THE BLINDSPOT OF CATEGORICAL CROSS-ENTROPY

Anonymous authors
Paper under double-blind review

ABSTRACT

Categorical cross-entropy (CCE) on softmax outputs has been the cornerstone of modern machine learning systems and large language models. Despite its central role, we uncover a systematic blind spot where erroneous probability mass can reside disproportionately long in the top rank prediction for high-entropy states. This means that as entropy increases, the encouragement of actually demoting invalid top-rank predictions diminishes. We prove that this undesirable bias is a fundamental property of learning distributions via CCE, and proceed to empirically demonstrate its existence across various settings. Using controllable synthetic settings, we are able to explicitly track this inefficiency, and find that introducing neural networks tend to further exacerbate this issue. This discovery holds true for both dense neural networks and autoregressive Transformers trained with CCE for next-token prediction. Moreover, we find no indication that this disproportionately slow learning for high entropy states disappears as we scale the number of model parameters. Simply up-weighting the loss to counteract this slow learning in high entropy states does not result in any perceivable improvements. However, introducing consistency for high entropy states can significantly quicken the learning of good top ranks. Finally, we investigate the recently discovered hyperfitting phenomenon and find that its counterintuitive results can be understood from a similar principle, as it provides a training environment with extreme consistency, allowing the model to circumvent the CCE blindspot.

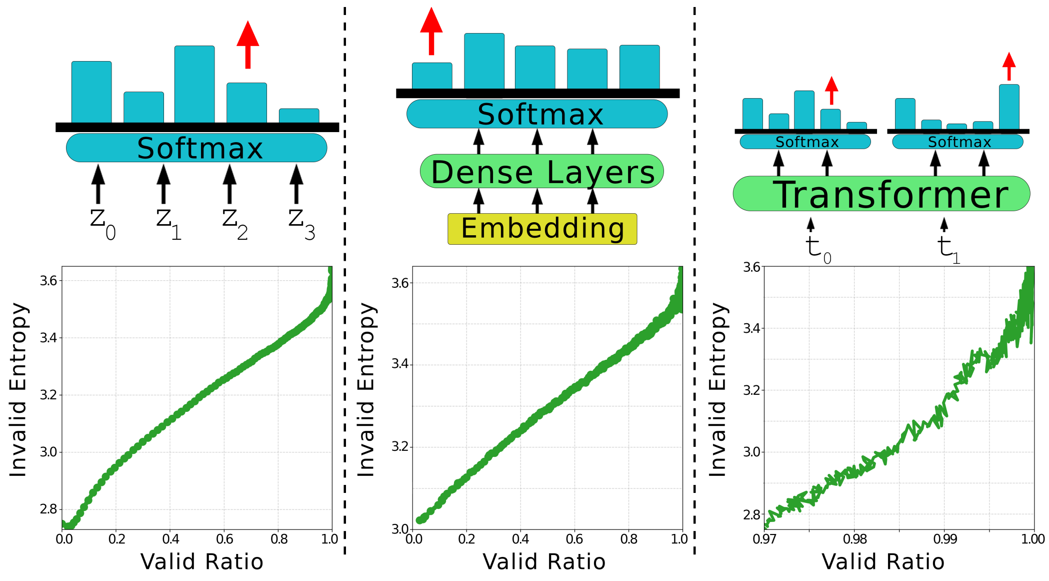

Figure 1: Illustration of three methods with increasing complexity, that all exhibit the same slow learning of high entropy states when trained with CCE. Bottom part shows the ratio of valid samples vs. the mean entropy of the remaining invalid samples. **Left**: Direct CCE optimization of logits. **Middle**: Small neural network to memorize independent distributions. **Right**: A 10M Llama transformer architecture on sequential data.

1 Introduction

Categorical cross-entropy (CCE) coupled with next-token prediction has been the backbone of training modern large language models (LLM)s (Brown et al., 2020; Dubey et al., 2024; Bi et al., 2024). This simple optimization objective has enabled efficient pre-training, where both model size and data quantity can be scaled. Although this has resulted in previously unseen capabilities across numerous tasks, the theoretical understanding of LLM pre-training remains limited (Tian et al., 2023; Chang et al., 2024). To add to this, pre-trained LLMs still exhibit some surprising undesirable behavior, often warranting additional post-training phases and methods (Welleck et al., 2019; Stiennon et al., 2020; Bai et al., 2022).

In this paper we uncover a key insight in the standard LLM pre-training optimization process. We demonstrate that CCE on softmax yields an undesirable training setting for high entropy states, where getting *any* valid token to emerge as the top rank requires a disproportional number of updates as compared to low entropy states. CCE correctly incentivizes the model to learn when to provide high entropy predictions, but it does not properly incentivize the model to demote noisy top rank tokens for high-entropy states. We refer to this as the **CCE blindspot**.

Using synthetic data where entropy can be explicitly controlled we empirically demonstrate how the CCE blindspot occurs across three levels of complexity: direct logit optimization, small MLPs, and transformers (Vaswani et al., 2017; Dubey et al., 2024). Unfortunately, we find that neural networks have a tendency to exacerbate the slow learning in high entropy states, and that the trend remains as the number of model parameters increases. Furthermore, we find that simply up-weighting the loss to counteract the CCE blindspot to be unsuccessful. However, using an oracle model as a learning target for high entropy states, results in significantly faster learning of valid top rank tokens. Finally, using our knowledge of the CCE blindspot we investigate the recently discovered hyperfitting phenomenon, and demonstrate how its counterintuitive positive results stem from how it circumvents the CCE blindspot.

