizer for structured aluminum alloy sequences, this study compares it with the CharWord baseline and the official byte tokenizer. Existing studies on byte-level language models have shown that byte-level representations provide strong coverage, but usually produce longer input sequences. In materials inverse design, the representation method also affects the model's ability to learn the composition space and control generation. Therefore, this study evaluates the encoding effects of different tokenizers using metrics such as average token length, compression ratio, attention cost proxy, and num-fertility. The tokenizer comparison results are summarized in Table 4.

Table 4: Representation efficiency and downstream performance of different tokenizers. (A) Representation efficiency; (B) Downstream prediction and inverse-design performance.

(A)					
Tokenizer	Vocab Size	Avg. Tokens/ Sample	Compression	Attn. Cost Proxy	Num. Fertility
EnhancedTokenizer	116	127.397	1.7181	16,246.65	0.5399
CharWord baseline	78	200.885	1.0896	40,354.92	1.0000
Official byte tokenizer	282	218.885	1.0000	47,910.77	1.0000
(B)					
Tokenizer	Forward Test $R^2 \uparrow$	Mean best_ max_rel $\downarrow$	P90 $\downarrow$	SR@0.05 $\uparrow$	ON-Base Validity $\uparrow$
CharWord	0.949441	0.043823	0.086076	0.72	0.679667
Official byte	0.969639	0.043394	0.078300	0.71	0.604367
EnhancedTokenizer	0.959841	0.039586	0.070607	0.74	0.631022

Note: Table 4B compares the three tokenizers under the same GPT-lite architecture, $\lambda = 1$, training seed, data split, 100 fixed held-out targets, B = 300, sampling settings, and external MLP surrogate.

Table 4A shows that EnhancedTokenizer provides the most compact representation, reducing the average sequence length to 127.397 tokens/sample compared with 200.885 for CharWord and 218.885 for the byte tokenizer. It also achieves the lowest numerical fertility and attention-cost proxy, indicating reduced numerical fragmentation and sequence cost.

Table 4B shows that tokenizer choice affects forward prediction and inverse generation differently. The byte tokenizer achieves the highest forward test R^2 (0.969639), whereas EnhancedTokenizer achieves the lowest mean best_max_rel (0.039586) and the highest SR@0.05 (0.74). Because target-conditioned inverse design is the primary objective of the present framework, EnhancedTokenizer is retained as the default tokenizer.

4.3 Sensitivity Analysis of the Joint Loss Weight for Forward and Inverse Tasks

The joint loss weight λ controls the relative contribution of the continuous property-regression objective to the shared GPT-lite representation. Evaluating λ only through forward property-prediction metrics is insufficient because stronger regression supervision may improve property-aware representations while simultaneously affecting the autoregressive sequence distribution used for inverse composition generation. Therefore, the sensitivity analysis was extended to evaluate both forward prediction and inverse target-matching performance directly.

For each $\lambda \in \{0, 1, 5, 10, 20\}$, the same GPT-lite architecture was trained using three model seeds (42, 43, and 44). The train/validation/test split and the 100 held-out inverse-design targets were fixed using seeds of 42. The shared Stage-1 model was jointly trained on both Pred and Gene sequences: the language-modeling loss was applied to both tasks, whereas the regression loss was applied only to Pred sequences through the regression mask. No Stage-2 regression-only fine-tuning was performed in this experiment so that the inverse sensitivity could be evaluated directly from the same jointly trained checkpoint. For inverse evaluation, each model generated $B = 300$ candidates per target using temperature = 0.9 and top-p = 0.95, and all candidates were evaluated using the same external MLP surrogate. Corresponding targets used matched generation seeds across different λ values to reduce variation caused by stochastic sampling. The results of the joint loss weight sensitivity analysis are summarized in Table 5.

Table 5: Sensitivity of forward and inverse performance to the joint loss weight λ .

λ	Forward Val $R^2 \uparrow$	Val LM Loss $\downarrow$	Mean best_max_rel $\downarrow$	P90 $\downarrow$	SR@0.05 $\uparrow$	SR@0.08 $\uparrow$	SR@0.10 $\uparrow$
0	0.0403 $\pm$ 0.0145	0.1139 $\pm$ 0.0004	0.03524 $\pm$ 0.00088	0.07430 $\pm$ 0.00387	0.780 $\pm$ 0.010	0.917 $\pm$ 0.015	0.953 $\pm$ 0.006
1	0.9568 $\pm$ 0.0134	0.1478 $\pm$ 0.0029	0.05591 $\pm$ 0.01926	0.12011 $\pm$ 0.04396	0.620 $\pm$ 0.171	0.800 $\pm$ 0.137	0.870 $\pm$ 0.092
5	0.9624 $\pm$ 0.0118	0.2154 $\pm$ 0.0100	0.12920 $\pm$ 0.04507	0.27326 $\pm$ 0.08271	0.333 $\pm$ 0.159	0.470 $\pm$ 0.156	0.553 $\pm$ 0.142
10	0.9678 $\pm$ 0.0114	0.2864 $\pm$ 0.0188	0.24557 $\pm$ 0.06075	0.49734 $\pm$ 0.13911	0.093 $\pm$ 0.045	0.233 $\pm$ 0.047	0.317 $\pm$ 0.049
20	0.9686 $\pm$ 0.0084	0.5044 $\pm$ 0.0676	0.16215 $\pm$ 0.00920	0.29521 $\pm$ 0.00506	0.097 $\pm$ 0.006	0.250 $\pm$ 0.020	0.360 $\pm$ 0.010

Table 5 shows a clear trade-off between forward property-aware learning and inverse generation. Without regression supervision ($\lambda = 0$), forward validation R^2 is only 0.0403 ± 0.0145 , although inverse target matching is strongest. Introducing $\lambda = 1$ increases forward R^2 to 0.9568 ± 0.0134 while retaining substantially better inverse performance than the larger λ settings.

Increasing λ to 5–20 provides only limited additional improvement in forward R^2 but substantially increases language-modeling loss and inverse target-matching error. Therefore, $\lambda = 1$ is retained not as the optimum for either task individually, but as a joint-task compromise between property-aware representation learning and autoregressive inverse generation.

4.4 Property Prediction Evaluation

4.4.1 Internal Comparison between GPT reg_head and MLP Surrogate

The purpose of the property prediction experiment is not to demonstrate that GPT reg_head outperforms MLP in a pure regression task, but to clarify their different roles in the proposed framework. GPT reg_head shares the representation space with the autoregressive backbone and is mainly used to verify whether GPT-lite can learn property-aware representations from structured alloy sequences. In contrast, MLP performs supervised regression directly based on numerical composition features and is therefore more suitable as an external surrogate in the subsequent inverse design stage. Existing studies on material property prediction [46,47] have shown that machine learning models based on material representations can form strong baselines in various material property prediction tasks [48]. Therefore, the comparison between GPT reg_head and MLP provides an empirical basis for evaluating whether MLP is suitable as a unified evaluator in the following experiments.

