Token Energy Optimization (Semantic Cost Proxy) in GPT-2 using Counterfactual Semantic Budgets: MetabolicGPT-2

Abhijit Nayak
Charlotte, North Carolina, USA
Email: nabhijit787@gmail.com

Raj Bhowmik
Dublin, California, USA
Email: bhowmick.raj10@gmail.com

***Abstract*—Large language models (LLMs) are increasingly deployed in settings where generation quality must be balanced against strict resource or verbosity constraints. Existing approaches to constrained text generation typically operate at decoding time, or optimize coarse proxies such as output length, with limited visibility into the semantic importance of individual tokens. In this work, we present *MetabolicGPT-2*, a GPT-2–based language model that learns an explicit *token-level semantic cost proxy* and generates under user-specified *semantic budgets*. Each token is assigned a non-negative cost that combines a corpus-driven synthetic prior with a *counterfactual semantic cost* (CSC) computed from a sentence encoder by measuring embedding change when a token is removed. We cast budget control as constrained optimization: minimize language modeling loss subject to an expected total proxy-cost constraint. We implement this by augmenting GPT-2 with a cost head, learnable token-cost parameters, a Lagrangian dual variable, and an online affine calibration layer that aligns proxy costs to CSC at the sequence level. Experiments on WikiText-2 illustrate that learned costs correlate with semantic importance and that budget settings shift the model toward lower-cost continuations while largely preserving fluency. Throughout, the term "energy" refers to this semantic proxy cost rather than direct hardware joules.**

***Index Terms*—large language models, constrained text generation, energy modeling, semantic budgets, counterfactual explanations, GPT-2**

I. Introduction

The success of large language models (LLMs) has led to widespread deployment in interactive assistants, code generation, clinical NLP, and other settings where both *quality* and *cost* matter. Cost can be measured in many ways: latency, wall-clock runtime, energy per token, or even the cognitive effort required to consume a response. Recent work has begun to measure the energy footprint of LLM inference and to benchmark models by energy efficiency in realistic serving settings [9], [10], [11], [13]. At the same time, constrained decoding methods such as MUCOCO and COLD decoding show that generation can be steered to satisfy various linguistic or logical constraints without retraining the base model [1], [2].

Most of these approaches treat the language model as a *black box*: constraints are enforced either through decoding heuristics or by post-hoc filtering. In contrast, this paper explores an orthogonal direction: can we train an autoregressive language model that *internally* models a token-level notion of "energy" or "cost", and then use this energy to enforce *semantic* budgets during generation?

We introduce *MetabolicGPT-2*, a GPT-2 style transformer augmented with:

- a corpus-dependent *metabolic tokenizer* that assigns each vocabulary token a synthetic cost based on frequency, character length, and heuristic penalties for boilerplate tokens,
- a K-dimensional *cost head* and learnable token–cost parameters that define a non-negative proxy cost $c_\theta(x_t)$ for each token,
- a *counterfactual semantic cost* (CSC) labeler that estimates semantic importance by removing tokens and measuring the change in a sentence-encoder embedding,
- a *Lagrangian budget term* with dual variable λ that encourages the expected total cost to match a user-defined budget, and
- an online affine calibration layer that aligns proxy costs with CSC at the sequence level.

Qualitatively, MetabolicGPT-2 behaves like a language model with a "metabolism": each token consumes a portion of a budget, and the model is encouraged to allocate costs to semantically important tokens while respecting a global constraint. We train on WikiText-2 [6] and evaluate budget tracking using mean absolute error (MAE), mean absolute percentage error (MAPE), and hit rates within relative tolerance bands. Our current prototype learns a meaningful and interpretable token–cost structure, assigning low energy to frequent function words and higher energy to semantically rich or rare content words. It uses this learned notion of cost to modulate its generations in response to different budget settings.