This paper follows a bottom-up approach, and is structured as follows: Section 3 proves the existence of the CCE blindspot at the most fundamental level, Section 4 demonstrates how introducing neural networks exacerbates this issue, Section 5 uses sequential data to demonstrate that this issue is not resolved by increased model size and complexity, and finally Section 6 makes use of the knowledge established in previous sections to provide insights into the hyperfitting phenomenon.

2 Related Work

A growing body of work reveals that CCE introduces subtle but important biases. In linear settings, gradient descent under CCE has been shown to implicitly align logits with log-odds while maximizing margins (Thrampoulidis, 2025), and in self-attention layers, CCE optimization produces SVM-like separation patterns across tokens (Li et al., 2024). When applied to next-token prediction, CCE has been linked to subspace collapse in contextual embeddings (Zhao et al., 2025), and capacity analyses demonstrate near-tight limits on the number of context-to-distribution mappings that transformers can memorize (Madden et al., 2025). Further studies of next-token prediction under CCE also highlight pitfalls stemming from teacher forcing and training–inference mismatch, as well as the inconsistent decoding behaviors that arise from CCE-trained distributions (Bachmann & Nagarajan, 2024; Trauger & Tewari, 2025).

Entropy plays a central role in learning with CCE and the training of LLMs; however, prior work reports conflicting effects, and a unified understanding has yet to emerge. In accordance with Maximum-entropy principles (Jaynes, 1957), it has been shown that increases in entropy during RL training results in stronger LLM reasoning (Cheng et al., 2025; Wang et al., 2025). Conversely, directly minimizing entropy minimization has also proven effective as a training signal itself, for both RL and fine-tuning settings (Agarwal et al., 2025). Most strikingly, the hyperfitting phenomenon shows that aggressively collapsing predictive entropy, by overfitting to near-zero training loss, can improve generative capabilities despite poor validation performance (Carlsson et al., 2025). This dual role of entropy as both resource and liability frames the blindspot of softmax-CCE: whereas prior accounts focus on distributional fit or representational geometry under teacher forcing, we study how CCE updates redistribute probability mass in near-tie regimes, and why high-entropy states slow the demotion of invalid top-1 tokens.

3 The Blindspot of Softmax Categorical Cross-Entropy

High entropy, by definition, entails the uncertainty of correctly guessing *the* answer. It does *not* mean that it is difficult to produce *a possible* answer. This crucial distinction should be captured by a modeled distribution, and although the probability mass may be spread across many tokens, the top token should correspond to a possible continuation. Intuitively, producing a possible answer is a simpler task than modeling the full distribution, and arguably more closely related to what should occur during autoregressive generation.

Unfortunately, we discover that softmax CCE is disproportionately slow at modeling *possible* answers in high entropy scenarios. This means that, for high entropy states, the predicted top rank may remain erroneous, even if all updates promote valid tokens. Section 3.1 is dedicated to empirically demonstrating this undesirable phenomenon, and Section 3.2 provides theoretical explanation of why this occurs.

3.1 A minimal experiment: Independent Distributions

As a fundamental scenario to demonstrate the existence of the blindspot behavior, we use CCE to directly optimize the logits of softmax distributions using gradient descent. We do this by randomly assigning a set of label tokens for the distribution, then track the number of updates required to get *any* of the assigned labels to the top rank.

Each distribution contains 1000 randomly initialized logits and $[1, 40]$ randomly assigned label classes. During each optimization step, we randomly select one of the assigned labels for each distribution, and optimize towards it with CCE and a fixed learning rate. This means that samples with only a single label will always train towards that label. Distributions with multiple assigned labels will randomly alternate between these, but each CCE update is still towards a valid label. We perform this experiment for three different learning rates: $5\text{e}{-3}$, $1\text{e}{-3}$ and $1\text{e}{-4}$. For each learning rate we perform the experiment for 10.000 independent distributions. The results are visualized in the left part of Figure 2, showing the average number of required updates for a given number of available labels.

For each of the three learning rates, the number of required updates increases linearly in terms of available labels. For example, with a learning rate of $1\text{e}{-3}$ it takes ≈ 800 updates to learn *any* valid top rank if there are 40 available labels, but less than 200 updates if there are 5 available labels. Furthermore, the slope of this trend is clearly correlated with the learning rate. Indeed, since all distributions are independent it would be possible to achieve a valid top rank after a single update with sufficiently high learning rate. However, as shown in Section 4 (and the right part of Figure 2) this option is not viable when neural networks are involved.

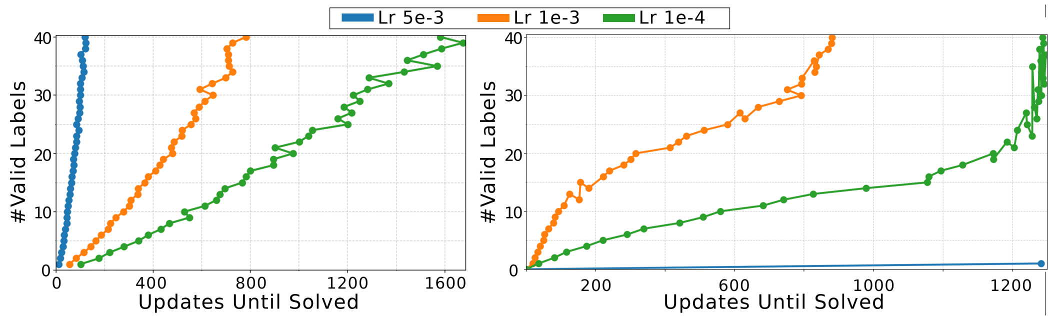

Figure 2: Required number of updates until *any* of the valid labels are in the top rank. Displayed numbers are the average for the 10% slowest samples.

