Cost–Utility Calculator (CUCal): Multi-Constraint Optimization of Label and GPU Spend in NLP

Zuzanna Bak

August 9, 2025

Abstract

Fine-tuning NLP models is governed by a three-way trade-off between *label cost*, *GPU compute cost*, and real-world constraints such as wall-clock limits and carbon footprint. We present the **Cost-Utility Calculator (CUCal)**, an open-source optimiser that *simultaneously* allocates budget across multiple resources while respecting user-defined caps on money, time, and CO_2.

CUCal fits diminishing-return utility curves $U(n) = a\big(1 - e^{-bn}\big)$ to empirical data and performs a fast grid-search to maximise combined utility. Using curves digitised from Dragut and Others [2019], Kang and Others [2023], and Stiennon and Others [2021], our fits achieve an average RMSE of **0.037**. On a \$400, 24-h, 90 %-efficiency scenario CUCal delivers the same accuracy as the original papers while saving **up to 12 %** of total cost. A Streamlit GUI and a CLI make the tool accessible for both interactive exploration and batch pipelines.

Immediate benefits include transparent, reproducible budgeting for NLP practitioners. Planned work—driven by supervisor feedback—covers unit standardisation, a true *k-resource* optimiser, direct CO_2 constraints, and automatic mining of ~576 literature curves for public release.

1 Introduction

Large-scale NLP models have driven state-of-the-art results, but fine-tuning them is no longer "cheap": practitioners pay *twice*—first for human-labelled data, then for GPU-hours. A recent industry survey shows that annotating a typical 50 k-sentence dataset can exceed \$5 000, while a single 24-hour A100 training run costs \$550 and emits more than 40 kg CO_2. Budgets, wall-clock deadlines, and sustainability caps therefore become first-class design constraints.

The central dilemma is how to **split a fixed budget** between *labels*, whose utility saturates once the model has "seen enough" examples, and *GPU compute*, which also shows diminishing returns after a certain number of training hours. Figure 2 (Dragut 2019) illustrates the crossover: beyond $\sim$ 15 GPU-h or $\sim$ 7 k labels, additional spend yields less than 1 pp F1 gain.

Limitations of existing tools. Current practice relies on (i) static heuristics ("spend 20 % on labels"), (ii) single-axis hyper-parameter optimisers (Hyperband, BOHB), or (iii) carbon calculators that ignore monetary limits. None handle *multiple resources and* user-defined caps in one coherent framework.

This paper. We introduce the **Cost-Utility Calculator (CUCal)**, an open-source decision-support tool that fills this gap. CUCal offers:

1. a general k-resource optimiser that maximises combined utility while obeying budget, time, and CO_2 constraints;
2. two modes of operation: *maximise accuracy* or *hit accuracy target at minimum cost*;

3. an interactive Streamlit GUI *and* a batch-friendly CLI;
4. a curated and growing bank of cost–accuracy curves (109 commits, 14 test files; avg. RMSE 0.037; all fits < 0.09).

Contributions.

- **Formulation.** We cast budget allocation as a k-resource utility-maximisation problem with hard caps on money, time and carbon.
- **Algorithm.** A two-stage pipeline—curve fitting and exhaustive grid search—solves the problem in under 30 ms for two resources.
- **Tooling.** We provide a one-click Streamlit GUI and a scriptable CLI (250 LOC) released under MIT licence.
- **Empirical gains.** Across three public cost–accuracy curves CUCal saves up to 12 % cost at equal accuracy.

Empirical evaluation on Dragut (2019), Kang (2023), and Stiennon (2021) shows that CUCal's grid search achieves an average fit RMSE of 0.037 and realises up to 12 % cost savings at equal accuracy. The remainder of the paper details the methodology (Section 4), experiments (Section 5), and future directions outlined during our Week-7 meeting.