For the dedicated property-prediction comparison in this section, the GPT branch adopts a two-stage training strategy. In the first stage, joint optimization of the language modeling loss and regression loss is performed on complete Pred sequences. In the second stage, only the prefix is provided as input, and pure regression fine-tuning is conducted, making the training condition closer to the prefix-only prediction scenario during testing. The input of the regression head adopts last-mean pooling, where the hidden state

of the last token in the prefix is concatenated with the mean hidden state over the prefix interval for property prediction. The MLP baseline uses standardized compositional features as input and outputs three properties: Elongation, Tensile strength, and Yield strength. Both models are trained and tested under the same data split, and R^2 , MAE, and RMSE are used as evaluation metrics. The property prediction results of GPT reg_head and MLP are summarized in Table 6.

Table 6: Comparison of property prediction results between GPT reg_head and MLP.

Metric	GPT reg_head	MLP
R^2_{all}	0.9786	0.9915
MAE_{all}	9.3345	5.8865
RMSE_{all}	25.8707	16.2810
MAE_E	1.1534	0.8277
MAE_T	12.8612	8.6438
MAE_Y	13.9888	8.1879
R^2_E	0.9779	0.9881
R^2_T	0.9851	0.9918
R^2_Y	0.9717	0.9912

Table 6 shows that both models provide strong property prediction, while the numerical MLP achieves better overall regression performance. Its overall R^2 reaches 0.9915 compared with 0.9786 for the GPT reg_head, and it also produces lower MAE and RMSE across the three properties. Fig. 4 further shows a more concentrated predicted-vs.-true distribution for the MLP. Therefore, the MLP is retained as the independent external surrogate for evaluating inverse-generated candidates, whereas the GPT regression head remains an auxiliary property-aware supervision component of GPT-lite.

Fig. 4 further confirms that predictions from both models remain close to the ideal diagonal, while the MLP predictions are more concentrated. Together with Table 6, this supports the use of the MLP as the unified external evaluator in subsequent inverse-design experiments.

4.4.2 Independent External Validation on an Aluminum Alloy Dataset

To evaluate whether the learned composition–property representations can transfer beyond the original data source, an independent external validation experiment was conducted using the aluminum alloy dataset reported by Pfeiffer et al. [35]. The published dataset contains 14,884 aluminum alloy composition records and 1278 mechanical-property records extracted from scientific literature and US patents. Because the original data descriptor provides composition and property records as separate datasets and notes the difficulty of reliably associating arbitrary property entries with specific compositions, a conservative matching strategy was adopted in this study.

Specifically, an external property record was linked to a composition only when the two records originated from the same DOI, had exactly matching normalized row names, and the matching key corresponded to a unique composition candidate. Records with conflicting four-digit Aluminum Association designations were excluded. Repeated property measurements associated with the same exact 69-element source composition were then averaged to obtain one composition-level target. After filtering and strict matching, 75 property records were retained, corresponding to 40 unique external compositions from 29 source DOIs. After aggregation, all 40 retained composition-level samples contained the three target mechanical properties used for the present evaluation.

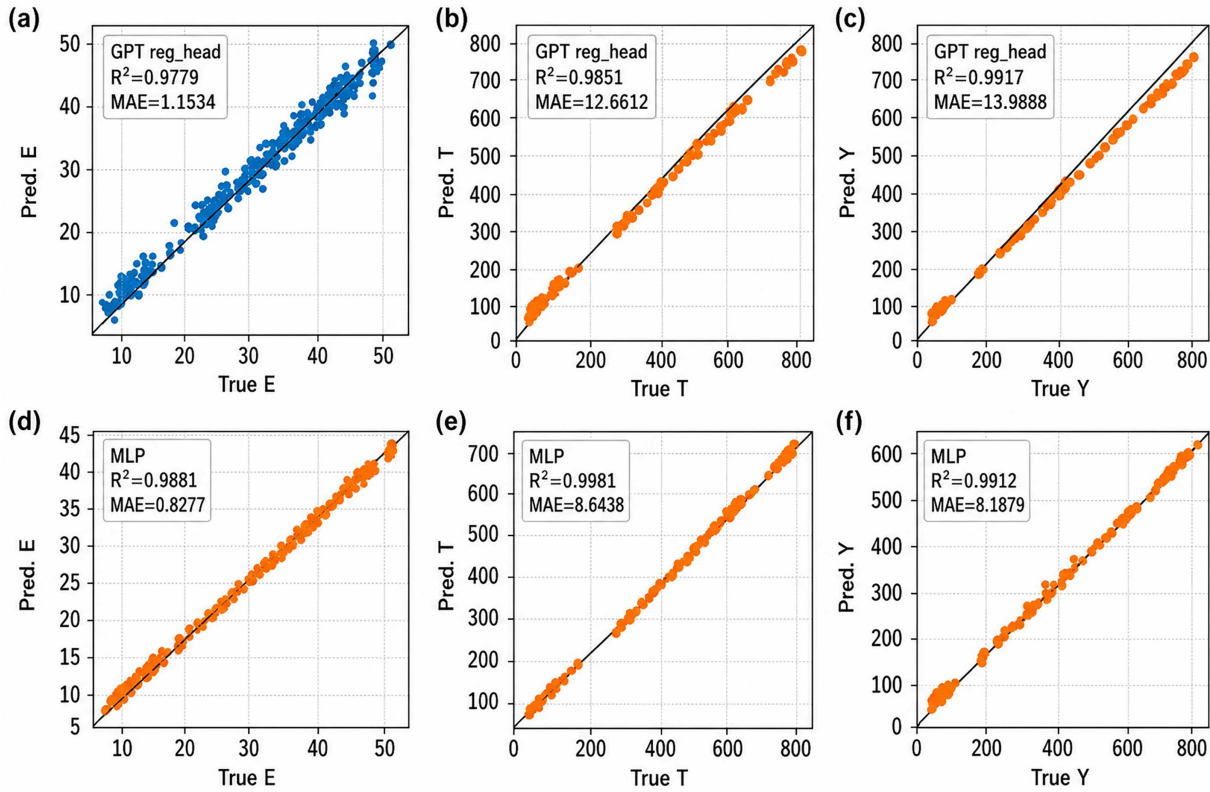


Figure 4: Test-set scatter plots for GPT reg_head and MLP surrogate. (a) Elongation prediction by GPT reg_head; (b) tensile strength prediction by GPT reg_head; (c) yield strength prediction by GPT reg_head; (d) elongation prediction by MLP surrogate; (e) tensile strength prediction by MLP surrogate; and (f) yield strength prediction by MLP surrogate.

External validation was performed in a strictly zero-shot setting. The final GPT-lite checkpoint trained on the original dataset was kept frozen, and the original training-fitted composition standardization and property scaler were retained. No external sample was used for model training, fine-tuning, calibration, scaler fitting, model selection, or target clipping. All 25 elemental variables required by GPT-lite were directly available in the external composition table. The 25-dimensional model composition accounted for 99.97% of the source composition on average. In addition, composition overlap with the original training set was explicitly audited before evaluation.

The zero-shot external validation results are summarized in Table 7. To ensure that the evaluation reflected cross-dataset transfer rather than memorization of previously observed compositions, the external compositions were additionally compared with the original training set in the normalized 25-dimensional composition space. None of the 40 external compositions was an exact training-set match under an L1 threshold of 10^{-6} , and only two were identified as near overlaps under an L1 threshold of 0.01. The mean nearest-training-composition L1 distance was 0.0354. Therefore, the external evaluation is not dominated by exact composition duplication with the original training data.