3.2 The underlying dynamics

We now proceed with a principled investigation of the underlying factors of the CCE blindspot behavior. Formally, we start with a logit vector $Z = \{z_i\}_{i=1}^{V}$ that produces a probability distribution $P = \{p_i\}_{i=1}^{V}$ via the softmax function. Let k be the index of the top-ranked token such that $p_k = \max_i p_i$, and let t be the index of a target token we wish to push to become the top rank.

Our goal is to find an update vector $\Delta = \{\Delta_i\}_{i=1}^{V}$ so that $Z' = Z + \Delta$ results in the equality $p'_t = p'_k$, as any infinitesimally additional push achieves this. Of course, many possible Δ achieve this, so as a best-case scenario, we investigate the Δ vector that results in the minimum absolute change $\|\Delta\|_1 = \sum_i |\Delta_i|$. The Δ_t required for this to occur can be derived as follows:

$$\Delta_t = \log\left(\frac{p_k}{p_t}\right). \tag{1}$$

Now consider a sequence of events where we first push the probability of an additional valid target p_t^*, followed by a push on p_t. Regardless of the logit Δ during the initial push, as p_t^* is increased by $\Delta_{p_t^*}$ the total sum of probability in other tokens must also decrease with $\Delta_{p_t^*}$. Assuming the decrease in probability for each token to be proportional to its probability mass, we can formulate the new probabilities via the shrinkage factor $c \in (0, 1)$:

$$p'_i = c\,p_i \quad \text{for all } i \neq t^*, \qquad c = \frac{1 - p_{t^*} - \Delta_{p_t^*}}{1 - p_{t^*}}, \quad \Delta_{p_t^*} \in \big(0,\, 1 - p_{t^*}\big), \tag{2}$$

Finally, we can use Equation 1 to calculate the minimum logit push Δ'_t required to push p_t, after we have pushed p_t^*. As shown below in Equation 3, the push Δ'_t remains identical to Δ_t. The initial push towards p_t^* does therefore not contribute to p_t becoming top rank, although p_k decreases. CCE updates towards different targets are therefore *independent* of each other, in regards to getting *any* valid target into the top rank. Hence, the linear increase in updates as seen in Section 3.1, as we alternate between an increasing number of targets.

$$\Delta'_t = \log\left(\frac{p'_k}{p'_t}\right) = \log\left(\frac{c\,p_k}{c\,p_t}\right) = \log\left(\frac{p_k}{p_t}\right) = \Delta_t, \tag{3}$$

4 CCE for Neural Networks

The complex training dynamics of neural networks include many factors not accounted for by the theoretical investigation of Section 3.2. Therefore, we provide further empirical studies of how this phenomenon behaves when the complexity increases. In particular, we compare how the probability shifts during the optimizer updates, and how these shifts differ depending on the entropy.

This scenario introduces neural networks with an Adam optimizer (Kingma & Ba, 2014) into the task of Section 3.1, but is otherwise identical. The samples are represented by learnable embeddings, where each corresponding distribution is generated by propagating the embedding through a tiny neural network that outputs a softmax distribution. Each embedding is randomly initialized using Gaussian noise, and identically to Section 3.1 each sample is uniformly assigned $[1, 40]$ out of 1000 available classes.

We perform this experiment for three different learning rates: $5\text{e}{-3}$, $1\text{e}{-3}$ and $1\text{e}{-4}$. The network contains 3 dense hidden layers with a dimensionality of 1024 and each sample embedding has 256 dimensions. For all three runs we iterate over the 10.000 samples using CCE, a batch size of 256, and the Adam optimizer.

4.1 Results

The right part of Figure 2 displays the number of updates required to get a valid top rank, for all samples corresponding to a given number of labels. For clarity of the overall trend, the displayed number of updates required is the average for the 10% slowest samples, instead of only the slowest.

The higher learning rate of $1\text{e}{-3}$ achieves valid top ranks quicker than $1\text{e}{-4}$. However, a learning rate of $5\text{e}{-3}$ does not result in any meaningful learning at all. Hence, this run is denoted with a single straight line at $Y = 0$. As expected, training neural networks with too high learning rate collapses the training, even when simply trying to get a good prediction at the top rank, unlike the linear scenario of Section 3.1.

Similar to the independent distributions, the runs with $1\text{e}{-3}$ and $1\text{e}{-4}$ has a clear trend of learning good top ranks for the low entropy states first, and then incrementally higher and higher entropy states. However, the increase in time is no longer linear. Rather, the network is quick to learn good top ranks for the low entropy states, but then follows a prolonged duration where learning is slower, followed by a short period where the network learns top ranks for the remaining high entropy states. The introduction of neural networks has therefore exacerbated the problem, and the undesirable behavior can no longer be explained solely by the alternating targets described in Section 3.2.

4.2 Probability shifts during training

Using the run with the learning rate of $1\text{e}{-4}$, we investigate how probability mass is shifted during training, and how this differs across samples with different numbers of available labels. Therefore, we track the immediate change in logits and probabilities after an update, on the same samples that were just trained on. The results are divided into 4 percentile bins based on the number of available labels. For example, Q_1 corresponds to $[1, 10]$ available labels, and Q_4 to the $[31, 40]$. Additionally, we provide filtered versions of each bin where the target token was not already in the top rank, meaning $p_k \neq p_t$. These bins are denoted by Q_i^*.

