Reshaping Bonsai: Pruning LLMs Using Comprehensive Metrics

Vashisth Tiwari
vashistt@andrew.cmu.edu

Amanda Li
xal@andrew.cmu.edu

Emily Guo
epguo@andrew.cmu.edu

Abstract

This project builds upon Bonsai (Dery et al., 2024), a forward-only regression-based neural network pruning method. We focus on improving model performance with respect to a specific downstream task of improving mathematical and logical reasoning (with a special focus on the GSM8K (Cobbe et al., 2021) dataset) and introduce a novel metric designed to capture more information about the model's performance. This metric combines lexicographical similarity, semantic similarity, and accuracy as comprehensive components with which to evaluate the answers generated by our model against the ground truth. From our experimentation, we found that models pruned with respect to this novel metric outperformed the baseline model from our previous report on logical benchmarks with 0.5 target sparsity. We also note that pruning with respect to GSM-8k in the CoT prompt fashion as described in the paper leads to better performance than the baseline.

Code can be found at this repository: `https://github.com/Vashistht/anlp-project`.

1 Introduction

Large Language Models (LLMs) have shown remarkable abilities across multiple complex tasks. However, these models with billions of parameters require significant and prohibitively large computational resources. In the light of this problem, pruning has emerged as a popular method for compressing neural networks alongside others like quantization. Pruning refers to the process of shrinking the size of the model by removing specific weights while striving to preserve performance of the original network, often with respect to a downstream task. (Sun et al., 2023)

Recent literature has shown that sufficiently large language models can exhibit reasoning abilities, the exact extent of which (and the relationship between these reasoning abilities and model size) is unclear (Huang and Chang, 2022). While model pruning has been extensively researched, we find that very few include model performance on reasoning tasks as part of their tabulations, and the ones that do typically incorporate knowledge graphs as a significant part of their model architecture (Ren and Zhu (2022), Su et al. (2024), Sun et al. (2021)). In this work we focus on pruning with a specific focus on improving mathematical reasoning of pruned models. In the following sections we motivate why this problem is important and related previous work.

2 Background

2.1 Pruning Overview

The two categories of pruning are structured and unstructured. Unstructured pruning removes specific individual parameters from the model. This results in more sparse weight metrics with a reduced memory footprint. However, a key downfall is that any real inference speedups are not possible without specialized hardware. Semi-structured sparsity (2:4, 4:8) that can make use of specialized hardware is known to degrade the performance even further than unconstrained unstructured pruning and are less effective than structured pruning methods (Sun et al., 2023; Zhou et al., 2021).

Structured pruning, on the other hand, imposes structured sparse patterns by taking a more modular view of the model; here we remove entire units / sub-modules of the model. These methods lead to a significant speedup across all hardware but also lead to more considerable performance loss.

2.2 Limitations of Gradient-Based Pruning Methods

Most prior structured pruning approaches have focused on gradient-based pruning methods. While these methods are memory and compute-efficient during inference time, they are memory-bound at pruning. Previous works have shown that the forward pass of a popular optimizer like Adam, AdamW needs to store $\sim 3x$ of the original model weights as it stores the first and second order estimates (Loshchilov and Hutter, 2019).

Thus, even for a half-precision (`float16`) LlaMA-2-7B model with around 14 GB of trainable weights, a method with backward passes needs an additional 28 GB memory bandwidth, making it extremely prohibitive. See Table 1 for a more detailed summary.

Regime	Resource	Quantization (Mixed Precision)	Distillation	Unstructured Pruning	Gradient-Based Structured Pruning	Bonsai
Train	Memory	✓	✓	✓	×	✓
	Compute	✓	×	✓	✓	✓
Inference	Memory	✓	✓	✓	✓	✓
	Compute	×	✓	×	✓	✓

Table 1: Landscape of resource consumption (memory and compute) of different model compression methods at training time and the inference time resource consumption of the models they deliver. × means the method incurs a prohibitive cost to the lay practitioner whilst ✓ denotes that it is a viable option with respect to that resource ((Dery et al., 2024))

2.3 Improving LLM Reasoning

The seminal paper on Chain of Thought (COT) (Wei et al., 2023) showed how generating a *chain of thought* (intermediate reasoning steps) significantly improves the complex reasoning abilities of large language models. However, the chain of thought is fundamentally tied to the model size and its ability to generate intermediate reasoning steps. COT has only been shown to be beneficial for sufficiently big models. However, based on this idea, multiple works have focused on incorporating a chain of thought in model compression.

Notably, (Shridhar et al., 2023; Li et al., 2023) have explored employing a chain of thought generated from a larger teacher model to generate the rationale and then training the smaller model to predict the rationale. However, as discussed in (Xia et al., 2022), distillation is prohibitively expensive. It requires large amounts of unlabeled data and is expensive to train.

On the other hand, fine-tuning based on a chain of thought has also been employed and has been shown to increase reasoning abilities. (Kim et al., 2023) look at instruct tuning small language models with a chain of thought rationales generated by the larger model. In their work, they report an increase of 4.34% on the Big Bench Hard benchmark. Building on these works, in the paper, we introduce a method for reasoning incorporated pruning. **Therefore, in the work, we focus on finding ways to satisfy the three following desired qualities:**

1. **Pruning Method that uses Forward-pass only**
2. **Structured Pruning Method**
3. **Improved Mathematical Reasoning Performance (via incorporating CoT-based metric into pruning itself)**

In the light of the following limitations, we draw attention to the recent work by (Dery et al., 2024), **Bonsai**, which is a forward pass-only, structured pruning approach under memory limitations that is more representative of an average practitioner. In the next sections, we introduce Bonsai (in section 3 and our work that builds off Bonsai in section 4, given our focus on reasoning tasks.