As shown in Table 7, GPT-lite retains predictive capability on the independent external dataset for all three mechanical properties. Tensile strength shows the strongest external performance, with an R^2 of 0.748, followed closely by yield strength with an R^2 of 0.740. Elongation achieves a lower but still positive R^2 of 0.420. These results indicate that the composition–property relationships learned from the original dataset

remain transferable to independently collected aluminum-alloy data, although the degree of transfer differs among the three target properties.

Table 7: Zero-shot external validation on the Pfeiffer et al. dataset.

Property	N	R ²	MAE	RMSE
Elongation	40	0.420	9.562%	11.708%
Tensile strength	40	0.748	104.194 MPa	143.691 MPa
Yield strength	40	0.740	92.979 MPa	122.538 MPa

Comparable R² levels have also been reported in previous machine-learning studies of aluminum alloys. Jatti et al. [49] reported yield-strength R² values of 0.7128 and 0.7546 for linear-regression and random-forest models, respectively, which are close to the external R² of 0.740 obtained in the present study. The same study also reported substantially lower elongation R² values of 0.3550 and 0.5312 for some model and processing settings, indicating that moderate predictive performance for elongation can occur in aluminum-alloy property prediction. In addition, Choi et al. [50] reported a test R² of approximately 0.752 for ultimate tensile strength in an industrial HPDC aluminum-alloy dataset, which is highly comparable to the external tensile-strength R² of 0.748 obtained here.

The property-distribution audit provides additional context for the comparatively weaker elongation performance. The mean elongation of the external samples is 13.71%, compared with 34.88% in the original training set, and only 60.0% of the external elongation values fall within the 5th–95th percentile range of the training distribution. In comparison, the corresponding proportions are 90.0% for tensile strength and 82.5% for yield strength. Therefore, the lower external R² for elongation is consistent with a larger distributional difference between the internal and external data for this property. In addition, GPT-lite currently uses elemental composition as the primary modeling input and does not explicitly incorporate temper, processing route, or microstructural information, which may further contribute to cross-source property variation.

Overall, the independent zero-shot validation provides additional evidence that GPT-lite is not limited to the original data source. In particular, the R² values of 0.748 and 0.740 for tensile strength and yield strength demonstrate relatively strong cross-dataset transfer, while the R² of 0.420 for elongation indicates moderate transfer under a larger distributional difference. Therefore, the external experiment supports measurable cross-dataset transfer within the evaluated aluminum-alloy domain, while also showing that the transfer strength remains property-dependent.

4.5 Inverse Composition Design Evaluation and Comparison

4.5.1 Unified Comparison Protocol

To further verify the effectiveness of GPT-lite in target-property-driven aluminum alloy composition design, this study compares it with three representative methods: the traditional search-based optimization method NSGA-II, the latent-variable generative model CVAE, and the adversarial generative model cWGAN-GP. NSGA-II represents the multi-objective search paradigm of “predictor + optimizer”; CVAE represents the conditional latent-variable generation paradigm [51,52]; and cWGAN-GP represents the adversarial conditional generation paradigm [53,54]. These methods provide representative comparisons across search-based optimization and deep generative modeling. In addition, to further incorporate recent language-model-based alloy design approaches, a separate matched comparison with an AlloyGPT-style autoregressive LLM baseline inspired by AlloyGPT [32] is provided in Section 4.5.4. Because this LLM

comparison follows a dedicated matched generation protocol, its results are reported separately rather than being directly merged with the original four-method comparison.

To ensure a fair comparison, all four methods use the same data preprocessing rules, the same 25-dimensional elemental composition space, the same training/validation/test split, and the same 100 fixed held-out target property vectors. Each method generates the same number of candidate compositions for each target, with the candidate budget uniformly set to ($B = 300$). All generated candidates are evaluated by the same external MLP surrogate for property back-prediction. This setting avoids additional bias caused by differences in target sets, candidate budgets, or evaluators.

This study uses `best_max_rel` as the main evaluation metric for inverse design. For each target property vector, the candidate set generated by a given method is first evaluated using the surrogate model. The relative errors of each candidate across the three properties are then calculated, and the candidate with the smallest maximum relative error is selected as the best candidate for that target. Finally, the mean, standard deviation, median, and P90 of `best_max_rel` are calculated over 100 targets. In addition, `SR@0.05`, `SR@0.08`, and `SR@0.10` are reported, representing the proportion of targets for which the best candidate satisfies the relative error threshold for all three properties. This evaluation strategy is stricter than reporting only the average error, because it requires the candidate to simultaneously satisfy the targets of elongation, tensile strength, and yield strength.

4.5.2 Target Matching and Candidate Validity

The target-matching and candidate-validity results are summarized in [Table 8](#).

Table 8: Inverse-design performance and candidate validity. (A) Target-matching performance; (B) Candidate validity and repair dependence.

(A)							
Method	Mean <code>best_max_rel</code>	Std	Median	P90	<code>SR@0.05</code>	<code>SR@0.08</code>	<code>SR@0.10</code>
NSGA-II	0.112511	0.032240	0.103416	0.211702	0.31	0.43	0.49
GPT-lite	0.029679	0.026312	0.019472	0.064254	0.85	0.97	0.98
CVAE	0.051511	0.035174	0.041470	0.097634	0.62	0.84	0.90
cWGAN-GP	0.032240	0.029915	0.020811	0.073137	0.80	0.91	0.96
(B)							
Method	Mean Parse Rate		Mean Off-Valid Rate		Mean Fix-Base Rate		
NSGA-II	1.0000		0.9993330		0.00		
GPT-lite	1.0000		0.890267		0.55		
CVAE	1.0000		0.000000		1.00		
cWGAN-GP	1.0000		0.993300		0.00		

Note: proportion of target pools containing at least one candidate requiring repair. OFF-validity is a strict binary constraint-compliance metric. For CVAE, all 30,000 raw candidates contained at least one negative component, while only 3.84% contained a component greater than 1. Nevertheless, 98.77% of the composition sums were within ± 0.01 of 1.0. After clipping and normalization, projected validity reached 1.0000, with mean and P90 L1 correction magnitudes of 0.008415 and 0.011961, respectively. Therefore, the zero raw-validity rate mainly reflects systematic boundary violations rather than large composition-sum errors.

Candidate validity complements rather than replaces target-matching performance. NSGA-II and cWGAN-GP satisfy the raw compositional constraints almost directly, whereas GPT-lite combines relatively high raw validity with the strongest target matching in Table 8A. CVAE depends more strongly on ON-base correction; however, the numerical diagnosis in the note to Table 8B shows that the required corrections are generally small and are primarily associated with boundary violations.

Table 8A shows that GPT-lite achieves the lowest mean best_max_rel (0.029679) and P90 (0.064254) among the four methods. Its SR@0.05 reaches 0.85, compared with 0.80 for cWGAN-GP, 0.62 for CVAE, and 0.31 for NSGA-II. The corresponding SR@0.08 and SR@0.10 values of 0.97 and 0.98 further indicate that GPT-lite maintains strong target alignment across increasingly relaxed tolerances. The distribution of best_max_rel across the four inverse-design methods is further illustrated in Fig. 5.