Starting from 1000 epochs we collect distribution shift data from the 10 subsequent epochs, resulting in 100 batch updates and 100,000 total samples. Target Prob Δ denotes the increase in probability for the target token, and $|Logit\Delta|$ the total sum of absolute shifts across all logits. All metrics are averaged across all 100.000 sample updates and are available in Table 1.

Notably, the $|Logit\Delta|$ across the different bins remains similar, regardless of the corresponding CCE loss. Indeed, the loss is more than 3 times higher in Q_4 than in Q_1. We hypothesize that this logit shift consistency is an effect of parameter sharing, and lack of modularization. Since all samples propagate through the same weights, there may not be sufficient separation to perform an update that disproportionally changes the logits of some samples. But regardless of the cause however, this means that although CCE correctly incentivizes more learning on high entropy samples, the model is unable to surgically achieve this.

Unlike the $|Logit\Delta|$, the difference in target prob Δ is striking, with Q_1 moving 35 times more than Q_4. As further explained in 4.3, this difference can be partially explained as a property of the softmax function itself, where high entropy distributions require a larger logit push to yield the same probability push. We note that it is theoretically possible to also achieve good top ranks via decreasing the probability of the top token. However, considering our observations that we *do* learn top ranks slower for high entropy samples, such effects do not seemingly play a major part in the learning.

4.3 Entropy and logit sensitivity

In Section 4.2 we observed that the total logit Δ is fairly consistent across different entropy regions, despite the significant difference in loss. Due to the nature of the softmax function, tokens with low

Table 1: Distribution behavior of a CCE trained MLP model, over different entropy regions.

Entropy Bin	Q_1	Q_2	Q_3	Q_4	Q_1^*	Q_2^*	Q_3^*	Q_4^*
Top Rank Acc	0.34	0.08	0.05	0.02	0	0	0	0
Entropy	2.17	4.67	5.48	5.74	3.24	5.22	5.66	5.81
CCE	1.66	3.67	5.1	5.9	2.10	5.16	5.66	5.80
Target Prob Δ	$7\text{e}{-3}$	$2\text{e}{-3}$	$4\text{e}{-4}$	$2\text{e}{-4}$	$1\text{e}{-3}$	$7\text{e}{-4}$	$5\text{e}{-4}$	$2\text{e}{-4}$
|Logit Δ|	43	40	38	37	39	37	37	37

probability are less sensitive to a given logit push. Unfortunately, this is the opposite of what we desire with regard to pushing valid tokens to the top rank of high entropy states, where the token probability is lower.

This property of the softmax function can be understood as follows. Let x_t denote a surgical logit Δ for our target token, so that $z'_t = z_t + x_t$.

$$p'_t = \frac{e^{z_t + x_t}}{\sum_{j \neq t} e^{z_j} + e^{z_t + x_t}} = \frac{p_t e^{x_t}}{(1 - p_t) + p_t e^{x_t}} \implies \frac{dp_t}{dx_t} = p_t(x_t)\left(1 - p_t(x_t)\right) \tag{4}$$

Equation 4 lets us calculate how sensitive p_t is to a given Δ of its corresponding logit. Below are some qualitative examples of how this sensitivity differs in terms of p_t, where we keep $x_t = 0.1$. As can be observed, $p_t = 0.5$ is $25\times$ more sensitive than $p_t = 0.01$. In conclusion, the model is less responsive to small logit updates when the probability is already very small (and also very high). More details and pedagogical derivations are available in Appendix A.3.

$$\Delta p_{0.01}\,(0.1) = 1\text{e}{-3}, \qquad \Delta p_{0.1}\,(0.1) = 1\text{e}{-2}, \qquad \Delta p_{0.5}\,(0.1) = 2.5\text{e}{-2},$$

5 CCE FOR SEQUENTIAL DATA

Having established the CCE blindspot, and how neural networks have the tendency to exacerbate this issue, we proceed to demonstrate how this can occur in more realistic scenarios. Therefore, we train transformer models on sequential data using CCE via next-token prediction, and validate on held-out validation data. These experiments thus deal with more complex models and data that stem from an underlying shared distribution, rather than memorization of independent samples.

By using synthetically generated sequences we can explicitly determine if tokens are valid according to the data, or undesirable artifacts of the optimization process. This evaluation is performed for two different scenarios as visualized in the right part of Figure 3. We refer to these as: **Gen** and **No-Gen**. For the **Gen** scenario the model is given a 16 token context, from which it generates a continuation by greedily selecting the top token at each subsequent time step, until a 128 token sequence is completed. For the **No-Gen** scenario we feed the model a full existing sequence, and independently investigate the top rank predictions at each time step. This means that for the No-Gen scenario, the input to the model is always valid and in the same distribution as the training data. Due to the simplicity of the task, a baseline that randomly selects tokens will achieve $\approx 93.6\%$ valid steps for the No-Gen scenario, and have ≈ 9.72 repetitions per sequence for the Gen scenario.

For all experiments we use the LLaMA transformer architecture, and train using a batch size of 128 with the AdamW optimizer. We perform this on three different model sizes: 1M, 5M and 10M. More details and hyperparameters are available in Appendix A.2.