3 Bonsai: Method and Limitations

Bonsai is a forward-pass-only, structured pruning method that decides which modules to prune from the LLM based on estimates of module importance. The estimate of module importance is done perturbatively by generating submodules and evaluating their performance over a small number of samples.

As described in section 2.1, for a large language model (LLM), $\mathbf{M}_\theta$, parameterized by $\theta \in \mathbb{R}^D$, and a utility function U that measures the model's performance on a target task, the objective is to prune $\mathbf{M}_\theta$ to create a smaller and more effective model.

Consider the model M_θ comprising of non overlapping modules $\mathbf{m} = m_i i \in [N]$ with corresponding parameter counts $\mathbf{s} = s_i i \in [N]$ such that $\sum_i s_i = D$. For structured pruning, we want to compress $\mathbf{M}_\theta$ by identifying accurate sub-models defined by subsets of $\mathbf{m}$.

For a sparsity target p, structured pruning can be formulated as the following combinatorial optimization problem:

$$\mathbf{m}^* = \text{argmax}_{\bar{\mathbf{m}} \in \mathcal{F}_p} \quad U\left(\mathbf{M}_{|\bar{\mathbf{m}}}\right) \qquad \text{where}$$
$$\mathcal{F}_p = \left\{ \bar{\mathbf{m}} \subseteq \mathbf{m} \;\middle|\; \left(\sum_{[j: m_j \in \bar{\mathbf{m}}]} s_j \right) \leq (1-p)D \right\} \tag{1}$$

$\mathcal{F}p$ represents all sub-models that satisfy the sparsity threshold. The optimal sub-model $\mathbf{M}|\mathbf{m}^*$ has a smaller memory footprint and faster inference due to fewer modules compared to $\mathbf{M}$.

In the setting we are working with (forward passes only), we wish to estimate the solution to 1, without any gradients. To do so, Bonsai performs a small number of n evaluations to gather the relevance of each module in $\mathbf{M}$ with respect to some metric U, upon which we can choose to retain the modules with higher relevance scores and discard the ones with lower relevance score. This gives us the dataset of sampled sub-modules and their corresponding evaluated performances $\mathbb{D} = \{\bar{m}_k, U_k\}$. Thus, we treat the score estimation β as an under-specified regression problem given by the equation 2 where $\alpha_{\bar{\mathbf{m}}_k}$ is a binary mask which is 0 where we are dropping the modules and 1 otherwise.

$$\hat{\beta} = \arg\min_{\beta \in \mathbb{R}^N} \left(\frac{1}{n} \sum_{(\bar{\mathbf{m}}_k, U_k) \in \mathbb{D}} (U_k - \beta^T \alpha_{\bar{\mathbf{m}}_k})^2 \right) + \gamma \|\beta\|_2 \tag{2}$$

Note that they combine different modules using a mask, since creating every sub-model would be too intensive. The interesting part is how they select sub-models for evaluation. Each sub-model has some probability of being included, but instead of uniformly sampling, methods from previous unstructured pruning metrics, for example Wanda (Sun et al., 2023), is used instead. Overall, this method prunes a certain fraction of the model $[p_i ter]$ upon until the desired sparsity $[p]$ is reached. The number of submodels per iteration $[n_{\mathtt{iter}}]$ is related to p such that $n_{\mathtt{iter}} = \lceil \frac{n}{\mathtt{iter}} \rceil$ and $\mathtt{iter} = \frac{p}{p_{\mathtt{iter}}}$. Putting together we have the Bonsai Algorithm as described in the Figure 1.

```
Algorithm 1 Bonsai
 1: Input:
    Model [$\mathbf{M}_\theta$], sub-models per iteration [$n_{\text{iter}}$]
    Sparsity per iteration [$p_{\text{iter}}$], Target sparsity [$p$]
    Module list [$\mathbf{m}$]
 2: for $l = 1$ to $\lceil \frac{p}{p_{\text{iter}}} \rceil$ do
 3:     $\rho^l \leftarrow$ Calculate unstructured pruning metric for all
        modules in $\mathbf{m}$
 4:     $\bar{\rho}^l \leftarrow$ Fix the top $(1 - 2p_{\text{iter}})$ of $\rho^l$ to $\infty$
 5:     Sample $\{\bar{\mathbf{m}}_i\}_{[\frac{n_{\text{iter}}}{2}]}$ sub-models according to $\bar{\rho}^l$
 6:     Generate $\{\bar{\mathbf{m}}_i^c\}_{[\frac{n_{\text{iter}}}{2}]}$ complements of $\{\bar{\mathbf{m}}_i\}_{[\frac{n_{\text{iter}}}{2}]}$
 7:     Run forward pass on each sub-model and compute $U$.
        Construct $\mathbb{D}^l = \{\bar{\mathbf{m}}_i, U_i\}_{[\frac{n_{\text{iter}}}{2}]} \bigcup \{\bar{\mathbf{m}}_k^c, U_k\}_{[\frac{n_{\text{iter}}}{2}]}$
 8:     $\beta^l \leftarrow$ Regress $(\mathbb{D}^l)$
 9:     $\{m_{\text{pruned}}\} \leftarrow$ sort $\beta^l$ and remove the bottom $k$ modules that correspond to $p_{\text{iter}}$ fraction of the model.
10:     $\mathbf{m} \leftarrow$ update module list to exclude $\{m_{\text{pruned}}\}$
11: end for
12: Output: Pruned model $\mathbf{M}_{|\mathbf{m}}$
```

Figure 1: Bonsai Algorithm from Dery et al. (2024)

3.1 Limitations in Recreating & Deviations from the Original Results

Dery et al. (2024) prune Llama-2-7B (Touvron et al., 2023a) to 50% sparsity with the Bonsai method described in Figure 1. The best model from the paper uses the following hyper-parameters; this is the model used as a comparison with respect to other models in the 50% sparsity of Llama $\sim 3B$ (Touvron et al., 2023b) parameter range. The results of the article are reported in **Table 3** of the Bonsai paper.

```
# Best Bonsai Hyper-parameters
sparsity_ratio=0.5, masks_per_iter=200,
    nsamples=32 prune_frac=0.05, bsz=1
```

Here `sparsity_ratio` refers to the final sparsity with respect to the original model $[p]$. Llama in both our run and the original run was pruned to 50%. `masks_per_iter` refers to the number of perturbative evaluations required to obtain good estimates of module importance after performing regression. The number of these masks, the finer the pruning, but the more time it takes. These performances are evaluated across `nsamples` for each submodule.