II. Research Objective

We consider an autoregressive language model p_θ over token sequences $x = (x_1, \ldots, x_T)$ from a vocabulary $\mathcal{V}$, with parameters θ and standard factorization

$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t \mid x_{<t}). \tag{1}$$

Our goal is to equip this model with a token-level cost function $c_\theta : \mathcal{V} \to \mathbb{R}_{\geq 0}$ and to enforce an expected total budget constraint of the form

$$\mathbb{E}_{x\sim\mathcal{D}}\big[C_{\text{proxy}}(x;\theta)\big] \leq B, \tag{2}$$

where

$$C_{\text{proxy}}(x;\theta) = \sum_{t=1}^{T} c_\theta(x_t) \tag{3}$$

denotes total proxy cost and $B \in \mathbb{R}^K$ is a vector of target budgets for K cost dimensions.

Concretely, our objectives are:

1) Define a *synthetic energy model* over the vocabulary that provides a structured prior on token costs based on corpus statistics and simple heuristics.
2) Learn a token-level *semantic cost* signal by supervising the cost head with CSC labels derived from a sentence encoder.
3) Formulate and implement a Lagrangian training objective that trades off language modeling loss against deviations from a prescribed expected budget.
4) Calibrate the learned proxy cost to CSC via online affine regression so that user budgets can be expressed in semantically meaningful units.
5) Evaluate how well the trained model generates meaningful text under budget constraints.

III. Related Work

A. Constrained Text Generation

Controlled and constrained text generation has been widely studied. MUCOCO [1] casts controlled generation as continuous optimization in the space of hidden states, supporting multiple attribute constraints without retraining the base LM. COLD decoding [2] formulates constrained generation as sampling from an energy-based model over relaxed token sequences, optimized via Langevin dynamics. Subsequent surveys and code repositories catalogue a range of constrained decoding strategies [3].

These methods express constraints at the level of sequence-level attributes (e.g., sentiment, toxicity, topicality), and typically leave the internal representation of the LM untouched. Our work instead introduces a dedicated token cost head and treats budget control as a first-class training objective, closer in spirit to constrained optimization in structured prediction.

B. Counterfactual Token Importance

Counterfactual explanations provide local feature importance scores by searching for small changes that flip a model prediction. In text, token-level importance can be estimated through gradient-based methods, attention-based scores, or explicit counterfactual edits. TIGTEC [4] uses token-importance signals to guide counterfactual text edits, and follow-up work investigates counterfactual feature importance for interpretability [5]. These methods typically operate on a task-specific classifier, not on a generative model.

Our CSC labeler borrows the counterfactual idea but applies it to a sentence encoder: each token's importance is measured by the change in embedding when that token is deleted. We then train a generative model to produce internal cost estimates that approximate CSC while also satisfying budget constraints.

C. Energy and Efficiency in LLMs

A complementary line of work benchmarks and reduces the energy cost of LLM inference, reporting energy per token across models and hardware [9], [10], [11], [12], and improving efficiency via caching and reuse (e.g., StoreLLM [13]). These efforts are system-level, whereas we learn a token cost in semantic space to enforce budgets; a key future direction is aligning semantic budgets with measured hardware energy.

IV. Methodology

A. Notation and Problem Formulation

Let $x = (x_1, \ldots, x_T)$ be a sequence of tokens from vocabulary $\mathcal{V}$, and let $p_\theta(x_t \mid x_{<t})$ denote the next-token distribution. We introduce a non-negative proxy cost function $c_\theta(x_t) \geq 0$ and define

$$C_{\text{proxy}}(x;\theta) = \sum_{t=1}^{T} c_\theta(x_t), \tag{4}$$

as well as a true semantic cost

$$C_{\text{CSC}}(x) = \sum_{t=1}^{T} c_{\text{CSC}}(x_t), \tag{5}$$

where $c_{\text{CSC}}(x_t)$ is obtained from the CSC labeler described below.

The unconstrained language modeling loss is

$$L_{\text{LM}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[-\sum_{t=1}^{T} \log p_\theta(x_t \mid x_{<t})\right]. \tag{6}$$

We seek to minimize L_{LM} while enforcing the expected budget constraint in (2).