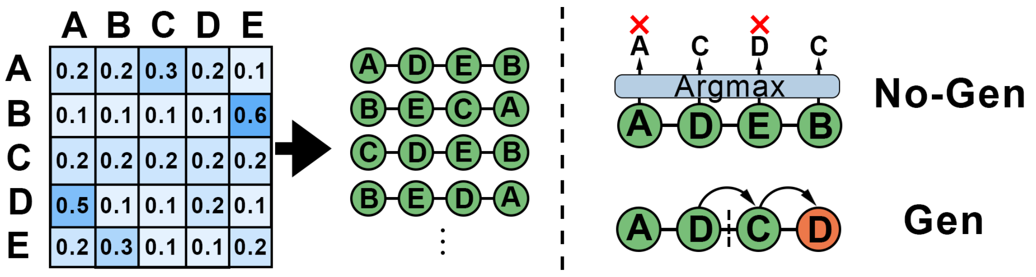

Figure 3: **Left:** Illustration of the oracle transition matrix used to generate sequences without any repetitions. **Right:** Visualizations of the Gen and No-Gen evaluation scenarios.

5.1 Sequential Data Generation

The sequence data is generated via an autoregressive oracle model consisting of a token-based transition matrix, along with a hard constraint of no repeating tokens in the same sequence. This simple constraint enables us to determine if a predicted token is valid or truly out-of-distribution. Specifically, if a predicted token is invalid if it already exists at a previous time step in the same sequence, as this never occurs in the data.

We set the vocabulary size to 1000, entailing a 1000×1000 transition matrix. Essentially this matrix is a bigram model, each row corresponding to a token in the vocabulary, and determines the transition probability for the next token. By design, this matrix is randomly initialized so that the entropy of each row is uniformly spread over our target span of $[0.7, 4.0]$. This is achieved by first uniformly sampling a target entropy in the span for each token, then randomizing a corresponding vocabulary size long probability distribution matching the target entropy.

When generating data, we start by uniformly selecting the first token, then sample subsequent tokens using the transition matrix and the no-repetition rule. At each time step we mask out the probabilities of previous tokens, and normalize the transition vector before sampling. We generate 20 Million synthetic sequences, each with a length of 128 tokens, and keep 1000 sequences for validation.

5.2 Sequential CCE Results

Right part of Figure 4 shows the average number of generated tokens that break the underlying no-repetition constraint for the Gen evaluation scenario. All models initially perform significantly worse than the random baseline. Eventually however, 5M and 10M learn to generate fully valid sequences, whilst 1M sees no improvement. However, as demonstrated in the left part of Figure 4, the CCE difference between 1M and 10M models is fairly small, with the final epoch of 10M having ≈ 0.05 lower train and validation loss. This means that the 1M is fairly adequate according to the CCE metric although the actual top ranks, as exposed via the greedy decoding, contain a lot of noise.

Figure 5 contains results for the No-Gen scenario where the models are fed full sequences. All models start identical to the random baseline but quickly achieve valid top ranks for the lower entropy time steps, as noted by the early increase in valid predictions and the entropy for invalid steps. After this, 5M and 10M improve the ratio of time steps with valid top rank predictions, with 10M learning at a faster pace. Most importantly however, there is still a clear trend where high entropy states are the last to get valid top ranks. This relationship is visualized for the 10M model in Figure 1.

In conclusion, we find that the CCE blindspot appears even in complex model settings, and its effect can be observed on held-out data. Although we observe that bigger models learn good top ranks faster, there is still a trend of disproportionately learning slower good top ranks for high entropy states. For example, the 10M model achieves 99% valid time steps in 1.5 epochs, then it spends another 1.5 epochs for the remaining high entropy steps. Moreover, if the model is not capable enough to capture the dataset complexity, the CCE curves might look good but high entropy states may still contain noisy top ranks. More complex data may therefore produce similar situations for larger models.

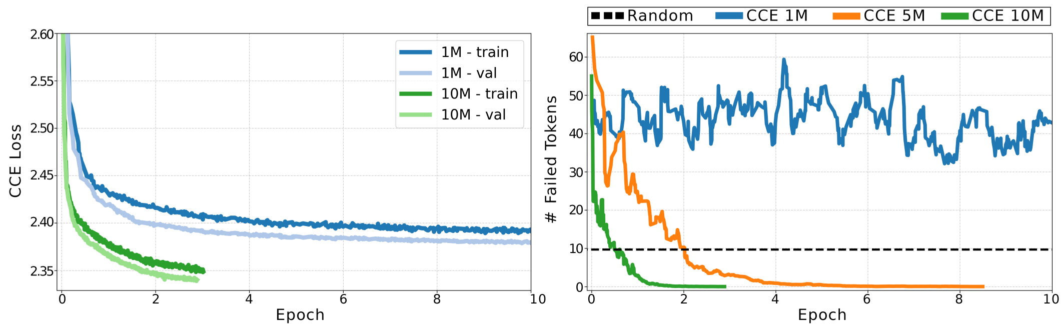

Figure 4: **Left:** CCE loss for training and validation set, for the 1M and 10M Transformer models. **Right:** The average number of repeated tokens when generating from contexts in the validation data.