However, for these settings, the authors report that pruning takes $\sim$ **40 hours on the Nvidia A6000** which has a 48GB memory. Trying to recreate this, we faced considerable computational and time limitations. We were unable to access this exact machine and recreated benchmarks on various Amazon Web Service Elastic Compute Cloud machines in the Accelerated Computing G Family (`g4dn.2xlarge, g5.2xlarge`) that met the necessary memory requirements to load the full weights of Llama-2 7B (Touvron et al., 2023b). Interestingly, Bonsai was aiming to alleviate memory-related issues because they avoided computing gradients and developed a memory-friendly structured pruning method. *The underlying irony is left as an exercise for the reader.*

Given the time and computing constraints, we had to edit the hyperparameters of the main model in the paper. We reproduce a version of the pruned Llama-2 7B model (Touvron et al., 2023a) with respect to the Wikitext-2 (Merity et al., 2016a) dataset with the following hyperparameters.

```
# Our LLama-7B Hyper-parameters
sparsity_ratio=0.5, masks_per_iter=100,
    nsamples=8 prune_frac=0.2, bsz=1
```

In our sample tests and according to the paper, the biggest contributor to latency was the lower prune fraction in the original paper. As can be seen in Figure 1, it takes $\lceil \frac{p}{p_{iter}} \rceil$. The original paper was set to `prune_frac = 0.05` , which corresponds to 10 iterations. To reduce the time to a manageable time, we increased `prune_frac= 0.2`, which corresponds to 3 iterations (20% $\rightarrow$ 40% $\rightarrow$ 50% sparsity).

4 Problem Statement and Our Approach

With $p_{iter} = 0.05$ that means in the setting that takes 10 runs to get to 0.5 sparsity, and with finetuning, Bonsai shows great promise. It receives state-of-the-art results in 4/6 tasks on the Huggingface Open LLM Leaderboard in its parameter category. However, one notable exception to this generally good performance is its performance on the GSM-8K dataset, which is a mathematical reasoning dataset (described further in 5with barely achieving 6% accuracy in its best hyperparameter setting.

In our reconstruction, with the settings of $p_{iter} = 0.20$, that means in the setting that takes 3 runs to get to 0.5 sparsity and without finetuning, we receive the results as seen in Table 2. The reasons for deviations taken from the original paper are described in section 3.1

4.1 Approach

Given the low performance as seen in Table 2, our research question is

Method	Adapted	Wikitext-2	BoolQ	HellaSwag	WinoGrande	ARC-e	ARC-c	Avg	GSM-8k (5 shot)
Bonsai	✓	10.92	67.22	43.09	61.64	54.92	26.28	50.63	6.37
Ours	x	**46.594**	**51.66**	**29.07**	**52.13**	**26.54**	**19.75**	**35.83**	0

Table 2: Post-pruning Performance on Various Datasets, pruned with respect to Perplexity on Wikitxt-2. Note that these are not on the same hyperparameters as the paper. Performance on GSM-8K not included in average calculation.

"How can we improve Bonsai's method for better performance of mathematical and logical reasoning tasks?"

In this work, we propose integrating a novel metric into Bonsai's regression-based pruning framework. This new metric takes into account the lexicographical similarity, semantic similarity, and accuracy of a model's generated output compared to the ground truth rationale and answer. We hypothesize that by pruning with respect to this more comprehensive metric, the final sparse network will exhibit improved performance on the GSM-8K dataset and other mathematical and logical reasoning task.

The key idea behind our method is based on the observation that while methods discussed in section 2.3 and other works in the pruning literature prune with respect to perplexity (choosing values such that the perplexity of the new pruned model is smaller).

However, we observe that Bonsai offers a unique flexibility in this regard. Given that it is formated as a under specified regression problem as seen in Equation 2, 1 with respect to some metric U. While other methods require U to have certain desired qualities like differentiability, in this formulation any *reasonable* metric can be a suitable candidate for U. Reasonable metric here implies that the metric is well defined and a high score U corresponds to the desired behaviour of the model. In our case, this allows us to use accuracy with respect to the ground truth, similarity scores as compared to CoT/rationale, or some combination of these metrics.

Thus, let $\{M_i\}_{i=1}^{n}$ be the set of n-desired metric that we want in our final model (accuracy, lexigraphical similarity, semantic similarity, or any task-specific metric). Then we can define

$$U^{\dagger} = \sum_{i=1}^{n} a_i M_i \text{ where } \sum_{i=1}^{n} a_i = 100 \qquad (3)$$

Here a_i refers to the weight of metric M_i in the final metric $U^{\dagger}$. The formulation of Bonsai as a regression problem with respect to this new metric $U^{\dagger}$ remains the same as Equation 2. Note that the desired metrics should be bounded.

We note that metric in Eq 3, can be any combination (non-linear) of $M_i's$. However, in our further study we only consider the linear combinations of $\{M_i\}_{i=1}^{n}$. The details of the combined metric used in our ablations are covered in further detail in Section 6.5. e discuss different combinations of these metrics that we experimented with and the results seen in later sections of the report.

5 Datasets

Given your focus on mathematical and logical reasoning, there are broadly two categories of datasets we looked at (1) focused more on math and (2) focused more on logical reasoning. In this section, we provided a brief description of each of the datasets we used in our evaluations and why we chose them. The performance on these datasets is discussed in 7.