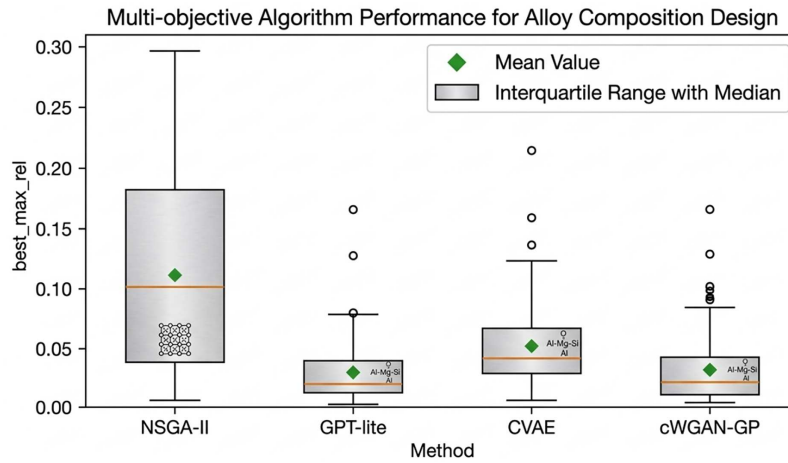


Figure 5: Distribution of best_max_rel for four inverse design methods.

4.5.3 Quantitative Analysis of Generated Composition Quality

Although target-matching metrics quantify whether at least one high-quality composition can be identified for each target, they do not fully characterize the overall quality of the generated candidate pool. In particular, an inverse-design model may achieve a low best_max_rel while repeatedly generating highly similar compositions or reproducing compositions already present in the training set. Candidate validity has already been quantitatively evaluated under the unified cross-method protocol in Section 4.5.2. Therefore, the present analysis complements that evaluation by examining the complete GPT-lite candidate pools in terms of uniqueness, novelty, and internal diversity. The analysis uses the selected GPT-lite model with $\lambda = 1$ and seed 42, the same 100 fixed held-out targets, and a candidate budget of $B = 300$ for each target.

Uniqueness is calculated after ON-base projection by grouping compositions that become identical after rounding each elemental fraction to a fixed numerical precision. Two precisions are considered: 10^{-4} and 10^{-3} . The uniqueness rate is defined as the number of distinct projected compositions divided by the number of projected-valid candidates. Novelty is evaluated against the projected training-composition set using the nearest-neighbor L1 distance in the projected, unstandardized composition space. A candidate is regarded as an exact copy when its nearest-training L1 distance is no greater than 10^{-6} . In addition, a stricter distance-based novelty criterion is reported using a nearest-training L1 distance greater than 0.02. Internal diversity is quantified by the mean pairwise L1 distance among projected-valid candidates within each target-conditioned candidate pool. Duplicates are retained in the primary diversity calculation so that repeated

generation appropriately reduces the diversity score; a supplementary diversity value calculated using unique candidates only is also reported. The quantitative candidate-quality results are summarized in Table 9.

Table 9 shows moderate candidate uniqueness, with unique rates of 0.5940 and 0.5091 at 10^{-4} and 10^{-3} precision, respectively. The exact novelty rate reaches 98.68%, indicating that most generated candidates are not exact reproductions of the training samples. However, under the stricter $L1 > 0.02$ criterion, the novelty rate decreases to 43.91%, showing that many generated candidates still remain relatively close to the observed training-composition space.

Table 9: Quantitative quality analysis of GPT-lite-generated alloy compositions.

Quality Dimension	Metric	Value
Uniqueness	Unique rate at 10^{-4} precision	0.5940
Uniqueness	Unique rate at 10^{-3} precision	0.5091
Novelty	Mean nearest-training L1 distance	0.07287
Novelty	Exact novelty rate, $L1 > 10^{-6}$	0.98675
Novelty	Strict distance novelty, $L1 > 0.02$	0.43906
Diversity	Mean pairwise L1 distance	0.11503
Diversity	Mean pairwise L1 among unique candidates	0.12224

The mean pairwise L1 distance is 0.11503 and increases to 0.12224 after duplicate removal, indicating measurable internal compositional diversity. Therefore, GPT-lite largely avoids exact memorization while maintaining moderate uniqueness and diversity, although exploration far from the training distribution remains more limited.

4.5.4 Comparison with a Recent Alloy-Specific LLM Baseline

To further strengthen the comparison with recent language-model-based materials design methods, an additional alloy-specific autoregressive language-model baseline was constructed with reference to AlloyGPT [33]. The published AlloyGPT framework concurrently performs forward prediction and inverse alloy design using structured alloy text and an autoregressive GPT architecture. Because the official pre-trained model and its original training data are not openly available for unrestricted direct reuse, the present study does not claim to reproduce the official AlloyGPT checkpoint. Instead, a transparent resource-scaled reimplementation, denoted as AlloyGPT-style LLM (resource-scaled), was trained from scratch on the same aluminum alloy dataset used for GPT-lite.

The resource-scaled baseline retains the main autoregressive language-modeling characteristics used for the comparison, including a character-word numerical representation, mixed Pred/Gene sequence training, and pure causal language-model optimization. The model uses a hidden dimension of 1024, 12 Transformer layers, 16 attention heads, a vocabulary size of 256, and a context length of 320, resulting in 151.747M trainable parameters. For a controlled comparison with GPT-lite, both models use the same 70/10/20 data split, the same 100 fixed held-out target property vectors, the same external MLP surrogate, the same candidate budget of $B = 300$, temperature = 0.9, top-p = 0.95, the same ON-base compositional correction, and the same maximum-relative-error ranking criterion. The generation random seed is also matched for corresponding targets. The matched comparison results are summarized in Table 10.

[Table 10](#) demonstrates an accuracy–efficiency trade-off. The AlloyGPT-style baseline achieves lower mean best_max_rel and higher SR@0.05, whereas GPT-lite uses approximately 44.6 times fewer parameters and a substantially smaller checkpoint. Under the matched raw-generation benchmark, GPT-lite is approximately 2.74× faster. Therefore, the comparison does not support universal accuracy superiority for GPT-lite; instead, it supports its positioning as a substantially more compact alternative with competitive inverse-design performance.

Table 10: Comparison between GPT-lite and the AlloyGPT-style LLM baseline. (A) Model scale and inverse target matching; (B) Matched generation efficiency.

(A)							
Method	Parameters (M)	Checkpoint (MB)	Mean best_max_rel ↓	P90 ↓	SR@0.05 ↑	SR@0.08 ↑	SR@0.10 ↑
GPT-lite	3.406	13.034	0.039586	0.070607	0.74	0.92	0.95
AlloyGPT-style LLM (resource-scaled)	151.747	579.924	0.029719	0.066194	0.85	0.95	0.96
(B)							
Method	Generation Time (s/target)			Peak GPU Memory			
GPT-lite	10.014			619 MB			
AlloyGPT-style	27.474			782 MB			

Note: [Table 10](#) uses the dedicated matched LLM comparison protocol. Panel B reports raw generation using 64 candidates per target and batch size 8; these runtime values should not be directly substituted for the hardware benchmark in [Table 3B](#).