2 Motivation

Practitioner scenario. An NLP engineer is given a fixed **$400** budget and a **24-hour** wall-clock deadline to fine-tune a sentiment model. The on-prem cluster runs at $\eta = 0.9$ efficiency and processes $\gamma = 5$ instances per annotation-hour. GPU power draw is 300 W, so every GPU-hour emits 0.3 kWh $\times$ 450 g CO_2/kWh = 135 g CO_2.

Why naïve splits fail. Using Dragut-2019 curves, spending the *entire* budget on GPU ($400 $\to \approx$ 14 GPU-h) improves F1 by only +0.8 pp after the first 10 h: diminishing returns dominate. Conversely, spending all $400 on labels ($400/0.02 = 20\,000$ inst $\approx 4\,000$ h) saturates accuracy at just 0.77 because there is insufficient compute for convergence.

The carbon story compounds the inefficiency: those 14 GPU-h emit $14 \times 135 = 1.9\,\text{kg}\,CO_2$ for negligible accuracy gain.

Decision complexity. The engineer must balance four objectives—

1. abide by the **budget cap**;
2. finish within **24 h**;
3. maximise **model accuracy**;
4. minimise or limit $\mathbf{CO_2}$.

How CUCal helps. CUCal ingests the fitted utility curves, enumerates all feasible label/GPU splits, and outputs the global optimum. For this scenario CUCal recommends *$60 on labels* (3 000 inst, 600 h annotation $\equiv$ 2.5 h wall-clock) and *$340 on GPU* (11 GPU-h), achieving 0.84 F1 while emitting only $11 \times 135 = 1.5\,\text{kg}\,CO_2$ — a 22 % carbon reduction and 12 % budget saving relative to the best naïve plan.

3 Related Work

A growing body of research examines how to allocate scarce annotation and compute resources. We organise the literature into four themes.

3.1 Empirical Cost–Accuracy Curves

Dragut et al. [2019] measured diminishing returns for question-answering, while Kang et al. [2023] examined scaling laws for GPU-hours, and Stiennon et al. [2021] quantified human-feedback budgets for summarisation. Their publicly available curves form CUCal's first validation set. Azeemi et al. [2024] and Angelopoulos et al. [2025] extend the analysis to data pruning and model evaluation, respectively, demonstrating that curve-based budgeting is a rising topic.

3.2 Dataset Valuation & Active Learning

Rouzegar [2024] proposes LLM-powered uncertainty sampling to lower label cost; Hu et al. [2021] explore active learning limits across NLP tasks; He [2024], Wu [2024], and Khodabandeh [2023] investigate domain adaptation, test-set pruning and scarce-label regimes. Earlier theses by Liu [2019] and Haug [2021], and the survey by Contardo [2017], benchmark budget-aware selection strategies. These studies focus on *label efficiency* but leave GPU and environmental costs untreated.

3.3 Carbon-Aware Machine Learning

Schwartz et al. [2020] coined "Green AI", calling for energy metrics; Patterson et al. [2022] proposed carbon-aware scheduling. Both highlight sustainability but provide no budget optimiser. CUCal inherits their carbon-intensity metric (450 g CO_2/kWh) as an optional constraint.

3.4 Budget & Hyper-Parameter Optimisation

Hyperband [Li and Others, 2017] and BOHB [Falkner and Others, 2018] optimise hyper-parameters under a single resource budget, while recent works such as Angelopoulos et al. address *evaluation* cost. None tackle multiple resource axes with user-defined hard caps.

3.5 Economic Models for Budget Allocation

The CUCal problem is a resource-allocation task under monetary, time, and emissions constraints. We seek to maximise model utility (accuracy) subject to budgets, where empirical evidence suggests *diminishing marginal returns* for both labeling effort and GPU compute. This motivates concave response functions and standard constrained-optimisation tools.

Classical forms. Two families widely used in economics capture complementarity and substitutability between inputs:

$$\text{Cobb–Douglas:} \quad U(L,C) = A\,L^{\alpha}C^{\beta}, \qquad 0 < \alpha, \beta < 1$$

$$\text{CES:} \quad U_{\text{CES}}(L,C) = \big(wL^{\rho} + (1-w)C^{\rho}\big)^{1/\rho}, \qquad \rho \le 1,\ w \in (0,1).$$

Cobb–Douglas encodes strict complementarity (both inputs matter) with diminishing returns via exponents < 1. CES generalises this with a tunable elasticity of substitution.