5.1 Mathematical Reasoning Datasets

5.1.1 Grade School Math-8k: (`GSM-8k`)

Cobbe et al. (2021) curated a dataset of Grade School Math problems. The key characteristic of this dataset that we leveraged was the inclusion of a rationale that explains the reasoning between the question and the final answer. Additionally, the final answer to the problem was also identifiable by itself. Thus, they motivated the use of several metrics: accuracy based on the ground truth final answer and similarity metrics on the rationale. While this may seem specific, note that many reasoning datasets do not include both the rationale and answer as separately identifiable information. This dataset remains one of the hardest datasets for most language models, notably the small LMs. An example of the dataset is shown in Figure 2.

5.1.2 MMLU Elementary Math (`hendrycksTest-elementary_mathematics`)

MMLU (Massive Multitask Language Understanding) is a new benchmark that measures pre-trained knowledge in models using zero-shot and few-shot settings, similar to how humans are evaluated. It covers 57 subjects across various disciplines, ranging from elementary to advanced professional levels (Hendrycks et al., 2021b,a). For our analysis, we focus on elementary math problems, as gsm8k proved too challenging for the sparse, non-fine-tuned models we are considering. This subset of MMLU allows us to assess the models' problem-solving abilities and mathematical knowledge at a fundamental level. The following is a sample question from the said dataset. See the Figure 4 for more details.

5.2 Logic Datasets

BoolQ BoolQ is a question answering dataset for yes/no questions containing 15942 examples of naturally occurring questions—they are generated in unprompted and unconstrained settings. Each example contains *(question, passage, answer)* tuple, with some containing the title of the passage (optional additional context). The task is to answer a true or false question based on the given context. In this regard, the task is similar to nlp inference and deduction tasks (Clark et al., 2019).

HellaSwag HellaSwag evaluates the model's common sense reasoning by evaluating the sentence completions given a certain context. The model has to choose the most likely ending to the given sentence. These are completions that require an understanding of everyday activities and human behavior (Zellers et al., 2019).

ARC-e, ARC-c AI2 Reasoning Challenge (ARC) contains easy and challenge question sets based on natural, grade-school science questions of 7,787 questions. This is a multiple choice dataset where most questions contain 4 answer choices (around 1% with more or less than 4 choices) (Clark et al., 2018).

WinoGrande WinoGrande is a collection of 44,000 problems that are formulated as a fill-in-a-blank task with binary options. The goal of this dataset is to test the commonsense and logical reasoning abilities of language models to fill in the right option given the context and knowledge of the world around us (ai2, 2019). See 3 for more details.

Big Bench Logical Deduction BIG-bench is a collaborative benchmark with 204 tasks across various topics, created by over 450 contributors from 132 institutions (bench authors, 2023). Its goal is to push the limits of language models by providing tasks that are considered beyond their current capabilities. For our study, we focus on the "three objects" questions within the Logical Deduction task (`bigbench_logical_deduction_three_objects`). These questions assess an AI's ability to perform logical reasoning and draw conclusions when given information about three objects and their relationships.

The task involves arranging a sequence of N objects based on given conditions. Each puzzle begins with an initial context for the objects and $(N-1)$ conditions that dictate their order. The goal is to determine the absolute position of each object using these conditions. The model's performance is assessed by whether it correctly identifies the position of objects when queried. Puzzles are uniquely solvable and tailored to difficulty levels corresponding to $N = 3, 5, 7$, with higher values indicating increased complexity due to more objects and relationships. We also note that the performance of models (gpt-sota) at the time of the release of the dataset was worse than $1/N$ (random chance). See Figure 5 for details.

5.3 Other Datasets

Wikitext-2 WikiText-2 contains the Good and Featured articles on Wikipedia. The text is not filtered for English, and still contains the original punctuation, casing, and numbers (Merity et al., 2016b).

C4 C4 is a collection of English-based text from the Common Crawl web scrape (Raffel et al., 2019). It is insanely large, using 750 GB. Given this, we only prune using the `c4-train.00000-of-01024` split. We mainly used this dataset as a proof of concept for pruning on logical reasoning. We saw improvement in evaluation, hypothesizing that there are more explanations and reasoning-based text on the web than there is in Wikipedia.

6 Method

6.1 Existing Codebase

Dery et al. (2024) have released a work-in-progress codebase on Github [1] which details their novel gradient-free, perturbative pruning method. Additionally, it includes code for finetuning pruned models using the Wikitext-2 (Merity et al., 2016a) dataset. We did not modify any components of the codebase except a legacy issue; a dependency included in the codebase has since been deprecated, so we made the necessary adjustment to the current version.

6.1.1 Accepted Contributions to the Open Source Project

We were able to recognize multiple errors with the code in its existing forms; the two GitHub issues have been accepted by the authors. The issues pertained to inconsistency in data types and problems with evaluation in electronic datasets. There were certain files that were missing on the codebase that were needed for the evaluation (namely the customized file of lm-eval where the authors made certain edits). The issue regarding updating these files was also accepted.

6.2 Speedup Modifications

After further exploration of the codebase, we made a small but important modification. The evaluation functions are called every time we set a certain mask to the model to decide which combination of masked submodules scores higher based on the formula in 2. However, within these evaluation functions, loading the dataset happens every time the function is called, even though the code is evaluated on the same dataset.

To mitigate this redundancy, we ended up initializing the dataset once and passing in the processed version to the evaluation functions. We saw a **2x speedup** in performance for this seemingly small modification.

6.3 Pruning with Chain-of-Thought Prompt

Borrowing ideas from the section 2.3, our novel contribution to this work is that we utilize the rationale in our pruning method directly.