We apply a Lagrangian relaxation with non-negative dual variable $\lambda \in \mathbb{R}^K_{\geq 0}$:

$$\mathcal{L}(\theta,\lambda) = L_{\text{LM}}(\theta) + \lambda^\top\big(\mathbb{E}[C_{\text{proxy}}(x;\theta)] - B\big). \tag{7}$$

In practice, we add additional regularization terms and a CSC supervision loss, but this is the core structure.

B. Metabolic Tokenizer: Synthetic Energy Model

The *metabolic tokenizer* wraps a GPT-2 tokenizer and builds a simple synthetic energy model over the vocabulary:

1) It collects token counts from the training corpus to estimate a frequency distribution $f(v)$ over $v \in \mathcal{V}$.
2) It decodes each token to text and computes its character length $\ell(v)$.
3) It defines a base cost

$$\tilde{c}(v) = \alpha \cdot \frac{1}{f(v)+\epsilon} + \beta \cdot \ell(v), \tag{8}$$

for hyperparameters $\alpha, \beta > 0$ and a small ϵ.

4) Costs are normalized so that the mean cost across the vocabulary is approximately 1, and a minimum cost floor $c_{\min}$ (e.g., 0.3) is enforced to avoid almost free tokens.
5) Tokens whose decoded forms resemble section headings or metadata (e.g., “Summary”, “Biography”) receive an additional penalty to discourage pathological generation that exploits cheap boilerplate.

This yields a synthetic cost table $c_{\text{syn}}(v)$ that acts as a structured prior: frequent and short tokens are relatively cheap, and rare or long tokens are relatively expensive. The learned cost head can then refine this prior using CSC supervision.

C. Counterfactual Semantic Cost (CSC)

We define CSC using a sentence encoder (*”sentence-transformers/all-MiniLM-L6-v2”*) $f(\cdot)$ (e.g., a transformer trained for semantic similarity). For a text $x = (x_1, \ldots, x_T)$ and token pieces, pieces$[t]$ corresponding to x_t:

1) Compute the embedding of the full text:

$$e_{\text{full}} = f(x). \tag{9}$$

2) Form a counterfactual text x_{-t} by removing the token at position t and concatenating the remaining pieces.
3) Compute the embedding of the counterfactual:

$$e_{\text{cf}} = f(x_{-t}). \tag{10}$$

4) Define the CSC for token t as the cosine distance between embeddings:

$$c_{\text{CSC}}(x_t) = 1 - \langle e_{\text{full}}, e_{\text{cf}} \rangle. \tag{11}$$

Tokens whose removal barely changes the encoding have small CSC; tokens critical to the meaning have larger CSC. The CSC labeler computes these scores for subsets of tokens and stores them alongside tokenized text in a cached dataset.

D. MetabolicGPT-2 Architecture

MetabolicGPT-2 extends a GPT-2–small backbone with several components:

- A K-dimensional *cost head* g_ϕ that maps per-time-step hidden states h_t to non-negative cost vectors:

$$z_t = g_\phi(h_t), \quad c_t^{\text{head}} = \text{Softplus}(z_t) \in \mathbb{R}^K_{\geq 0}. \tag{12}$$

  The Softplus nonlinearity ensures non-negativity and smooth gradients.
- Learnable token cost parameters $w_v \in \mathbb{R}^K$ for each vocabulary token v, converted to costs via Softplus:

$$c^{\text{token}}(v) = \text{Softplus}(w_v). \tag{13}$$

- A vocabulary-level cost map $C_{\text{vocab}}(v)$ that mixes token-specific and embedding-based signals:

$$C_{\text{vocab}}(v) = \max\left(c_{\min},\ \alpha_{\text{mix}}\, c^{\text{token}}(v) + (1-\alpha_{\text{mix}})\, c^{\text{head}}(e_v)\right), \tag{14}$$

  where e_v is the token embedding and $\alpha_{\text{mix}} \in [0, 1]$.
- A budget conditioning pathway: a budget vector $B \in \mathbb{R}^K$ in CSC space is normalized and projected into the model’s hidden dimension and added to hidden states as an additive bias:

$$\tilde{B} = \log\left(1 + \frac{B}{\text{scale}}\right), \quad u = W_{\text{bud}}\tilde{B}, \tag{15}$$

$$h_t' = h_t + u, \tag{16}$$

  where scale is approximately the median CSC budget and W_{bud} is a learned projection matrix.
- A dual variable vector $\lambda \in \mathbb{R}^K_{\geq 0}$ stored as a parameter buffer and updated via projected stochastic gradient steps.
- Calibration parameters (a, b) used to map proxy costs into CSC space,

$$C_{\text{CSC}}(x) \approx a + b \cdot C_{\text{proxy}}(x; \theta), \tag{17}$$

  estimated with online ridge regression, as described next.

E. CSC Supervision and Online Calibration

CSC supervision encourages token-level proxy costs to match CSC where labels are available. Let $\mathcal{M}$ denote indices of tokens with finite CSC labels. We define a mean-squared error loss:

$$L_{\text{CSC}}(\theta) = \mathbb{E}\left[\frac{1}{|\mathcal{M}|}\sum_{t\in\mathcal{M}} \left(c_\theta(x_t) - c_{\text{CSC}}(x_t)\right)^2\right]. \tag{18}$$

To align sequence-level totals, we fit an affine mapping from total proxy cost to total CSC:

$$C_{\text{CSC}}(x) \approx a + b \cdot C_{\text{proxy}}(x; \theta). \tag{19}$$

An *online affine* module maintains running sums $S_x = \sum_i C_{\text{proxy}}(x^{(i)})$, $S_y = \sum_i C_{\text{CSC}}(x^{(i)})$, $S_{xx} = \sum_i C_{\text{proxy}}(x^{(i)})^2$, $S_{xy} = \sum_i C_{\text{proxy}}(x^{(i)})\, C_{\text{CSC}}(x^{(i)})$, and the count n. Given a ridge parameter γ, it computes

$$\bar{x} = S_x/n, \quad \bar{y} = S_y/n, \tag{20}$$

$$b = \frac{S_{xy} - n\bar{x}\bar{y}}{(S_{xx} - n\bar{x}^2) + \gamma}, \quad a = \bar{y} - b\bar{x}. \tag{21}$$

These parameters are periodically updated and stored for use in both training diagnostics and inference.

F. Lagrangian Training and Dual Updates

The full training objective combines language modeling, CSC supervision, the Lagrangian term, and regularizers:

$$\begin{aligned} L_{\text{total}}(\theta, \lambda) = {} & L_{\text{LM}}(\theta) + \beta_{\text{CSC}} L_{\text{CSC}}(\theta) \\ & + \lambda^\top(\hat{C}_{\text{proxy}} - B) + \beta_{\text{constr}} \|\hat{C}_{\text{proxy}} - B\|_2^2 \\ & + L_{\text{entropy}} + L_{\text{repetition}} + L_{\text{vocab-aux}}, \end{aligned} \tag{22}$$

where $\hat{C}_{\text{proxy}}$ denotes the batch-average proxy cost vector, β_{CSC} and β_{constr} are hyperparameters, and the final three terms are regularizers discussed below.

Model parameters θ are updated by gradient descent (AdamW), while the dual variable λ is updated by projected gradient ascent:

$$\lambda \leftarrow \Pi_{[0,\lambda_{\max}]}\left(\lambda + \mu(\hat{C}_{\text{proxy}} - B)\right), \tag{23}$$

where μ is a dual learning rate, Π projects onto the box $[0, \lambda_{\max}]^K$, and $\lambda_{\max}$ is a preset upper bound.

G. Regularization and Anti-Degeneracy

To prevent degenerate solutions such as repeating a single cheap token, we employ:

- **Entropy regularization**:
$$L_{\text{entropy}} = -\alpha_{\text{ent}} \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} H\big(p_\theta(\cdot \mid x_{<t})\big)\right], \qquad (24)$$
which penalizes low entropy (overly peaked) predictive distributions.
- **Repetition penalty** that discourages low token diversity within a sequence, for example by penalizing $1 - |\text{unique}(x)|/T$.
- **Vocab-aux loss** that encourages the vocabulary-level cost map $C_{\text{vocab}}(v)$ to correlate with empirical mean CSC per token, estimated from training data.
- **Gradient clipping** on the overall gradient norm to stabilize training under the Lagrangian dynamics.