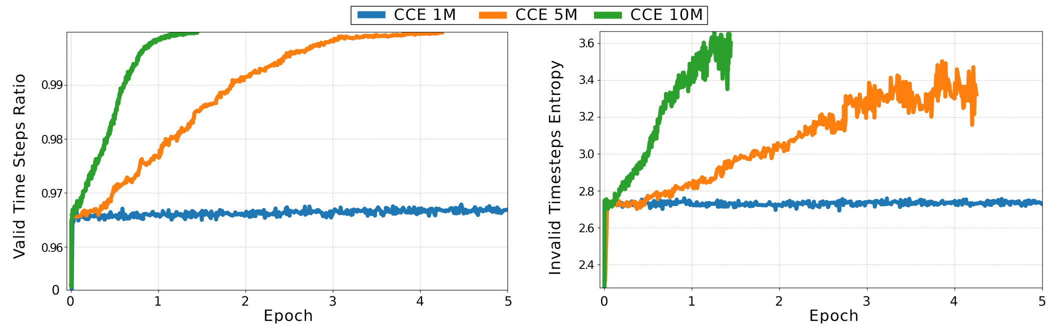

Figure 5: Results for the No-Gen setting. **Left:** Ratio of time steps for where the top-1 predicted token is not a repetition. **Right:** The average oracle entropy for the time steps where the model fails, and has a repetition as the top-1 token.

5.3 Loss Modifications and Investigations

Considering the unsuccessful 1M run of Section 5.2, we investigate the possibility of modifying the standard CCE loss to address the disproportionately slow learning of top ranks in high entropy states. We therefore train a 1M model, using two investigative loss functions, that directly modifies the training for the 25% highest entropy time steps, corresponding to $Q4$. Both these methods utilize the oracle model, meaning they are not directly translatable to real scenarios.

The first method, **Entropy Scaling**, uses the oracle to further exaggerate the CCE loss by upweighting the loss for time steps where the true entropy is high. For each batch, the time steps with the 25% highest entropy has their corresponding CCE loss multiplied by a factor of 2. The second method, **Entropy Consistency**, uses the oracle to provide a consistent target for high entropy states, regardless of the next target in the sequence. For time steps with the 25% highest entropy, the CCE prediction target is replaced to the most likely token according to the oracle. Note that this does not break the causal nature for subsequent steps, as the input sequence remains intact and valid. A visualization of each method is available in Appendix A.1.

Identical to the Non-Gen scenario of section 5.2, the left part of Figure 6 contains the average number of time steps where the top rank is valid. The right part of the figure 5.3 contains the logit Δ for Q_1 and Q_4. It is clear that the further CCE weighting of the Entropy Scaling method does not improve the result, and that entropy consistency does. The positive results for the entropy consistency method indicates that reducing the perceived entropy of top 25% highest entropy states, is enough to make the model learn good top ranks for all states. Interestingly, the difference in logit Δ between Q_1 and Q_4 is slightly higher for the entropy scaling. This corroborates our hypothesis about neural networks being limited in their ability to surgically update logits of specific time steps.

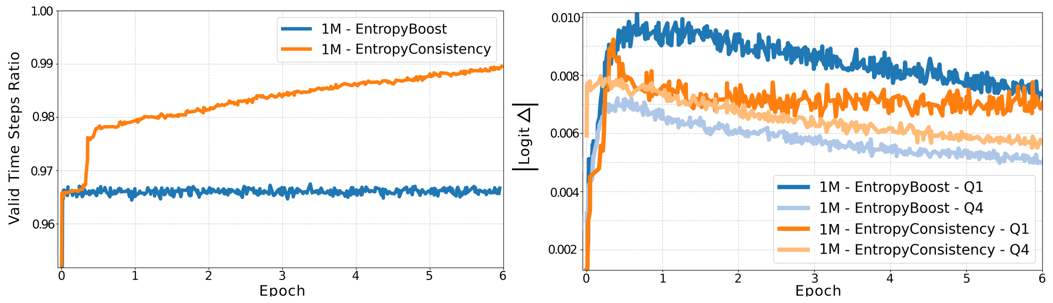

Figure 6: **Left:** Ratio of valid time steps for the No-Gen setting. **Right:** The absolute logit delta for different entropy regions, where Q_1 corresponds to the 25% lowest, and Q_4 the 25% highest.

Table 2: Behavior during Hyperfitting for a 1M Transformer pre-trained for 1 epoch.

Hyperfitting Epoch	0	100	200	300	400
Train CCE	2.57	1.81	0.88	0.26	0.06
Val CCE	2.42	3.03	4.33	6.16	8.13
Train Top-1 Acc	0.36	0.5	0.75	1.0	1.0
Val Top-1 Acc	0.36	0.3	0.25	0.22	0.22
Valid Ratio	0.96	0.96	0.96	0.96	0.96
Invalid Entropy	2.72	2.62	2.56	2.53	2.53
#Invalid Gen Tokens	56.32	11.52	10.24	7.6	7.5

6 Hyperfitting

Finally, we turn towards established LLM training dynamics and use the CCE blindspot to provide further insights. In particular we consider the hyperfitting phenomenon Carlsson et al. (2025), which occurs when pre-trained large language models experience a boost in generative sequence capabilities when overfitted on a small set of samples. On held-out data this results in a remarkably higher perplexity, but also a counterintuitive improvement of generative fluency, seemingly defying the conventional bias–variance tradeoff. In particular, the greedy decoding capabilities improve, a behavior that usually results in degenerative behavior. Additionally, a relevant effect of hyperfitting is that it results in the model producing very low entropy predictions for all time steps.