[1] https://github.com/ldery/Bonsai/tree/main

6.3.1 Dataset Preparation

Instead of pruning with respect to wikitext for just completion as done in the original paper, in the paper, we pruning with respect to the GSM-8k dataset. To curate the dataset for this method to work, we used the GSM-8k Dataset to create our dataset. The dataset is in the question-answer format. First, we extract the rationale and answer based on the splitting by the answer token (given by #### token). Thus, the dataset was a tuple of (question, rationale, answer).

6.4 Prompt Creation

To create the prompt, we first prepend the question with an example to mimic one-shot prompting in the case of pruning. The first question from the training set was cast aside as the example question throughout our experiments. This example is then prepared for all of the training steps.

In addition, we passed an instruction inspired by the Large Language Models are Zero-Shot Reasoners paper (Kojima et al., 2023). We append the instruction to think step by step to the model.

> *Example*:{example-question}, *Rationale*: {example-rationale}, *Answer*: {example-answer}.
>
> + *Question*: {new-question}
>
> + *"Let's think step by step to get the rationale and the answer:"*

This is then passed onto the model to generate rationale and answer. We used the generated output for calculating the metric described in eq (4). The perplexity is also calculated on the same model output.

6.5 Novel Metric

As introduced in Section 4.1, the main contribution of this work is the realization that the metric U can be more task-specific. We integrate our proposed metric in the original bonsai code. Instead of $U = ppl$, our proposed metric has the following structure:

$$U = A * lex\_sim + B * cos\_sim + C * ACC \quad (4)$$

Here lex_sim, cos_sim, and acc represent lexicographical similarity, semantic similarity (as measured by cosine similarity), and accuracy, respectively. $A, B, C \in [0, 100]$ are the weights of each component and represent how much the component contributes to the value of the metric under the constraint $A + B + C = 100$ (see also Equation (3) in Section 4.1). All components of this metric are bounded between 0 and 1, thus we multiply their values by sufficiently large weights for the sake of readability. Each component of the metric is meant to serve a particular purpose:

Lexicographical Similarity (lex_sim) This component focuses on the similarity of words and their arrangement between two texts and is calculated purely based on the surface form of the words in a text, their order, and their syntactic structure. If the model-generated answer and the ground truth answer share many common words in a similar ordering, they would be considered to have high lexicographical similarity. For our purposes, lexicographical similarity can be used to measure the presence of similar equations between texts and the relationships between subjects, eg. does the generated answer contain the same numbers in roughly the same order as the ground truth answer? Are the same actions occurring between the same subjects (ie. "Mary gave Jane 2 apples" versus "Jane gave Mary 2 apples")? However, this component may fail to capture similarity in meaning if the wording or syntax is different between two texts despite underlying concepts being equivalent. We calculate lex_sim using F1 score, which is a function of precision and recall that takes into account the proportion of similar words between the two texts versus the proportion of dissimilar words.

Semantic Similarity (cos_sim) This component evaluates the similarity between texts in terms of their meaning or semantics, irrespective of the specific words used or their arrangement. Even if the model-generated answer and the ground truth answer use different words or structures, they can still be considered semantically similar if they convey the same idea or concept. For our purposes, this provides us with some flexibility regarding the model-generated answer in that there is less pressure to mimic the verbiage or structure of the ground truth. Semantic similarity in our metric is calculated using cosine similarity, the distance between two non-zero vectors defined in some inner product space. We used the `all-MiniLM-L6-v2` HuggingFace checkpoint [2] of Sentence-Bert (Reimers and Gurevych, 2019) to map the generated and ground truth rationales to a 384-dimensional dense vector space. Cosine similarity was then calculated using the following formula:

$$sim(\theta) = \frac{U \cdot V}{||U|| \cdot ||V||}$$

Accuracy (acc) This last component measures whether the ground truth answer is present in the model-generated answer. We calculate accuracy using exact match; $acc = 1$ for a given an example if the ground truth answer is present anywhere in the generated answer as a standalone value (e.g., if the ground truth answer is "12", the string "512" in the generated answer will not be counted as a match).

It was not clear to us how the answer token can be directly extracted from the output. As the model is pruned, it loses its ability to follow instructions, especially when not fine-tuned. This means that we

[2] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

could not rely on splitting by a specific answer token. To overcome these challenges, instead of accuracy, we used exact matches. We note that $EM \geq Accuracy$, and our rationale was that a higher EM score should correspond to a higher accuracy (or at least a higher probability of getting an accurate answer).

The combination of components in these metrics in our combined metric allows us to measure the quality of model-generated rationale as well as the model's ability to arrive at the correct answer. This combination also captures scenarios where only the rationale or the final answer is correct, painting a more holistic picture of the model's performance.

An interesting observation when we initially ran results is that cosine similarity numbers were high, normally within the 0.4 to 0.7 range. Thus, we downweighted this metric slightly, and increased the weighting for lexicographical similarity and accuracy, which showed higher variance. This motivates one of our pruned models on GSM-8K to have weighting `38-24-38`.

Note: For the datasets under GSM-8K that are followed by **a-b-c**, this describes the weighting of lexicographical similarity, cosine similarity, and accuracy in our pruning metric, respectively.

7 Results

For our ablation studies, we ran various combinations of weights in Equation (4). We also experimented with evaluating different datasets (namely GSM-8K and C4) using the original method of perplexity alone. Most of our models were pruned to 0.5 sparsity under the settings described in section 3.1, meaning 50% of the original parameters were removed. This was done to match our baseline and the original Bonsai model's numbers.

Based on Table 3, we note that the baseline pruned model, i.e., our Assignment 3 reproduction, scores highest for the mathematical reasoning datasets. However, our pruned model using perplexity on the GSM-8K dataset yields higher results for most of the logical reasoning datasets. This could be explained by the rationale provided in the GSM-8K dataset, which contains more specific and detailed mathematical reasoning that translates to logical reasoning. See Table 3 for details.