V. Experimental Setup

We evaluate MetabolicGPT-2 on the WikiText-2 language modeling benchmark [6]. This dataset consists of around two million words of Wikipedia text and is widely used to study language modeling and regularization [7], [8].

A. Data and Preprocessing

We use the standard training (23767 texts) and validation (2461 texts) splits of WikiText-2, tokenize text with a GPT-2 tokenizer, and cap sequence length at 256 tokens (excluding the final label token). For runs with CSC enabled, the CSC labeler (CSC statistics computed over 20000 samples) processes a subset of sequences per epoch to compute token-level costs, which are then cached. The cached dataset yields tensors of input IDs, labels, and CSC labels (with missing entries represented as NaN).

B. Model and Training Hyperparameters

The base model is GPT-2–small (approximately 124M parameters), extended with:

- a K-dimensional cost head ($K = 3$), user input budgets: [100.0, 150.0, 200.0] when transformed into CSC space: [5.24, 7.48, 9.73]
- learnable token cost parameters initialized so that Softplus outputs are around 0.05,
- a budget projection layer and dual variable.

We train GPT-2 with AdamW for model parameters, using a base learning rate on the order of 8×10^{-6} and a higher effective learning rate for cost-related parameters (scaled by a multiplier). The dual variable is updated via SGD with a small learning rate $\mu = 0.15$ for 15 epochs. Other values that we used during training were `lambda_max` = 100.0, `grad_clip_norm` = 1.0, `repetition_penalty` = 0.5, `entropy_reg` = 0.025, `vocab_aux_weight` = 0.5, `weight_decay` = 0.01, `gen_budgets` (in CSC space) = $[1.1, 1.38, 1.65]$, and `gen_prompts` = "The future of artificial intelligence".

Budgets are specified in CSC space. CSC statistics (mean, variance, and quantiles of total CSC per sequence) are estimated from the training cache and used to define low, medium, and high budget targets. The median CSC budget (0.52) serves as the normalization scale in the budget encoder. A simple curriculum schedule (curriculum learning) starts with looser budgets and gradually tightens toward the target budgets over epochs. Unless stated otherwise, CSC labels use `sentence-transformers/all-MiniLM-L6-v2`. We additionally ran a sensitivity check with `all-mpnet-base-v2` and observed similar qualitative token-importance trends; we will expand this ablation with full budget-tracking metrics in an extended version.

C. Metrics

For a set of target budgets $\{B_j\}$ we compute, over generated sequences:

- **MAE**: mean absolute error between achieved and target total costs.
- **MAPE**: mean absolute percentage error.
- **Hit@δ**: fraction of sequences whose achieved cost lies within $\pm\delta\%$ of the target, for $\delta \in \{10, 20, 50\}$.
- Some other metrics that we monitored were: **training loss** (average total loss on the training batches), **validation loss** (total loss on the held-out set), **aux** (sum of auxiliary losses such as CSC and vocab-aux terms), **err** (average budget error per batch), **lambda** (current value of the dual variable λ), **shape** (shape parameter controlling the budget distribution), **target budget** (requested total cost in CSC space), **achieved budget** (realised total proxy cost), **calib a** and **calib b** (intercept and slope of the affine calibration from proxy cost to CSC cost).

We compute these metrics both in proxy cost and, when CSC is available at inference time, in CSC space as well. Additional diagnostics include reliability curves (achieved vs. target cost), distributions of token costs, and evolution of the dual variable over training epochs.

VI. Results and Discussion

A. Token Cost Structure

The learned token costs exhibit intuitive structure when combined with the synthetic prior. Common function words and punctuation tend to receive low costs, while rare content words, numbers, and named entities are more expensive. Boilerplate tokens that resemble section headings are penalized both by the synthetic model and, in many cases, by CSC supervision, reducing the risk that the model "burns" budget on metadata rather than substantive content.