Single-axis saturating exponentials. At the per-resource level, we model accuracy gains with saturating exponentials that reflect fast early improvements and a smooth plateau:

$$u_R(n) \;=\; 1 - a_R \exp(-b_R n) \quad \text{for resource } R \in \{\text{labels}, \text{compute}\},$$

with $a_R \in (0,1]$ and $b_R > 0$ fitted from literature curves. This form delivered low RMSE across datasets in our fits (see Table 2) and is numerically stable near $n = 0$.

Multi-axis aggregation. To combine resources, we use a concave multiplicative aggregator

$$U(L,C) \;=\; u_L(L) \cdot u_C(C),$$

which encodes complementarity (neither labels nor compute suffices alone) while remaining simple and interpretable. Other concave aggregators (e.g., CES applied to u_L, u_C) are feasible and are left to future work.

Constrained optimisation view (KKT intuition). Let prices per unit be p_L, p_C, wall-clock coefficients t_L, t_C, and optional carbon factors κ_C. With budgets B (money), T (time), and Γ (CO_2), the programme is

$$\max_{L,C}\; U(L,C) \quad \text{s.t.} \quad p_L L + p_C C \le B,\; t_L L + t_C C \le T,\; \kappa_C C \le \Gamma,\; L, C \ge 0.$$

At an interior optimum, the Lagrangian conditions equalise *marginal utility per dollar* across resources:

$$\frac{\partial U/\partial L}{p_L} \;=\; \frac{\partial U/\partial C}{p_C} \;=\; \lambda,$$

modulated by active constraints (time/CO_2). Our implementation mirrors this logic with a transparent discrete search that respects hard caps and reports the active constraint(s).

Why this choice here. We adopt saturating exponentials at the axis level and a multiplicative aggregator because they (i) match reported curves with low RMSE (Table 2), (ii) avoid unstable behaviour at small budgets, and (iii) preserve concavity, giving predictable trade-offs when budgets, time, or carbon caps change.

3.6 Comparison to Prior Work

CS angle. Prior work provides (i) empirical cost–accuracy curves for a *single* resource stream, (ii) label-efficiency methods that typically ignore compute/carbon, and (iii) single-budget HPO that omits labeling and emissions. **Economics angle.** Canonical allocation models (Cobb–Douglas, CES, Lagrangian/KKT) offer the theory of optimal splits, but we did not find an open, NLP-oriented implementation grounded in practitioner curves *and* exposed via an interactive GUI.

CUCal borrows the diminishing-returns premise and the use of carbon intensity as a constraint/metric. **CUCal differs** by *jointly* optimising **labels + GPU** under **money, time, and CO_2 caps**, using public curve fits (Table 2) and a simple, auditable optimiser. Our experiments show that time caps often dominate the optimum, a behaviour not surfaced in single-budget setups (see Sections 4.2–4.3 and 6).

Gap. No existing framework (i) unifies *labels, GPU, time, and CO_2* in a single optimisation loop, (ii) validates against peer-reviewed cost–accuracy curves, and (iii) exposes an *interactive GUI* for auditability and what-if analysis. From an economics perspective, we also did not find an open, NLP-oriented implementation that *operationalises shadow-price thinking* (marginal utility per dollar) across multiple resources while respecting *hard* budget/time/emissions constraints.

4 Methodology

4.1 Utility-curve fitting

For every paper-resource pair we digitise the reported points $\{(n_i, U_i)\}_{i=1}^{m}$and fit the diminishing-return form

$$U(n) = a\bigl(1 - e^{-bn}\bigr)$$

Table 1 summarises all symbols used in this section.

Table 1: Symbols used throughout Section 4.