4.5.5 Practical Manufacturability and Metallurgical Feasibility

The inverse-design results primarily establish computational rather than manufacturing feasibility. Numerically valid compositions satisfy the parsing, elemental-boundary, and composition-sum constraints used in the present workflow, but these conditions do not guarantee metallurgical stability or practical processability. Aluminum-alloy properties also depend strongly on phase constitution, precipitation behavior, microstructure, and processing history [55], none of which are explicitly included in the current composition–property dataset. Therefore, generated compositions should be treated as computational candidates requiring further metallurgical screening.

Manufacturability is also process-dependent. Casting, wrought processing, and additive manufacturing impose different requirements related to solidification, phase formation, precipitation response, formability, and hot-cracking susceptibility. Although the candidate-quality analysis in [Section 4.5.3](#) indicates that most GPT-lite outputs are not exact training copies and many remain close to the observed composition domain, compositional proximity alone cannot establish physical feasibility. Thermodynamic and process-specific screening is therefore required before experimental implementation.

A practical workflow can use GPT-lite as a front-end candidate generator, followed by CALPHAD-based equilibrium or Scheil calculations to evaluate phase constitution and solidification behavior [56]. Process-specific criteria can then be applied before higher-fidelity simulation and experimental validation. Accordingly, GPT-lite is intended to reduce the initial search space and prioritize candidates rather than replace thermodynamic calculations or experimental qualification.

4.6 Robustness Analysis of GPT-Lite

Robustness was evaluated from three complementary perspectives: training randomness, training-data availability, and numerical input perturbation. For training-seed robustness, five independently trained GPT-lite models with seeds 42–46 were evaluated while keeping the data split, inverse-design targets, generation settings, and external surrogate fixed. For training-data fraction robustness, the original training split was reduced to 25%, 50%, and 75%, and each reduced-data setting was repeated using five subset sampling seeds (42–46). The model/training seed was fixed at 42 so that variation in this experiment primarily reflected differences in training-data availability and subset composition. Within each subset seed, the 25%, 50%, and 75% training subsets were constructed in a nested manner, while the validation set, test set, inverse-design targets, generation settings, and external evaluator remained fixed. The 100% setting was retained as a single full-data reference. Finally, robustness to numerical input perturbation was evaluated by applying multiplicative Gaussian noise of up to 5% to the test compositions without retraining the model. The robustness results are summarized in Table 11.

Table 11: Robustness analysis of GPT-lite under training and input perturbations. (A) Training-seed robustness; (B) Training-data fraction robustness; (C) Robustness to noisy composition inputs.

(A)							
Seed	Forward Val R ² ↑	Mean best_max_rel ↓	P90 ↓	SR@0.05 ↑	ON-Base Validity ↑		
42	0.9550	0.037888	0.073438	0.740	0.632		
43	0.9710	0.076073	0.161857	0.450	0.485		
44	0.9444	0.061829	0.154473	0.590	0.566		
45	0.9589	0.064580	0.124486	0.580	0.491		
46	0.9576	0.059296	0.171839	0.640	0.545		
Mean ± SD	0.9574 ± 0.0095	0.05993 ± 0.01389	0.13722 ± 0.03979	0.600 ± 0.105	0.544 ± 0.060		
(B)							
Training Fraction	Training Samples	Repeats	Forward Test R ² ↑	Mean best_max_rel ↓	P90 ↓	SR@0.05 ↑	ON-Base Validity ↑
25%	622	5	0.9056 ± 0.0146	0.2728 ± 0.0591	0.5490 ± 0.1302	0.086 ± 0.042	0.0885 ± 0.0387
50%	1244	5	0.9315 ± 0.0102	0.2034 ± 0.0593	0.4156 ± 0.1163	0.216 ± 0.038	0.2647 ± 0.0884
75%	1865	5	0.9511 ± 0.0163	0.0834 ± 0.0511	0.1917 ± 0.1394	0.528 ± 0.114	0.5112 ± 0.0948
100%	2487	1	0.9598	0.03835	0.07187	0.750	0.6314
(C)							
Input Noise	Forward Test R ² Mean ± SD ↑	R ² Retention vs. Clean	Mean Composition L1 Change	Mean Relative Prediction Shift	P90 Max Relative Prediction Shift		
0%	0.959841 ± 0.000000	1.000000	0.000000	0.000000	0.000000		
1%	0.957647 ± 0.000767	0.997714	0.000520	0.007850	0.028955		
3%	0.956665 ± 0.000624	0.996692	0.001558	0.010095	0.033592		
5%	0.956515 ± 0.001085	0.996535	0.002600	0.011219	0.036287		

Note: For the 25%, 50%, and 75% settings, values are reported as mean $\pm$ SD across five subset sampling seeds (42–46). The model/training seed was fixed at 42, and the reduced training subsets were nested within each subset seed. The 100% setting represents a single fixed full-data reference and was therefore not included in the repeated subset-sampling statistics.

Table 11A shows that forward validation R^2 remains relatively stable across the five training seeds (0.9574 $\pm$ 0.0095), whereas inverse target matching exhibits greater variability, with mean best_max_rel = 0.05993 $\pm$ 0.01389 and SR@0.05 = 0.600 $\pm$ 0.105. This indicates that forward property-aware learning is less sensitive to training initialization than stochastic inverse generation.

Table 11B shows a clear overall improvement with increasing training-data coverage across repeated subset sampling. Mean best_max_rel decreases from 0.2728 ± 0.0591 at 25% training data to 0.0834 ± 0.0511 at 75%, while SR@0.05 increases from 0.086 ± 0.042 to 0.528 ± 0.114 ; the full-data reference reaches 0.03835 and 0.750, respectively. Forward R^2 is less sensitive to reduced training data. The comparatively larger variation in the inverse metrics also indicates that inverse generation depends not only on data quantity but on the specific composition–property coverage of the training subset.

Table 11C shows limited degradation under moderate numerical input noise, with 5% perturbation retaining approximately 99.65% of the clean-input R^2 . Overall, the robustness experiments indicate comparatively stable forward property-aware learning, whereas inverse generation is more sensitive to training initialization and training-data coverage.

4.7 Inference-Time Sensitivity and Practical Guidance

Tokenizer sensitivity has already been evaluated together with representation efficiency in Section 4.2 and Table 4B. This section therefore focuses on two inference-time choices that directly affect candidate selection: the candidate generation budget and the ranking strategy.

The candidate generation budget B determines the number of inverse-design candidates explored for each target property vector. To isolate the effect of B from stochastic differences in independent sampling runs, the budget sensitivity experiment uses a nested generation protocol. Specifically, one stochastic candidate stream is generated for each target, and the candidate pools are defined cumulatively as $B = 50, 100, 200, 300$, and 500 , such that the smaller candidate pools are exact prefixes of the larger pools. Therefore, differences among budget settings primarily reflect the effect of increasing the number of explored candidates rather than differences between independently sampled candidate sets. The sensitivity results for candidate generation and ranking are summarized in Table 12.