Additionally, we also evaluated certain models with target sparsity 0.4 (retaining 60% of the model weights). Interestingly, pruning with GSM-8K yields higher scores than the baseline across the board when retaining just 10% more parameters. While the results for the sparsity of 0.5 are not significant enough, we see that in the case of 0.4 sparsity, the model pruned on GSM-8K outperforms the baseline as seen in Table 4. The gains are most notable in Arc-e, HellaSwag, and MMLU-Elemntary Math datasets.

To illustrate the difference between the sparsities, consider the dataset point with the question: "*Marco owns an ice cream truck. His ice cream cones are* \$5 *each. If his expenses are* 80% *of total sales for the day, how many ice cream cones would he need to sell to make a* \$200 *profit for the day?*" The given rationale is "*His profit would be 20/100 = 1/5. Total sales for the day is* \$200 × 5 = \$1000. *Total ice cream cones he sells is* \$1000/5 = 200." However, our model with sparsity=0.5 generated something like "A=10000000000000000" which is not a legible answer. However, given the presence of 100 in the output string the similarity scores were decently high. However, we see a noticeable improvement in the output quality when we consider 40% sparsity. The outputted rationale is "80% *of 200 = 160. 160/5*". We note that even though the rationale is not correct, the model gets the math right. However, it is low in all of the three metrics we have used. This more coherent, albeit incorrect, explanation demonstrates that the quality of the generated rationale decreases when sparsity increases in this case, which motivates a lot of the numbers we produced in the tables.

Since the accuracy on GSM-8k across all our considered models was 0; we opted to also test the model on generation on GSM-8k by calculating the perplexity on the test dataset where only the question was provided as the context to the model.

We note that for a tokenized sequence $X = (x_0, x_1, \ldots, x_t)$, then the perplexity of X is, $PPL(X) = \exp\left\{-\frac{1}{t}\sum_i^t \log p_\theta(x_i|x_{<i})\right\}$ where $\log p_\theta(x_i|x_{<i})$ is the log-likelihood of the ith token conditioned on the preceding tokens $x_{<i}$ according to our model. Thus, perplexity is a measure of how well a probability model predicts a sample. It can be thought of as the average number of tokens the model is uncertain about at each step when evaluating a sequence (the lower, the better). We note that the perplexity for our new pruned model is $20x$ lower as seen in Table 5. We note that this observation is specific to the `38-24-38` split and `50-50-0` split and is not seen across all weight combinations. It is possible that the model achieved some local minimum that trims the neurons well enough to achieve a relatively low perplexity on GSM-8k. This is promising for future work because it means that the model is more confident in its predictions even though we were not able to gain accuracy.

8 Discussion

As established before, we wanted to explore a few things related to the effect of datasets on pruned model performance and the effect of the metric. The question is, *Does this study provide any conclusive evidence about how to prune while retaining reasoning abilities? Can we improve the baseline performance of Bonsai using this metic?*. As we can see from Tables 3 and 3, the answers to these questions are mixed. In this section we consider some of the learnings and observations from the ablations. This section focuses mostly on the results on sparsity of

	WikiText-2 ppl	**C4 ppl**	**GSM-8K**	**GSM-8K**	**GSM-8K**	**GSM-8K**	**GSM-8K**
	A3 baseline		**ppl**	**0-0-100**	**38-24-38**	**50-50-0**	$\frac{1}{3}$-$\frac{1}{3}$-$\frac{1}{3}$
WinoGrande	0.5213	0.4929	0.5024	**0.5308**	0.5197	0.4913	0.5087
BoolQ	0.5166	0.5814	**0.6051**	0.4976	0.5466	0.4787	0.5798
ARC-c	0.1975	0.1975	**0.2180**	0.1817	0.1943	0.1975	0.1833
ARC-e	0.2654	0.2338	**0.2828**	0.2433	0.2464	0.2370	0.2370
HellaSwag	0.2907	0.3128	**0.3333**	0.2812	0.3112	0.2954	0.2938
Avg	0.3583	0.3637	**0.3883**	0.3469	0.3637	0.3400	0.3605
GSM-8K acc	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
MMLU	**0.2698**	0.1958	0.2222	**0.2698**	0.2593	0.2593	0.2540
BigBench	**0.3810**	0.3598	0.2910	0.3598	0.3598	0.3598	0.3757
Avg	**0.21693**	0.18519	0.17107	0.20988	0.20635	0.20635	0.20988

Table 3: Performance comparison across logical and mathematical reasoning datasets for target sparsity=0.5. Note that the datasets listed in rows on the left dictate which evaluations were run. The datasets and corresponding descriptions across the top describe what we pruned LLaMa2-7b on. The mathematical reasoning datasets used 5-shot prompting, similar to how it was done with Bonsai (Dery et al., 2024).

	WikiText-2	**GSM-8K**	**GSM-8K**	**GSM-8K**
	ppl	**ppl**	**0-0-100**	**38-24-38**
WinoGrande	0.50079	**0.53081**	0.50237	0.52449
BoolQ	0.58926	0.61295	0.51975	**0.61611**
ARC-c	0.22117	**0.24803**	0.19905	0.21485
ARC-e	0.36493	**0.40916**	0.32543	0.35703
HellaSwag	0.33333	**0.39021**	0.31754	0.33649
Avg	0.4019	**0.4382**	0.3728	0.4098
GSM-8K acc	0.00152	0.00000	0.00000	**0.00455**
MMLU	0.25926	**0.28042**	0.26984	0.24868
BigBench	0.32804	0.38095	0.37566	**0.42857**
Avg	0.19627	**0.33069**	0.21517	0.22727

Table 4: Performance comparison across logical and mathematical reasoning datasets for sparsity=0.4. The table is structured similar to Table 3

	WikiText-2 ppl	**GSM-8K**
	A3 baseline	**38-24-38**
GSM-8K ppl	365.5226	17.1083

Table 5: GSM-8K perplexity performance comparison for specific pruned models with target sparsity=0.5.