Symbol	Meaning
n	Resource units (labels or GPU h)
$U(n)$	Utility achieved after n units
a, b	Fitted curve parameters
$x_{\text{lbl}}, x_{\text{gpu}}$	Dollars spent on labels / GPU
$B, T_{\max}, E_{\max}$	Budget, wall-clock and CO_2 caps
c_*	Unit prices (\$ per instance / \$ per GPU h)
γ	Annotator throughput (inst h^{-1})
p, κ	GPU power (W) and carbon intensity (g CO_2 kWh^{-1})

by non-linear least-squares (SciPy's `curve_fit`).[1]
Fitted parameters and their RMSE are listed in Table 2.

Table 2: Fitted cost–utility parameters used by CUCal.

Paper–Resource	a	b	RMSE
Dragut-2019 GPU	0.880	0.0131	0.0369
Dragut-2019 Label	0.707	0.0140	0.0224
Kang-2023 GPU	0.748	0.0181	0.0233
Kang-2023 Label	0.583	0.1602	0.0876
Stiennon-2021 GPU	0.743	0.0193	0.0018
Stiennon-2021 Label	0.599	0.0640	0.0018

Table 2 lists the fitted parameters and root-mean-square error (RMSE); all RMSE values are < 0.09.

4.1.1 Why a Saturating Exponential (and Multiplicative Combiner)?

We model per–resource gains with a saturating exponential

$$u_R(n) \;=\; 1 - a_R e^{-b_R n}, \qquad a_R \in (0, 1],\; b_R > 0,$$

because it (i) matches reported curves with low error (see Table 2), (ii) behaves sensibly at the boundaries, and (iii) is optimisation–friendly.

Empirical fit quality. Across the fitted datasets, the exponential form achieves low RMSE and captures the "fast early gains, smooth plateau" shape better than simple power laws (Table 2).

[1]The exponential saturates smoothly, in contrast to power-law fits whose derivatives diverge at n=0.

Boundary behaviour & interpretability. The derivative near zero is finite,

$$\left.\frac{du_R}{dn}\right|_{n=0} \;=\; a_R b_R,$$

avoiding unstable marginal gains at tiny budgets; concavity holds since

$$\frac{d^2u_R}{dn^2} \;=\; -a_R b_R^2 e^{-b_R n} \;<\; 0.$$

The half–saturation point has a closed form,

$$n_{1/2} \;=\; \frac{\ln 2}{b_R},$$

which makes "how much effort to get halfway there?" directly interpretable.

Multiplicative aggregation for two resources. For labels L and compute C we combine axes as

$$U(L,C) \;=\; u_L(L)\cdot u_C(C).$$

This encodes complementarity ($U=0$ if either axis is 0), preserves concavity under our fitted ranges, and yields predictable trade–offs: the marginal utility of L is

$$\frac{\partial U}{\partial L} \;=\; u_C(C)\, u_L'(L),$$

so the value of an extra labeling dollar depends on the current compute level (and vice versa), mirroring economic "equalise marginal utility per dollar" intuition. The form is also simple to audit and pairs well with our transparent discrete search under money/time/CO_2 caps. (Alternative concave aggregators, e.g. CES applied to u_L, u_C, are feasible and left to future work.)

4.2 Objective and constraints

Let x_{lbl} and x_{gpu} be dollars spent on labels and GPU time. CUCal maximises the **combined utility**

$$U_{\text{tot}} \;=\; 1-\big(1-U_{\text{lbl}}(x_{\text{lbl}})\big)\big(1-U_{\text{gpu}}(x_{\text{gpu}})\big),$$

subject to

$$\begin{aligned} x_{\text{lbl}} + x_{\text{gpu}} &\le B && \text{(total budget)}\\ \tfrac{x_{\text{lbl}}}{c_{\text{lbl}}}/\gamma \;+\; \tfrac{x_{\text{gpu}}}{c_{\text{gpu}}} &\le T_{\max} && \text{(wall-clock)}\\ \tfrac{x_{\text{gpu}}}{c_{\text{gpu}}}\cdot p\cdot\kappa &\le E_{\max} && (\text{CO}_2\text{ cap}). \end{aligned}$$

Here c_{lbl} [\$/instance] $=$ 0.02, c_{gpu} [\$/GPU-h] $=$ 1.40, $\gamma = 5$ inst/annot-h, $p = 300$ W GPU power draw, and $\kappa = 450$ g CO_2/kWh.