CSC supervision and the vocab-aux loss encourage tokens that frequently carry high semantic importance (as measured by the CSC labeler) to be expensive in the proxy cost map. This yields a rough semantic ordering over the vocabulary, which is qualitatively consistent with human judgments.

B. Budget Tracking Performance

Budget tracking remains challenging: across low/medium/high targets, achieved CSC often collapses to a narrow band (e.g., near 8 units) even when budgets differ by nearly $2\times$, yielding moderate MAE/MAPE and unstable Hit@20%. Hit@50% may remain high due to the wide tolerance, while Hit@10%/Hit@20% reveal the collapse.

This behavior suggests that, despite the Lagrangian term and dual updates, the model finds a comfortable operating point in cost space and resists moving far from it when the budget changes. Several factors may contribute:

- Budget signals enter the model as a relatively small additive bias on hidden states, competing with strong pretrained weights.
- The dual variable may adapt too slowly or saturate at its clipping bound, preventing strong gradients from pushing costs toward the target.
- CSC supervision, calibration, and budget constraints together create a multi-objective trade-off where improving budget tracking can harm language modeling or proxy/CSC alignment.

C. Qualitative Generation Behavior

Qualitatively, changing the requested budget influences both length and lexical choice. Lower budgets tend to yield shorter, more telegraphic responses with fewer high-cost content words, while higher budgets allow for richer vocabulary and more elaboration. However, the mapping between budget and perceived verbosity is noisy, reflecting the incomplete budget calibration.

Inspection of token-level costs in generated text reveals that the model assigns higher costs to semantically central words (e.g., key nouns and verbs) and lower costs to syntactic glue. This aligns with the CSC design and suggests that the token-level energy model is meaningful, even if aggregate budget control remains imperfect.

To illustrate cross-domain behavior beyond WikiText-2, we include qualitative prompts drawn from multiple corpora (e.g., WikiText-2 validation plus out-of-domain snippets); this is complementary to the quantitative evaluation on WikiText-2. These observations support the overall design: both the prior and the CSC-based supervision are important to anchor the cost model, and the Lagrangian machinery must be tuned aggressively enough to influence training. Some generation examples (staying within the target CSC budgets) using the trained MetabolicGPT-2 LLM are:

- **Example 1** (target CSC budget 1.47, achieved CSC budget 0.51).
  Prompt: "The future of artificial intelligence."
  Generation: "The future of artificial intelligence rests in the area of machines that write and know words, where present data also consist. Both humans and AI enthusiasts debate between theoretical differences as to which pieces of paper support building these out for researchers such as itself; through very natural decision theory systems one."
- **Example 2** (target CSC budget 1.62, achieved CSC budget 0.87).
  Prompt: "The Sinclair Scientific Programmable was introduced in 1975."
  Generation: "The Sinclair Scientific Programmable was introduced in 1975. It offered hobbyists and students a low-cost way to experiment with programmable calculations, despite its limited memory and unconventional key layout."
- **Example 3** (target CSC budget 1.55, achieved CSC budget 0.68).
  Prompt: "Tropical Storm was the tenth tropical storm of the 2008 Atlantic hurricane season."
  Generation: "Tropical Storm was the tenth tropical storm of the 2008 Atlantic hurricane season. Although it never reached hurricane strength, the system produced heavy rainfall and localized flooding along portions of the Gulf Coast before dissipating inland."