Our thesis is that the counterintuitive benefits associated with hyperfitting stem from its ability to address pre-training issues originating from the CCE blindspot, by providing consistency analogous to the Entropy Consistency of Section 5.3. There is an important distinction between the underlying distribution that data is generated from, and what the data actually expresses. The near-zero training loss of hyperfitting entails that the model is able to memorize its small training set and always predict the next token correctly. This perfect accuracy collapses the perceived entropy of the dataset, and exposes the model to a scenario where it is able to push tokens in a consistent direction each update. Most notably, the original experiments of Carlsson et al. (2025) focus on greedy decoding, a setting the authors argue is specifically sensitive to noisy top-ranks.

6.1 Hyperfitting on Synthetic Sequential Data

Using the data and models from Section 5, we hyperfit on 2K randomly selected synthetic sequences by training for 500 epochs. We do this for both the 1M and 5M CCE trained models, starting from the corresponding models epoch 1. The 20M model was omitted as it had already converged to a flawless state by this checkpoint, in terms of valid top tokens. More hyperparameters are available in Appendix A.2.

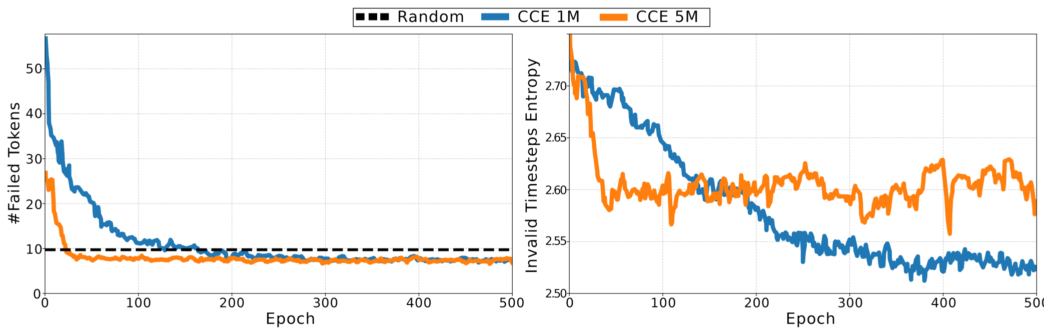

Figure 7: Hyperfitting results. **Left:** Average number of repeated tokens for the Gen evaluation setting. **Right:** Average oracle entropy for failed time steps in the No-Gen evaluation setting.

Figure 7 contains the number of invalid tokens when generating (left), and the entropy for invalid tokens when not generating (right). Both models see a strong increase in generative performance, along with a decrease in entropy of invalid tokens. Although 5M has a better starting point, both models converge to a similar value of ≈ 7.5 failed tokens per generated sequence. Table 2 displays various metrics for the 1M model across its run. As expected the validation CCE loss increases as training loss goes to zero and training top-1 accuracy goes to one. Interestingly, in terms of the valid time steps when not generating, the ratio stays consistent. This indicates that Hyperfitting mainly improves a model's robustness and ability to perform after having shifted out-of-distribution.

7 Discussion & Future Work

In Section 5.2 we discovered that top-rank validity can differ greatly between two models that virtually have the same CCE loss. This explains a familiar practical gap where LLMs with orders of magnitude more parameters, may show only modest CCE improvements compared to smaller models, but still yield drastically better performance on downstream tasks. Considering the importance of CCE for LLM training, this insight provides a direction for potential future improvements.

It is well known that most pre-trained models exhibit degenerative behavior when using greedy decoding. We hypothesize that the CCE blindspot is a strong contributor to this. If the top token is noisy, it means that the most likely token to be sampled during generation would push the model out of distribution. Notably, this does not explain why we in Section 5.2 observe that models are more prone to repetitions than a random baseline, even before any training occurs. Considering that all our experiments involved data without any repetitions, we conclude that data is not the reason for this bias, and something in the models themselves poses a bias towards repetitions. This contradicts the theory that repetition bias stems from repetitions found in natural language (Fu et al., 2021; Holtzman et al., 2020), and may therefore be an interesting avenue for future work.

All our experiments concern rather small models compared to today's LLM standards. Due to computational limitations we are unable to perform experiments for such large models. We do note however that the trend of high entropy states learning disproportionally slower, remained through our experiments that scale from 1000 independent parameters to 10 million network parameters. However, future work may investigate if anything unexpected happens to the CCE blindspot as models get significantly larger.

In order to explicitly and irrefutably track the validity of predicted top ranks, all our experiments utilized toy-setting and synthetic data. Having established the CCE blindspot empirically and theoretically, we now encourage future work to investigate its effect on real data. Admittedly, such work is intrinsically hard, as real data often lacks a corresponding oracle model. We therefore propose developing methods that perform well in synthetic settings without the usage of an oracle, and test their transferability to real data. To this end we release our code for synthetic data experiments, that provides an easy interface to switch out components such as loss function, optimizer, or model architecture. The code is available at: <— *Anonymized Link* —>.

8 Conclusion

We identified and analyzed a systematic blind spot of softmax-CCE: in high-entropy regimes, invalid top-1 predictions persist much longer due to (i) alternation over multiple valid targets and (ii) diminished softmax sensitivity at small p_t. We empirically demonstrated this phenomenon at three levels of complexity: direct logit optimization, small MLPs, and transformers on sequential data. For direct logit optimization the increase in time to learn a good top rank, is linearly correlated to the number of available labels. However, as we introduced neural networks we found that the time may increase significantly more for high entropy samples and the trend to be less predictable.