0.5; similar trends apply for sparsity 0.4 as well.

8.1 Choice of Dataset for Pruning

The first observation we make is that a richer and more in-domain dataset for perplexity can have a big impact on the performance on downstream reasoning tasks. As described in the 6.3, we incorporated an example and instructions for what we can call guided pruning. We see that in the 0.5 sparsity setting, this leads to an increase in accuracy in 5 out of the 8 datasets we have considered. Surprisingly, the gain was most notable in the datasets that we categorized as more logical reasoning-oriented, whereas the change in performance in the mathematical reasoning tasks was either lower or not significant. It might be that the weights that the method pruned based on perplexity on GSM-8K did not translate well to other tasks of similar difficulty, but were more suitable for datasets that might not need as much abstractive thinking. However, we are unsure of the exact reason why this might be happening.

8.2 Combined Metric Needs Tuning

As defined in eq 4, our metric was a linear combination of semantic similarity, lexicographic similarity, and accuracy. As discussed in section 6.5, accuracy was hard for us to measure as we were unsure how to extract the answer from a free generation of the model output. Therefore, an exact match, as detailed in the section, was used to approximate accuracy. We note that $EM \geq ACC$ (see Limitations). However, in our studies, we saw that even the exact match was very low (close to 0 as the model approached the 0.5 sparsity). Therefore, in most of the iterations, the contribution of the last term in the metric was 0.

We used Sentence-BERT (sentence-transformers/all-MiniLM-L6-v2) with an embedding space of 384 for calculating cosine similarities. We note that we found the semantic similarity, as calculated by this model to to be high across the board for our generated output and rationale. In our testing based on very high similarity scores, we found that the score was high even for two not so related sentences.

For instance, the sentences = ('This is a test that checks different length sentences,' 'I love my cat') gave us a cosine similarity of 0.114 despite not being semantically related. We suspect this model's small embedding space is unable to capture the semantics of the sentences well, especially in our more reasoning-based task. In future work, we want to use the embeddings of the Llama Tokenizer to guide the similarity scores, as we believe these will be richer and be able to capture the differences between two sentences better.

No apparent trend based on the weights: As we can see in the table 4, 3 there is no apparent trend in terms of which measure better encapsulates the differences in performances.

We hypothesize that our steps of trimming 20% of model weights each step might be too drastic for the model. The original paper trims 5% of the weights for each iteration, but due to computing resources and our desire to study different combinations in terms of the make of the metric in Equation 4, we chose this number. However, a more granular pruning step will be better suited to study the important questions like: **What is the effect of each component of the metric in navigating the regression space? What is the sparsity level where the loss in performance to the model size trade-off is at a desired level?**

8.3 Limitations

Resource constraints lead to major limitations in our approach, the most impactful two being the reduced number of tokens in generated responses and the reduced granularity of our pruning iterations. Additionally, our definition of the accuracy component of our metric also presented problems.

Our approach initially involved generating 100 tokens after the initial input prompt, however, during experimentation we found the runtime for $n\_tokens = 100$ to be prohibitively long. To finish all of our experimentation, we were required to cut this number down drastically to $n\_tokens = 20$ for some experiments. This led to a considerable speedup but removed a significant amount of information from our metrics. As expected, for the heavily abridged generated responses, both lexicographical and semantic similarity decreased significantly. With the reduced number of tokens to generate, we occasionally observed instances where the generated output appeared to be on a promising trajectory ("on the right track") toward the correct reasoning before reaching the token limit. As a result, we expect models pruned using the masks generated from shortened generated outputs to perform comparatively poorly at inference time.

As originally described in A3, we were also forced to reduce the granularity of our pruning iterations due to runtime. The original Bonsai paper reached 50% sparsity by performing 10 iterations of pruning 5% of parameters each iteration, whereas we performed 3 iterations pruning 20% of parameters each iteration. This resulted in less computational overhead but also increased the risk of performance degradation since critical information or representations of the original LLM were likely abruptly lost.

Our measure of accuracy in this project was also not the most appropriate. In Section 6.4, we detailed how we determined correctness based on the presence of the ground truth answer string in the generated output. This is clearly a fallible method, as the output "There were $3 * 5 = 15$ apples, they bought 25 in total" would be marked as correct if the ground truth answer was "15", but the model still technically answered incorrectly. Despite our pruned models generally exhibiting 0% accuracy during pruning, this definition of accuracy increases the likelihood that the few non-zero accuracy values we did observe during the pruning process were false positives and spuriously increasing the value of the combined metric.

Though these three major limitations combined reduces confidence in our findings, we hope to mitigate them in future work since the constraints we faced were almost entirely resource-based.

9 Future Work

Despite being inconclusive, we see that pruning with respect to GSM-8K while incorporating an example and rationale leads to higher model confidence as measured by lower perplexity in Table 5 and higher accuracy on logical reasoning datasets as seen in Table 3. There are things we wish to incorporate in future work and to address the limitations of our current approach.

As highlighted in the section above, we would like more granular trimming if we had more computing resources available. This would help us not only gain better performance but also better understand the trade-off between different metrics, model size vs. model performance. In addition, incorporating a richer embedding model to use for cosine similarity would be a better approach going forward.

As we saw in the discussion of model performance, our model's low perplexity was not seen across all combinations of hyperparameters. Given more time and compute, an exploration of hyperparameters, including the ones in the paper and the ones introduced by our metric would be a fruitful study.

On a more theoretical side, Bonsai could be further strengthened by choosing $\beta's$ for the regression problem in Equation (2) more dynamically. We propose the incorporation of Bayesian Linear Regression instead of the simple linear regression that assumes a uniform prior in the current approach.