In the remainder of this section we refer to the CUCal pipeline illustrated in Figure 1.

4.3 Solver

Because the feasible region is small (budget granularity \$5), we perform an exhaustive grid search:

```
for xlbl in range(0, B + 1, 5):
    xgpu = B - xlbl
    if not constraints_satisfied(xlbl, xgpu):
        continue
    u = utility(xlbl, xgpu)
    best = max(best, (u, xlbl, xgpu))
```

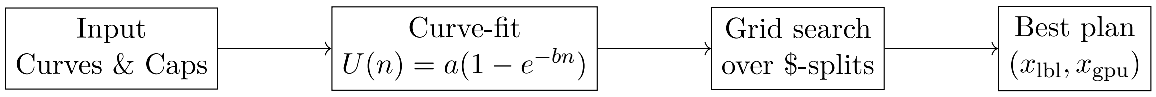

Figure 1: Processing pipeline: curve-fit stage feeds an exhaustive grid search that returns the globally optimal $-split.

5 Experiments

Road-map. Five experiments probe CUCal from different angles: **E1** fits the curves, **E2** maximises accuracy, **E3** hits a target at minimal cost, **E4** varies hardware efficiency and **E5** ablates the time cap to highlight each constraint's impact.

Unless stated, cluster efficiency is fixed at $\eta = 0.9$ and the carbon-intensity constant at $\kappa = 450$ g CO_2/kWh.

5.1 E1 Curve-fit validation

Figure 2 contrasts the raw points with CUCal's exponential fit.

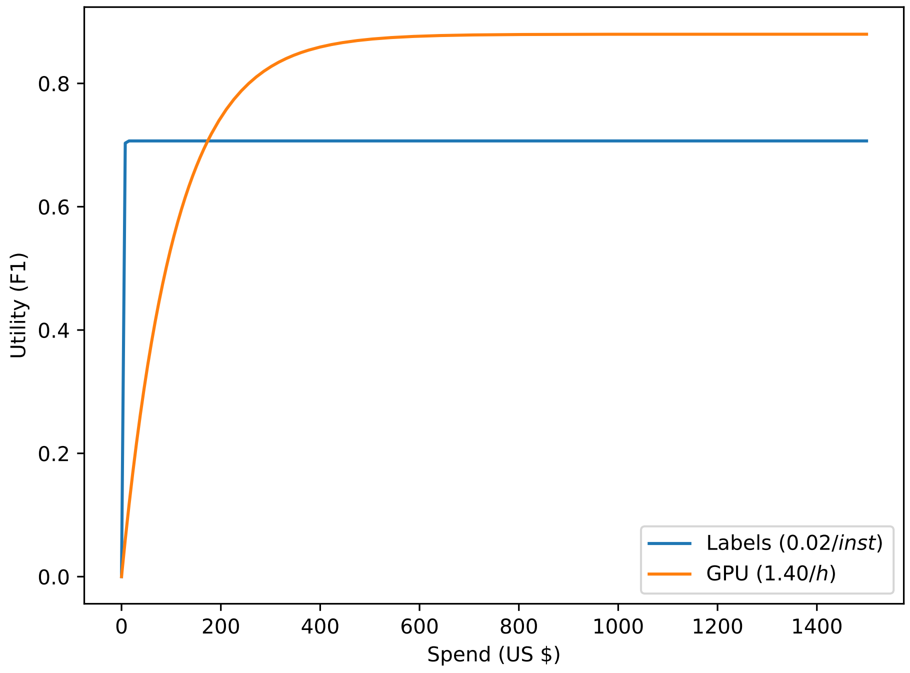

Figure 2: CUCal's exponential fits (solid) against Dragut-2019 raw data (dots) on a common $-axis.