Table 12: Sensitivity to candidate generation and selection settings. (A) Candidate generation budget; (B) Candidate ranking strategy.

(A)							
B	Mean best_max_rel ↓	P90 ↓	SR@0.05 ↑	SR@0.08 ↑	SR@0.10 ↑	Mean Latency (s/target) ↓	
50	0.070844	0.149980	0.49	0.70	0.76	1.786	
100	0.053235	0.103640	0.58	0.83	0.87	3.632	
200	0.042273	0.071874	0.69	0.90	0.95	7.113	
300	0.039586	0.070607	0.74	0.92	0.95	10.565	
500	0.034546	0.070507	0.79	0.93	0.98	17.184	
(B)							
Ranking Strategy	Mean Selected max_rel ↓	P90 max_rel ↓	SR@0.05 ↑	SR@0.08 ↑	SR@0.10 ↑	Mean Selected mean_rel ↓	Mean Selected L2_rel ↓
Mean-relative	0.044683	0.076461	0.66	0.90	0.93	0.023890	0.051008
L2-relative	0.041494	0.074593	0.71	0.91	0.94	0.024456	0.049604
Max-relative	0.039586	0.070607	0.74	0.92	0.95	0.027219	0.051773

Note: Panel A uses nested candidate streams ($50 \subset 100 \subset 200 \subset 300 \subset 500$), whereas Panel B evaluates the same $B = 300$ projected-valid candidate pools under different ranking criteria. The latency values in Panel A are intended for relative comparison within the nested-generation implementation; Table 3B remains the primary runtime benchmark of the final model.

Table 12A shows the expected accuracy–cost trade-off. Increasing B from 50 to 500 reduces mean `best_max_rel` from 0.070844 to 0.034546 and increases `SR@0.05` from 0.49 to 0.79, but also increases generation latency. $B = 300$ is therefore retained as a balanced default rather than as the accuracy-optimal setting.

Table 12B shows that the preferred ranking criterion depends on the design objective. Max-relative provides the lowest maximum-error metrics and the highest `SR@0.05`, whereas mean-relative and L2-relative optimize their corresponding aggregate error measures. Because the present task requires all three target properties to satisfy the tolerance simultaneously, max-relative is retained as the default strategy.

Accordingly, B controls the inference-time accuracy–cost trade-off, while the ranking criterion should be selected according to the desired definition of multi-property error.

5 Conclusion and Future Work

5.1 Conclusions

This study proposes GPT-lite, a lightweight generative framework for target-property-driven aluminum alloy composition design. The framework integrates structured Pred and Gene tasks, numerically aware tokenization, prefix-aware continuous-property supervision, compositional correction, and independent surrogate-based evaluation within a compact autoregressive architecture.

Under the unified inverse-design protocol, GPT-lite achieves a mean `best_max_rel` of 0.029679 and `SR@0.05`, `SR@0.08`, and `SR@0.10` values of 0.85, 0.97, and 0.98, respectively, outperforming NSGA-II, CVAE, and cWGAN-GP. Candidate-pool analysis further shows moderate uniqueness, measurable diversity, and a 98.68% exact novelty rate, indicating that strong target matching is not obtained simply through exact reproduction of training compositions.

The additional analyses clarify the role of the proposed design choices. EnhancedTokenizer reduces sequence length and numerical fragmentation, while $\lambda = 1$ provides a practical compromise between forward property-aware learning and inverse generation. Comparison with the AlloyGPT-style baseline reveals an accuracy–efficiency trade-off: the larger model achieves stronger target matching under the dedicated matched protocol, whereas GPT-lite uses substantially fewer parameters and provides faster generation.

Independent zero-shot validation on 40 external aluminum-alloy compositions yields R^2 values of 0.420, 0.748, and 0.740 for elongation, tensile strength, and yield strength, supporting cross-dataset transfer within the present aluminum-alloy domain. Robustness experiments further show stable forward prediction across training seeds and moderate input noise, while repeated subset-sampling results indicate that inverse generation is more dependent on sufficient and representative training-data coverage.

Overall, GPT-lite provides a compact and reproducible framework for computational aluminum alloy inverse design. The generated compositions should nevertheless be regarded as candidate formulations rather than experimentally validated alloys, and future work should incorporate processing information, physics-based screening, uncertainty estimation, and experimental validation.

5.2 Limitations and Future Work

The present results remain limited to a fixed 25-dimensional aluminum-alloy composition space and three mechanical properties: elongation, tensile strength, and yield strength. Extending GPT-lite to larger elemental spaces or additional properties would require corresponding changes to the tokenizer, sequence length, property representation, surrogate model, and candidate constraints, while potentially increasing data sparsity and multi-objective trade-offs. In addition, the current experiments are restricted to aluminum alloys; transfer to other alloy systems would require system-specific data, representations, constraints,

and independent validation. Therefore, the present results should not be interpreted as evidence of direct transferability to substantially larger composition spaces, richer property sets, or other alloy families.

A further limitation arises from representing chemical compositions as autoregressive text sequences. Alloy compositions have no intrinsic sequential ordering, whereas a standard Transformer is position-sensitive. The fixed elemental order used in this study provides deterministic serialization but does not guarantee permutation invariance [57]. In addition, continuous elemental fractions are ultimately represented by discrete tokens, so numerical proximity is not explicitly preserved in token or embedding space. EnhancedTokenizer reduces numerical fragmentation, and the regression objective introduces continuous-property supervision, but neither mechanism fully removes these representational limitations.

Future work should investigate representations that better reflect the mathematical structure of chemical mixtures. Possible directions include permutation-invariant set-based encoders, order-randomized training to reduce sensitivity to a particular serialization, element-identity embeddings paired directly with continuous concentration values, and hybrid generative architectures that predict elemental identities discretely while modeling concentrations through continuous outputs. More fundamentally, continuous generative models operating directly on constrained composition vectors or simplex-valued representations could avoid part of the mismatch between autoregressive language modeling and continuous alloy composition spaces. Such approaches would provide a useful direction for determining whether the current sequence formulation can be replaced or complemented by representations with stronger permutation and numerical inductive biases.

A further limitation is that the current model uses only elemental compositions and three mechanical properties and does not explicitly incorporate processing routes, heat-treatment conditions, phase information, or microstructural descriptors. Because these variables can strongly influence the properties and manufacturability of aluminum alloys, the generated candidates should be regarded as computational design candidates rather than experimentally validated alloy formulations. Moreover, the inverse-design results are primarily assessed through an external MLP surrogate and have not yet been validated through experimental testing or high-fidelity physical simulation. Future work should integrate processing and microstructural descriptors, thermodynamic or CALPHAD-based screening, uncertainty estimation, multi-surrogate consistency assessment, and experimental feedback to improve the physical reliability of candidate selection.

Finally, the current implementation follows a “generation–repair–evaluation–ranking” workflow rather than a fully closed-loop materials-design system. Future extensions could combine GPT-lite with active learning, uncertainty-guided sampling, physics-based screening, and experimental feedback. Such developments would allow the framework to be evaluated progressively across larger composition spaces, richer property sets, and multiple alloy systems while maintaining explicit control over model uncertainty and physical feasibility.