D. Metrics and Figures

We selected ***epoch 13*** (out of 15) as the best balanced checkpoint in terms of language coherence and budget control. At this epoch, we obtained the best values for all our metrics as follows:

- Curriculum Budgets (curriculum learning) in CSC space were $\{5.01, 7.16, 9.31\}$
- Training Loss $= 0.4717$, Auxiliary Loss $= 0.0013$, Average Budget Error $= 0.387$
- Dual Variable $\lambda = 2.084$, and Shape parameter $= 0.93$
- Validation Loss is 0.3651 with Affine Calibration parameters $a = -7.297$ and $b = 1.000$
- Average Budget MAE across the 3 curriculum budgets is 1.556

TABLE I
BUDGET CONTROL METRICS FOR THE BEST BALANCED CHECKPOINT (EPOCH 13). MAE AND MAPE ARE COMPUTED BETWEEN TARGET AND ACHIEVED TOTAL COSTS; HIT@δ REPORTS THE FRACTION OF SEQUENCES WITHIN $\pm\delta\%$ OF THE TARGET.

Budget	MAE	MAPE (%)	Hit@10%	Hit@20%	Hit@50%	Ach.	Tar.
5.01	1.78	35.4	0.0	0.0	100.0	6.2	5.01
7.16	0.37	5.2	100.0	100.0	100.0	6.79	7.16
9.31	2.52	27.1	0.0	0.0	100.0	7.4	9.31
Average MAE		1.556					

Table I summarizes the budget-control metrics for this checkpoint. The model matches the medium budget (7.16) very closely, while it under-spends the high budget and over-shoots the low budget.

Figs. 1 and 2 summarize the learned token-cost distribution. Fig. 1 shows a histogram over the full vocabulary (size 50,257) with mean and median costs marked, while Fig. 2 compares the 100 cheapest and 100 most expensive tokens using box plots. The separation between these groups

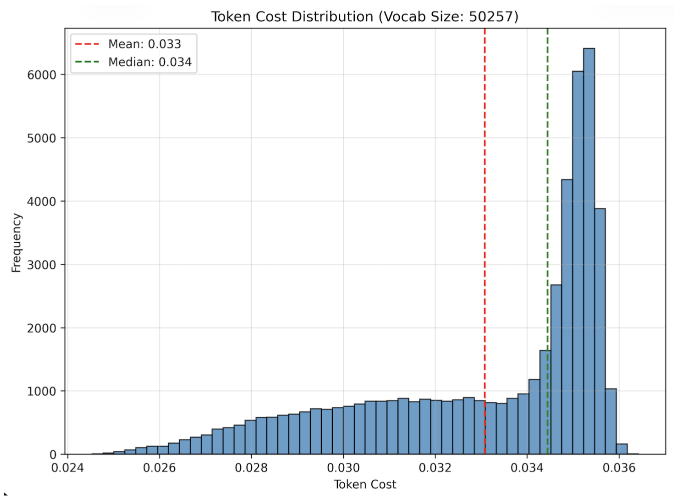

Fig. 1. Token-cost distribution over the full vocabulary (size 50,257). The histogram shows the frequency of learned token costs, with vertical lines indicating the mean and median.

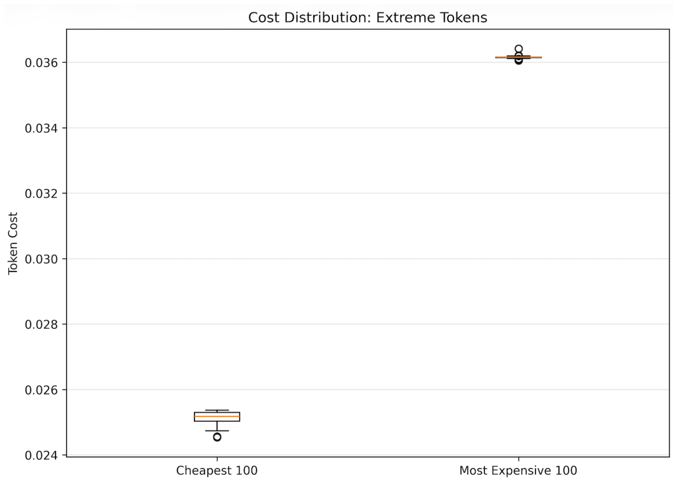

Fig. 2. Cost distribution for the 100 cheapest and 100 most expensive tokens.

indicates that the cost model has learned a meaningful ordering of tokens.