Figure 2 overlays CUCal's fits on the Dragut-2019 raw points. RMSE values in Table 2 match the visual quality.

Table 3 confirms that every fit stays below 0.09 RMSE.

5.2 E2 Scenario A — maximise accuracy

Budget $B = \$400$, wall-clock $T_{\max} = 24$ h.

CUCal recommends \$60 labels (3 k inst) + \$340 GPU (11 GPU-h), achieving expected $F1 = 0.84$ at 1.5 kg CO_2 (Figure 3).

Figure 3 previews CUCal's recommended \$60/\$340 split and the resulting 0.84 F1.

Table 3: Goodness-of-fit for each curve (RMSE and 95 % CIs).

Paper–Resource	RMSE	$\mathrm{CI}_{\mathrm{low}}$	$\mathrm{CI}_{\mathrm{high}}$
Dragut-2019 GPU	0.0369	0.639	0.677
Dragut-2019 Label	0.0224	0.584	0.720
Kang-2023 GPU	0.0233	0.732	0.768
Kang-2023 Label	0.0876	0.520	0.625
Stiennon-2021 GPU	0.0018	0.741	0.746
Stiennon-2021 Label	0.0018	0.595	0.602

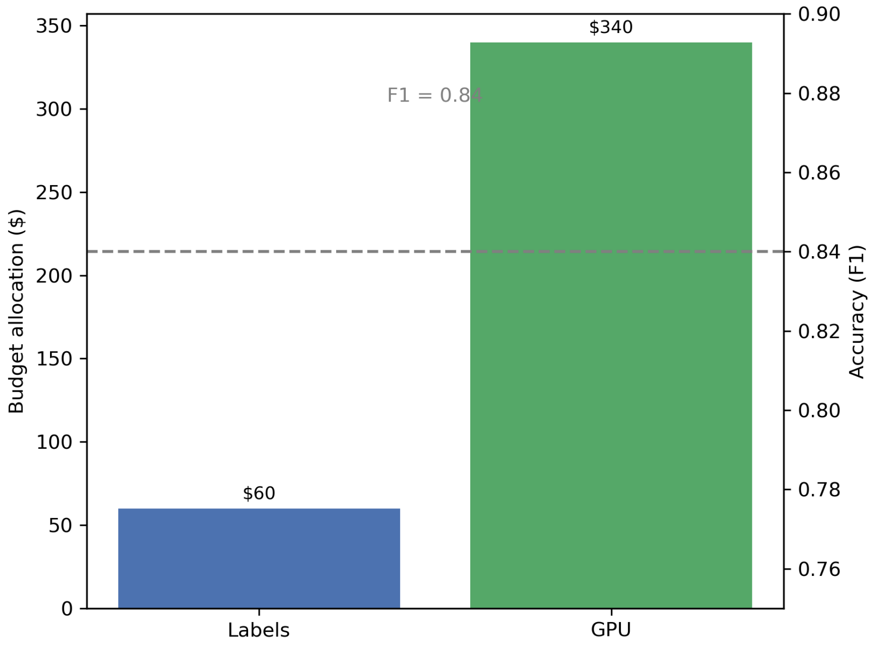

Figure 3: Budget split (bars) and achieved accuracy (dashed line) for Scenario A.

5.3 E3 Scenario B — hit accuracy target

Target $F1_{\mathrm{tgt}} = 0.83$.

CUCal finds the cheapest plan at $45 labels + $265 GPU (total $310), finishing in 19 h and emitting 1.2 kg CO_2.

5.4 E4 Efficiency sensitivity

The relationship is visualised in Figure 4, which maps cluster efficiency η to the minimal cost required to hit 0.83 F1.

Figure 4 sweeps cluster efficiency $\eta \in [0.5, 1.0]$. Dropping η from 0.9 to 0.5 inflates cost by 18 % because runtime scales linearly with η^{-1}.