In summary, GPT-lite provides a feasible generative modeling framework for target-property-driven aluminum alloy composition design. With the further integration of processing information, experimental validation, and closed-loop optimization mechanisms, the proposed method has the potential to provide a more practically valuable technical route for data-driven alloy design.

Acknowledgement: This work was supported by the National Natural Science Foundation of China (52061020).

Funding Statement: This work was supported by the National Natural Science Foundation of China under Grant No. 52061020.

Author Contributions: Xingci Hou (first author) completed methodology, data curation, formal analysis, investigation, and writing—original draft preparation. Yingli Liu (corresponding author) completed conceptualization,

supervision, and writing—review & editing. Jiancheng Yin (corresponding author) completed conceptualization, supervision, and writing—review & editing. Haihe Zhou completed methodology, investigation, and writing—review & editing. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The internal modeling data used in this study were derived from the computational alloy candidates reported by Li and Birbilis [34], whose underlying aluminum-alloy training data are publicly available through the Mendeley Data repository released by Bhat, Barnard, and Birbilis [35] (doi:10.17632/b6br4yk6r3.1). The independent external validation data were obtained from the dataset reported by Pfeiffer et al. [36]. The processed data and code used in the present study are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

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

References

1. Cheng M, Fu CL, Okabe R, Chottrattanapituk A, Boonkird A, Hung NT, et al. Artificial intelligence-driven approaches for materials design and discovery. *Nat Mater*. 2026;25(2):174–90. doi:10.1038/s41563-025-02403-7.
2. Pyzer-Knapp EO, Manica M, Staar P, Morin L, Ruch P, Laino T, et al. Foundation models for materials discovery—current state and future directions. *npj Comput Mater*. 2025;11(1):61. doi:10.1038/s41524-025-01538-0.
3. Merchant A, Batzner S, Schoenholz SS, Aykol M, Cheon G, Cubuk ED. Scaling deep learning for materials discovery. *Nature*. 2023;624(7990):80–5. doi:10.1038/s41586-023-06735-9.
4. Zeni C, Pinsler R, Zügner D, Fowler A, Horton M, Fu X, et al. A generative model for inorganic materials design. *Nature*. 2025;639(8055):624–32. doi:10.1038/s41586-025-08628-5.
5. Hoang TD, Nhat CV, Dang NT, Son BT, Thuan NB, Nguyen TQ. AI-Driven inverse design of 3D-printed continuous fiber composites using multiscale homogenization and dual neural networks. *Int J Adv Manuf Technol*. 2026;144(1):515–39. doi:10.1007/s00170-026-17787-2.
6. Bagal V, Aggarwal R, Vinod PK, Priyakumar UD. MolGPT: molecular generation using a transformer-decoder model. *J Chem Inf Model*. 2022;62(9):2064–76. doi:10.1021/acs.jcim.1c00600.
7. Cao Z, Luo X, Lv J, Wang L. Space group informed transformer for crystalline materials generation. *Sci Bull*. 2025;70(21):3522–33. doi:10.1016/j.scib.2025.09.035.
8. Wei L, Li Q, Song Y, Stefanov S, Dong R, Fu N, et al. Crystal composition transformer: self-learning neural language model for generative and tinkering design of materials. *Adv Sci*. 2024;11(36):2304305. doi:10.1002/advs.202304305.
9. Tang Y, Xu W, Cao J, Gao W, Farrell S, Erichson B, et al. A multimodal large language model for materials science. *Nat Mach Intell*. 2026;8(4):588–601. doi:10.1038/s42256-026-01214-y.
10. Yuan W, Chen G, Wang Z, You F. Empowering generalist material intelligence with large language models. *Adv Mater*. 2025;37(32):2502771. doi:10.1002/adma.202502771.
11. Zhang L, Deng H. NJmat 2.0: user instructions of data-driven machine learning interface for materials science. *Comput Mater Continua*. 2025;83(1):1–11.
12. Gao L, Lin J, Wang L, Du L. Machine learning-assisted design of advanced polymeric materials. *Acc Mater Res*. 2024;5(5):571–84. doi:10.1021/accountsmr.3c00288.
13. Xu P, Ji X, Li M, Lu W. Small data machine learning in materials science. *npj Comput Mater*. 2023;9(1):42. doi:10.1038/s41524-023-01000-z.
14. Li S, Yue X, Li Q, Peng H, Dong B, Liu T, et al. Development and applications of aluminum alloys for aerospace industry. *J Mater Res Technol*. 2023;27:944–83. doi:10.1016/j.jmrt.2023.09.274.
15. Sun Y. The use of aluminum alloys in structures: review and outlook. *Structures*. 2023;57(5):105290. doi:10.1016/j.istruc.2023.105290.
16. Dang B, Zhang X, Chen YZ, Chen CX, Wang HT, Liu F. Breaking through the strength-ductility trade-off dilemma in an Al-Si-based casting alloy. *Sci Rep*. 2016;6(1):30874. doi:10.1038/srep30874.
17. Chaudry UM, Hamad K, Abuhmed T. Machine learning-aided design of aluminum alloys with high performance. *Mater Today Commun*. 2021;26:101897. doi:10.1016/j.mtcomm.2020.101897.