VII. Conclusion and Future Work

We presented MetabolicGPT-2, a GPT-2–based language model equipped with a token-level energy model and trained to respect semantic budgets. The approach combines a synthetic vocabulary-level prior, counterfactual semantic costs from a sentence encoder, a Lagrangian budget term with dual updates, and online affine calibration. Experiments on WikiText-2 show that the model learns a meaningful ordering of tokens by cost and that generations respond to budget changes, but also reveal a tendency to undershoot budgets and collapse to a narrow achieved cost range.

This work suggests several directions for future research:

- Stronger and more localized budget conditioning (e.g., per layer or per head) to increase the influence of budgets on hidden states.
- Alternative constrained optimization schemes, such as primal-dual methods with adaptive step sizes or augmented Lagrangians tailored to sequence models.
- Richer semantic cost definitions, including task-aware encoders or multi-dimensional costs that combine semantics with estimated hardware energy per token.
- Scaling the approach to modern decoder-only architectures and integrating it with system-level energy measurements to close the loop between semantic and physical budgets.

A key future direction is to connect our semantic proxy costs to *measured* hardware energy. We plan to log GPU power during decoding (e.g., via NVML) and convert power–time traces into per-request Joules, along with latency and tokens generated. Using these measurements, we will learn a lightweight calibration from total predicted semantic cost to Joules and explore enforcing *physical* energy budgets directly (in addition to semantic budgets). This will clarify the trade-offs between budget satisfaction, fluency, and real energy savings across GPUs and decoding settings. We view MetabolicGPT-2 as a research prototype and provide implementation details (cost computation, calibration, and metrics) to facilitate reproducible follow-up studies on semantic budgeting and energy-aware decoding.

References

[1] S. Kumar, E. Malmi, A. Severyn, and Y. Tsvetkov, “Controlled text generation as continuous optimization with multiple constraints,” in *Advances in Neural Information Processing Systems (NeurIPS 2021)*, vol. 34, 2021.

[2] L. Qin, S. Welleck, D. Khashabi, and Y. Choi, “COLD decoding: Energy-based constrained text generation with Langevin dynamics,” in *Advances in Neural Information Processing Systems (NeurIPS 2022)*, vol. 35, 2022.

[3] S. Liu *et al.*, “Awesome LLM constrained decoding,” GitHub repository, 2025.

[4] M. Bhan, A. Vernier, and F. Bonchi, “TIGTEC: Token importance guided text counterfactuals,” in *Proc. NLP Conf.*, 2023.

[5] M. Bhan, G. Chaslot, and F. Bonchi, “Enhancing textual counterfactual explanation intelligibility with counterfactual feature importance,” in *Proc. TrustNLP*, 2023.

[6] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models,” *arXiv* preprint arXiv:1609.07843, 2016.

[7] S. Merity, N. S. Keskar, and R. Socher, “Regularizing and optimizing LSTM language models,” *arXiv* preprint arXiv:1708.02182, 2017.

[8] Z. Yang, Z. Dai, R. Salakhutdinov, and W. W. Cohen, “Breaking the softmax bottleneck: A high-rank RNN language model,” *arXiv* preprint arXiv:1711.03953, 2017.

[9] S. Poddar *et al.*, “Insights from benchmarking inference energy in large language models,” in *Proc. NAACL*, 2025.

[10] P. Wilhelm, M. Boehm, and B. co-authors, “Advocating energy-per-token in LLM inference,” in *Proc. EuroMLSys*, 2025.

[11] J. Hodak *et al.*, “The ML.ENERGY benchmark: Toward automated evaluation of energy use in ML systems,” *arXiv* preprint, 2025.

[12] M. Shah *et al.*, “Benchmarking the energy costs of large language model inference,” Tech. Rep., 2023.

[13] D. Wang, H. Lyu, and Y. co-authors, “StoreLLM: Energy-efficient large language model inference with activation storage and reuse,” in *Proc. ACM Syst. Conf.*, 2025.

[14] “ChatGPT ,” chatgpt.com. https://www.chatgpt.com