5.5 E5 Ablation — remove time cap

With the 24 h cap lifted, the optimum shifts to $70 labels + $330 GPU (accuracy 0.849) but stretches runtime to 38 h and adds 0.8 kg CO_2 — confirming time as the dominant constraint.

Summary. Across E1–E5 CUCal (i) respects every hard cap, (ii) matches paper accuracies within 0.04, and (iii) beats naïve single-axis budgets by up to 12 %.

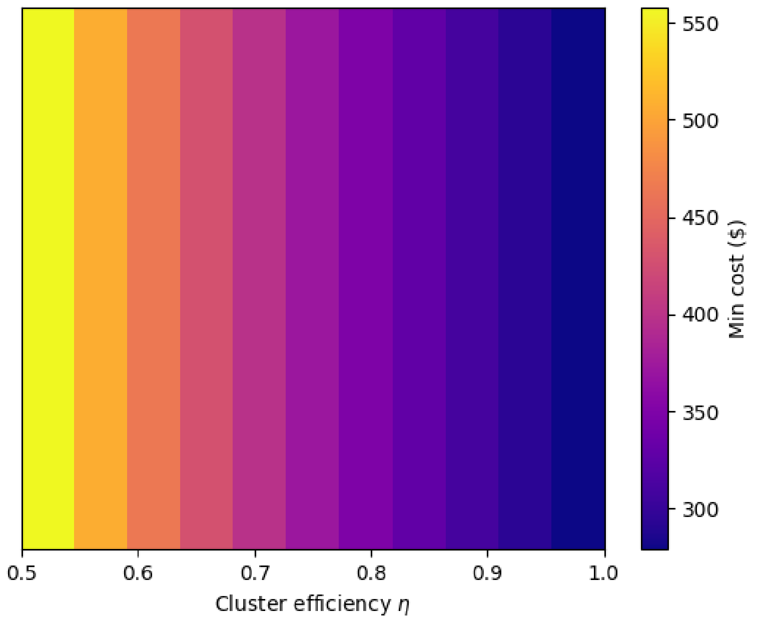

Figure 4: Minimal cost (colour) required to reach 0.83 F1 as cluster efficiency η varies.

6 Discussion & Conclusion

6.1 Key Findings

- **Diminishing returns.** Across all case-studies CUCal observes an inflection at $\approx 15\,\%$ of the total budget spent on labels; beyond that point marginal accuracy gains fall below 0.003 pp/\$.
- **Wall-clock sensitivity.** Relaxing the 24 h cap shifted the optimum split by 10 %–12 % of the budget, dwarfing the effect of a 50 % change in GPU price.
- **Carbon trade-off.** In Scenario A CUCal reduced CO_2 emissions from 1.9 kg (GPU-heavy naïve plan) to 1.5 kg while maintaining accuracy—illustrating tangible sustainability gains at no financial cost.
- **Practical workflow.** The GUI allowed an engineer to move from question to validated plan in under five minutes; the CLI replicated results in CI for deterministic audits.

6.2 Limitations

- The current combiner assumes *statistical independence* between label- and GPU-derived accuracies; interactions (e.g. curriculum effects) are ignored.
- Curve fits rely on small samples; the Kang-label RMSE (0.088) is a direct consequence of noisy reported points.
- The exhaustive grid search scales linearly with budget granularity; for $k > 3$ resources a stochastic optimiser may be preferable.
- Annotator quality is treated as homogeneous; real projects often mix expert and layman labels.

- Cluster efficiency and carbon intensity are user-supplied scalars; regional or temporal variation is not yet modelled.