18. Bhat N, Barnard AS, Birbilis N. Improving the prediction of mechanical properties of aluminium alloy using data-driven class-based regression. *Comput Mater Sci.* 2023;228(1):112270. doi:10.1016/j.commatsci.2023.112270.
19. Bhat N, Barnard AS, Birbilis N. Inverse design of aluminium alloys using multi-targeted regression. *J Mater Sci.* 2024;59(4):1448–63. doi:10.1007/s10853-023-09317-2.
20. Lee K, Song Y, Kim S, Kim M, Seol J, Cho K, et al. Genetic design of new aluminum alloys to overcome strength-ductility trade-off dilemma. *J Alloys Compd.* 2023;947(4):169546. doi:10.1016/j.jallcom.2023.169546.
21. Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput.* 2002;6(2):182–97. doi:10.1109/4235.996017.
22. Shahriari B, Swersky K, Wang Z, Adams RP, de Freitas N. Taking the human out of the loop: a review of Bayesian optimization. *Proc IEEE.* 2016;104(1):148–75. doi:10.1109/JPROC.2015.2494218.
23. Tian Y, Li T, Pang J, Zhou Y, Xue D, Ding X, et al. Materials design with target-oriented Bayesian optimization. *npj Comput Mater.* 2025;11(1):209. doi:10.1038/s41524-025-01704-4.
24. Fu N, Wei L, Song Y, Li Q, Xin R, Omee SS, et al. Material transformers: deep learning language models for generative materials design. *Mach Learn Sci Technol.* 2023;4(1):015001. doi:10.1088/2632-2153/acdcdd.
25. Liu Y, Cui Y, Zhou H, Lei S, Yuan H, Shen T, et al. Machine learning-based methods for materials inverse design: a review. *Comput Mater Contin.* 2025;82(2):1463–92. doi:10.32604/cmc.2025.060109.
26. Noh J, Kim J, Stein HS, Sanchez-Lengeling B, Gregoire JM, Aspuru-Guzik A, et al. Inverse design of solid-state materials via a continuous representation. *Matter.* 2019;1(5):1370–84. doi:10.1016/j.matt.2019.08.017.
27. Noh J, Gu GH, Kim S, Jung Y. Machine-enabled inverse design of inorganic solid materials: promises and challenges. *Chem Sci.* 2020;11(19):4871–81. doi:10.1039/d0sc00594k.
28. Lu S, Zhou Q, Chen X, Song Z, Wang J. Inverse design with deep generative models: next step in materials discovery. *Natl Sci Rev.* 2022;9(8):nwac111. doi:10.1093/nsr/nwac111.
29. Dan Y, Zhao Y, Li X, Li S, Hu M, Hu J. Generative adversarial networks (GAN) based efficient sampling of chemical composition space for inverse design of inorganic materials. *npj Comput Mater.* 2020;6(1):84. doi:10.1038/s41524-020-00352-0.
30. Türk H, Landini E, Kunkel C, Margraf JT, Reuter K. Assessing deep generative models in chemical composition space. *Chem Mater.* 2022;34(21):9455–67. doi:10.1021/acs.chemmater.2c01860.
31. Dong R, Song Y, Siriwardane EMD, Hu J. Discovery of 2D materials using transformer network-based generative design. *Adv Intell Syst.* 2023;5(12):2300141. doi:10.1002/aisy.202300141.
32. Ni B, Glaser B, Taheri-Mousavi SM. End-to-end prediction and design of additively manufacturable alloys using a generative AlloyGPT model. *npj Comput Mater.* 2025;11(1):294. doi:10.1038/s41524-025-01768-2.
33. Li Z, Birbilis N. NSGAN: a non-dominant sorting optimisation-based generative adversarial design framework for alloy discovery. *npj Comput Mater.* 2024;10(1):112. doi:10.1038/s41524-024-01294-7.
34. Bhat N, Barnard A, Birbilis N. Aluminium alloy dataset for supervised learning [dataset]. *Mendeley Data.* 2023:Version 1. doi:10.17632/b6br4yk6r3.1.
35. Pfeiffer OP, Liu H, Montanelli L, Latypov MI, Sen FG, Hegadekatte V, et al. Aluminum alloy compositions and properties extracted from a corpus of scientific manuscripts and US patents. *Sci Data.* 2022;9(1):128. doi:10.1038/s41597-022-01215-7.
36. Ward L, Dunn A, Faghaninia A, Zimmermann NER, Bajaj S, Wang Q, et al. Matminer: an open source toolkit for materials data mining. *Comput Mater Sci.* 2018;152:60–9. doi:10.1016/j.commatsci.2018.05.018.
37. Li K, DeCost B, Choudhary K, Greenwood M, Hattrick-Simpers J. A critical examination of robustness and generalizability of machine learning prediction of materials properties. *npj Comput Mater.* 2023;9(1):55. doi:10.1038/s41524-023-01012-9.
38. Li Q, Fu N, Omee SS, Hu J. MD-HIT: machine learning for material property prediction with dataset redundancy control. *npj Comput Mater.* 2024;10(1):245. doi:10.1038/s41524-024-01426-z.
39. Ma Y, Xu P, Li M, Ji X, Zhao W, Lu W. The mastery of details in the workflow of materials machine learning. *npj Comput Mater.* 2024;10(1):141. doi:10.1038/s41524-024-01331-5.
40. Wang AY, Kauwe SK, Murdock RJ, Sparks TD. Compositionally restricted attention-based network for materials property predictions. *npj Comput Mater.* 2021;7(1):77. doi:10.1038/s41524-021-00545-1.

41. Xue L, Barua A, Constant N, Al-Rfou R, Narang S, Kale M, et al. ByT5: towards a token-free future with pre-trained byte-to-byte models. *Trans Assoc Comput Linguist.* 2022;10:291–306. doi:10.1162/tacl_a_00461.
42. Clark JH, Garrette D, Turc I, Wieting J. Canine: pre-training an efficient tokenization-free encoder for language representation. *Trans Assoc Comput Linguist.* 2022;10:73–91. doi:10.1162/tacl_a_00448.
43. Yu L, Simig D, Flaherty C, Aghajanyan A, Zettlemoyer L, Lewis M. MEGABYTE: predicting million-byte sequences with multiscale transformers. *Adv Neural Inf Process Syst.* 2023;36:78808–23.
44. Spithourakis G, Riedel S. Numeracy for language models: evaluating and improving their ability to predict numbers. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics; 2018. p. 2104–15. doi:10.18653/v1/P18-1196.
45. Wallace E, Wang Y, Li S, Singh S, Gardner M. Do NLP models know numbers? Probing numeracy in embeddings. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: ACL; 2019. p. 5306–14. doi:10.18653/v1/d19-1534.
46. Choudhary K, DeCost B. Atomistic line graph neural network for improved materials property predictions. *npj Comput Mater.* 2021;7(1):185. doi:10.1038/s41524-021-00650-1.
47. Kamnis S, Delibasis K. High entropy alloy property predictions using a transformer-based language model. *Sci Rep.* 2025;15(1):11861. doi:10.1038/s41598-025-95170-z.
48. Xie T, Grossman JC. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys Rev Lett.* 2018;120(14):145301. doi:10.1103/PhysRevLett.120.145301.
49. Jatti VS, Saiyathibrahim A, Yadav A, Murali Krishnan R, Jayaprakash B, Kaushal S, et al. Predicting the tensile properties of heat treated and non-heat treated LPBFed AlSi₁₀Mg alloy using machine learning regression algorithms. *PLoS One.* 2025;20(6):e0324049. doi:10.1371/journal.pone.0324049.
50. Choi S, Kim S, Lee J, Choi J, Lee M, Kim J, et al. Data-driven design of HPDC aluminum alloys using machine learning and inverse design. *Metals.* 2026;16(1):99. doi:10.3390/met16010099.
51. Kingma DP, Welling M. Auto-encoding variational Bayes. In: *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*; 2014 Apr 14–16.; Banff, AB, Canada.
52. Sohn K, Lee H, Yan X. Learning structured output representation using deep conditional generative models. *Adv Neural Inf Process Syst.* 2015;28:3486–94.
53. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative adversarial networks. *Commun ACM.* 2020;63(11):139–44. doi:10.1145/3422622.
54. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC. Improved training of Wasserstein GANs. *Adv Neural Inf Process Syst.* 2017;30:5767–77.
55. Robson JD, Engler O, Sigli C, Deschamps A, Poole WJ. Advances in microstructural understanding of wrought aluminum alloys. *Metall Mater Trans A.* 2020;51(9):4377–89. doi:10.1007/s11661-020-05908-9.
56. Jung JG, Cho YH, Lee JM, Kim HW, Euh K. Designing the composition and processing route of aluminum alloys using CALPHAD: case studies. *CALPHAD.* 2019;64:236–47. doi:10.1016/j.calphad.2018.12.010.
57. Zaheer M, Kottur S, Ravanbakhsh S, Poczos B, Salakhutdinov RR, Smola AJ. Deep sets. *Adv Neural Inf Process Syst.* 2017;30:3391–401.