References

2019. Winogrande: An adversarial winograd schema challenge at scale.

BIG bench authors. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Lucio Dery, Steven Kolawole, Jean-Francois Kagey, Virginia Smith, Graham Neubig, and Ameet Talwalkar. 2024. Everybody prune now: Structured pruning of llms with only forward passes. *arXiv preprint arXiv:2402.05406*.

Dan Hendrycks, Collin Burns, Steven Basart, Andrew Critch, Jerry Li, Dawn Song, and Jacob Steinhardt. 2021a. Aligning ai with shared human values. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021b. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jie Huang and Kevin Chen-Chuan Chang. 2022. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*.

Seungone Kim, Se June Joo, Doyoung Kim, Joel Jang, Seonghyeon Ye, Jamin Shin, and Minjoon Seo. 2023. The cot collection: Improving zero-shot and few-shot learning of language models via chain-of-thought fine-tuning.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2023. Large language models are zero-shot reasoners.

Liunian Harold Li, Jack Hessel, Youngjae Yu, Xiang Ren, Kai-Wei Chang, and Yejin Choi. 2023. Symbolic chain-of-thought distillation: Small models can also "think" step-by-step.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016a. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016b. Pointer sentinel mixture models.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Siyu Ren and Kenny Zhu. 2022. Specializing pre-trained language models for better relational reasoning via network pruning. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2195–2207.

Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. 2023. Distilling reasoning capabilities into smaller language models.

Ying Su, Jipeng Zhang, Yangqiu Song, and Tong Zhang. 2024. Pipenet: Question answering with semantic pruning over knowledge graphs. *arXiv preprint arXiv:2401.17536*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2023. A simple and effective pruning approach for large language models. *arXiv preprint arXiv:2306.11695*.

Yueqing Sun, Qi Shi, Le Qi, and Yu Zhang. 2021. Jointlk: Joint reasoning with language models and knowledge graphs for commonsense question answering. *arXiv preprint arXiv:2112.02732*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023a. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

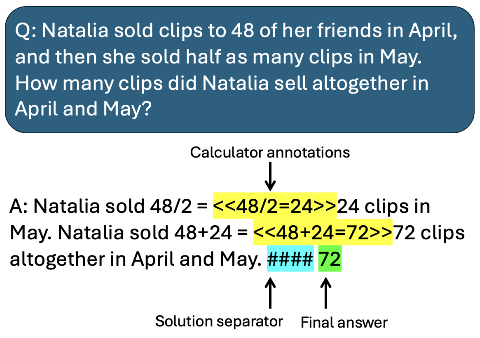
Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Mengzhou Xia, Zexuan Zhong, and Danqi Chen. 2022. Structured pruning learns compact and accurate models.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence?

Aojun Zhou, Yukun Ma, Junnan Zhu, Jianbo Liu, Zhijie Zhang, Kun Yuan, Wenxiu Sun, and Hongsheng Li. 2021. Learning n:m fine-grained structured sparse neural networks from scratch.

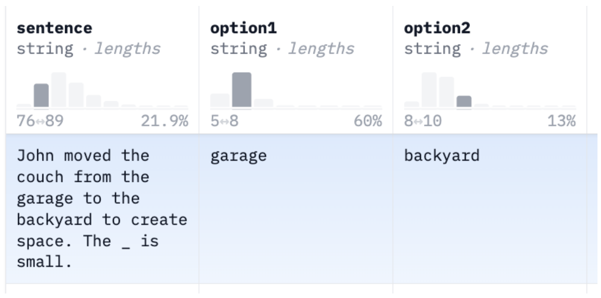
A Appendix



Q: Natalia sold clips to 48 of her friends in April, and then she sold half as many clips in May. How many clips did Natalia sell altogether in April and May?

A: Natalia sold 48/2 = <<48/2=24>>24 clips in May. Natalia sold 48+24 = <<48+24=72>>72 clips altogether in April and May. #### 72

Figure 2: GSM8k example from Cobbe et al. (2021)

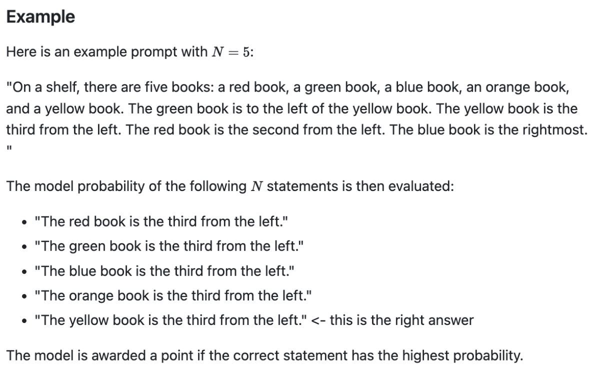


sentence	option1	option2
string · lengths	string · lengths	string · lengths
76↔89 21.9%	5↔8 60%	8↔10 13%
John moved the couch from the garage to the backyard to create space. The _ is small.	garage	backyard

Figure 3: WinoGrande Dataset Example Questions

Question: Ms. Perez drove a total of 40 miles in 5 days. She drove the same number of miles each day. How many miles did Ms. Perez drive each day?
Choices: ["5", "7", "8", "9"].
Answer: C

Figure 4: MMLU dataset example



Example

Here is an example prompt with $N = 5$:

"On a shelf, there are five books: a red book, a green book, a blue book, an orange book, and a yellow book. The green book is to the left of the yellow book. The yellow book is the third from the left. The red book is the second from the left. The blue book is the rightmost. "

The model probability of the following N statements is then evaluated:

- "The red book is the third from the left."
- "The green book is the third from the left."
- "The blue book is the third from the left."
- "The orange book is the third from the left."
- "The yellow book is the third from the left." <- this is the right answer

The model is awarded a point if the correct statement has the highest probability.

Figure 5: Big-Bench Logical Deduction Task Example
Note that the model is rated based on the answer with the highest probability(bench authors, 2023) (source github-logical-deduction)