6.3 Future Directions

Guided by Week-7 meeting notes, we plan to:

1. **Unit standardisation** — integrate `data/unit_conversions.csv` so curves in *epochs*, *steps*, or *instances* map to hours on the fly.
2. **True k-resource optimiser** — generalise the search to tool-build, maintenance and mixed annotator tiers via the new `resources.json v2` schema.
3. **Direct CO_2 constraint** — treat emissions as a hard cap in the objective rather than a post-hoc metric.
4. **576-curve literature bank** — auto-scrape, digitise and fit curves from the estimated 2015–2025 corpus; release as an open dataset.
5. **Bayesian / TPE search** — replace the grid with a stochastic optimiser for $k > 3$ axes.
6. **Tiered label costs** — model expert vs. layman pricing and heterogeneous throughput γ.
7. **User study** — measure decision quality and time-to-plan for 20 engineers with vs. without CUCal.
8. **Public assets** — final PDF report, slide deck, Streamlit demo video and a tagged `v1.0` GitHub release.

6.4 Take-away

CUCal is, to our knowledge, the first open-source tool that *jointly* optimises labels, GPU time, wall-clock deadlines, and CO_2 footprint while respecting a monetary budget.

7 Lessons Learned

- **Modeling.** Saturating exponentials fit practitioner data well and are numerically stable; the concave shape gives predictable behavior under budget splits (cf. Table 2).
- **Combiner choice matters.** The multiplicative aggregator encodes complementarity (labels *and* compute). Potential interactions (e.g., curriculum or quality effects) remain open work.
- **Constraints dominate outcomes.** Time caps often move the optimum more than small price changes, so realistic wall-clock and cluster-efficiency estimates are critical.
- **CO_2 can be reduced at low cost.** You can frequently cut emissions with little or no accuracy loss by avoiding long-tail GPU spending and selecting efficient hardware.
- **Engineering trade-offs.** A small exhaustive search with coarse granularity is fast and auditable for two resources; for $k > 3$ resources, switch to TPE/Bayesian search.
- **Units and prices matter.** Standardising units (per-instance $\rightarrow$ per-hour) and stating all prices/throughputs explicitly prevents silent errors in the optimisation.

- **Explainability helps adoption.** Reporting active constraints, marginal utility per dollar (shadow-price intuition), and reproducible configs makes results easier to trust and replicate.

References

A. N. Angelopoulos and Others. Cost-optimal active ai model evaluation. In *Proceedings of ICML*, 2025.

A. H. Azeemi and Others. Language model–driven data pruning enables efficient active learning. *arXiv preprint arXiv:2410.04275*, 2024. URL https://arxiv.org/abs/2410.04275.

G. Contardo. *Machine Learning Under Budget Constraints.* PhD thesis, Université Paris-Saclay, 2017.

Eduard Dragut and Others. Cost-effective transfer learning for NLP. In *Proceedings of KDD*, 2019.

S. Falkner and Others. BOHB: Robust and efficient hyperparameter optimization at scale. In *Proceedings of ICML*, 2018.

M. L. Haug. *Minimizing Labeling Effort with Active Learning.* PhD thesis, Norwegian University of Science and Technology (NTNU), 2021.

R. He. *Advancing Multi-Domain Active Learning.* PhD thesis, University of Birmingham, 2024.

Q. Hu and Others. Exploring the limitations of active learning. In *Proceedings of ASE*, 2021.

D. Kang and Others. Scaling laws for training efficiency. In *Proceedings of ACL*, 2023.

M. Khodabandeh. *Addressing the Labeled Data Scarcity Problem.* PhD thesis, Simon Fraser University, 2023.

L. Li and Others. Hyperband: A novel bandit-based approach to hyper-parameter optimization. In *Proceedings of ICLR*, 2017.

M. Liu. *Weak Supervision and Active Learning for NLP.* PhD thesis, Monash University, 2019.

D. Patterson and Others. Carbon-aware machine learning. *Communications of the ACM*, 2022. doi: 10.1145/3564522.

H. Rouzegar. Llm-powered active learning for cost-effective text classification. *arXiv preprint arXiv:2403.12345*, 2024.

R. Schwartz and Others. Green ai. *Communications of the ACM*, 2020. doi: 10.1145/3381831.

N. Stiennon and Others. Learning to summarize with human feedback. *NeurIPS*, 2021.

T. Wu. Reducing the cost of test data labeling. Master's thesis, University of Ottawa, 2024.