Directly up-weighting the loss for high-entropy tokens did not yield improvements. We speculate that this is due to how shared network parameters diffuse the effect of per-token reweighting, thereby prohibiting localized logit movement. However, synthetically increasing the consistency for prediction targets in high entropy states did result in clear improvements. Furthermore, we argue that the effects of Hyperfitting can be explained by a similar reduction of target entropy since overfitting on a small set of samples removes any ambiguous time steps, thereby sidestepping the blind spot.

REFERENCES

Shivam Agarwal, Zimin Zhang, Lifan Yuan, Jiawei Han, and Hao Peng. The unreasonable effectiveness of entropy minimization in llm reasoning. *arXiv:2505.15134*, 2025. URL `https://arxiv.org/abs/2505.15134`. v1.

Gregor Bachmann and Vaishnavh Nagarajan. The pitfalls of next-token prediction. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 2296–2318. PMLR, 21–27 Jul 2024. URL `https://proceedings.mlr.press/v235/bachmann24a.html`.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022. URL `https://arxiv.org/abs/2212.08073`.

Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wenjie Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, A. X. Liu, Bo Liu, Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Junxiao Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Minghui Tang, Bingxuan Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Y. Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yiliang Xiong, Hanwei Xu, R. X. Xu, Yanhong Xu, Dejian Yang, Yuxiang You, Shuiping Yu, Xingkai Yu, B. Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghua Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. Deepseek llm: Scaling open-source language models with longtermism. *CoRR*, abs/2401.02954, 2024. URL `https://arxiv.org/abs/2401.02954`.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *ArXiv*, abs/2005.14165, 2020. URL `https://api.semanticscholar.org/CorpusID:218971783`.

Fredrik Carlsson, Fangyu Liu, Daniel Ward, Murathan Kurfali, and Joakim Nivre. The hyperfitting phenomenon: Sharpening and stabilizing llms for open-ended text generation. *arXiv:2412.04318*, 2025. URL `https://arxiv.org/abs/2412.04318`. v2.

Hoyeon Chang, Jinho Park, Gichan Lee, Seonghyeon Ye, and Minjoon Lee. How do large language models acquire factual knowledge during pretraining? In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective on reinforcement learning for llms. *arXiv:2506.14758*, 2025. URL `https://arxiv.org/abs/2506.14758`. v3.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal

Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vítor Albiero, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The Llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. URL `https://arxiv.org/abs/2407.21783`.

Zihao Fu, Wai Lam, Anthony Man-Cho So, and Bei Shi. A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 12461–12469, 2021.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rygGQyrFvH`.

Edwin T. Jaynes. Information theory and statistical mechanics. *Physical Review*, 106(4):620–630, 1957.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Yingcong Li, Yixiao Huang, M. Emrullah Ildiz, Ankit Singh Rawat, and Samet Oymak. Mechanics of next token prediction with self-attention, 2024. URL `https://arxiv.org/abs/2403.08081`.

Liam Madden, Curtis Fox, and Christos Thrampoulidis. Next-token prediction capacity: general upper bounds and a lower bound for transformers, 2025. URL `https://arxiv.org/abs/2405.13718`.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021, 2020.

Christos Thrampoulidis. Implicit optimization bias of next-token prediction in linear models. In *Proceedings of the 38th International Conference on Neural Information Processing Systems*, NIPS '24, Red Hook, NY, USA, 2025. Curran Associates Inc. ISBN 9798331314385.

Yuandong Tian, Yiping Wang, Beidi Chen, and Simon Du. Scan and snap: Understanding training dynamics and token composition in 1-layer transformer. In *Advances in Neural Information Processing Systems*, volume 36, pp. 71911–71947, 2023.

Jacob Trauger and Ambuj Tewari. On next-token prediction in llms: How end goals determine the consistency of decoding algorithms, 2025. URL `https://arxiv.org/abs/2505.11183`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pp. 5998–6008, 2017.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv:2506.01939*, 2025. URL `https://arxiv.org/abs/2506.01939`.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. *CoRR*, abs/1908.04319, 2019. URL `http://arxiv.org/abs/1908.04319`.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/2020.emnlp-demos.6`.

Yize Zhao, Tina Behnia, Vala Vakilian, and Christos Thrampoulidis. Implicit geometry of next-token prediction: From language sparsity patterns to model representations, 2025. URL `https://arxiv.org/abs/2408.15417`.

A APPENDIX

A.1 CCE LOSS MODIFICATIONS

Following is a description of the two CCE loss modifications used in Section 5.3. For both explanations we let t_n denote the ground-truth token at time step n, and let E_n denote the oracle entropy at the same step. Figure 8 provides a visual demonstration of both loss modifications.

For the **Entropy Scaling** method, we up-weight the contribution of high-entropy time steps. Define the set of indices corresponding to the top quartile of oracle entropies:

$$Q4 = \{n : E_n \in \text{top } 25\%\}.$$

The modified loss becomes

$$\mathcal{L}^{\text{scale}} = \frac{1}{N} \sum_{n=1}^{N} w_n \, \mathcal{L}_n, \quad w_n = \begin{cases} 2, & n \in Q4, \\ 1, & \text{otherwise.} \end{cases}$$

For the **Entropy Consistency** method, the CCE loss is calculated as normal, but the ground-truth token for high-entropy steps is replaced with the oracle's most likely prediction:

$$t_n^* = \begin{cases} \arg\max_j \, p_{\text{oracle}}(j \mid t_{<n}), & n \in Q4, \\ t_n, & \text{otherwise.} \end{cases}$$

Both methods preserve the autoregressive structure of the input sequence but alter how loss is attributed at high-entropy steps.

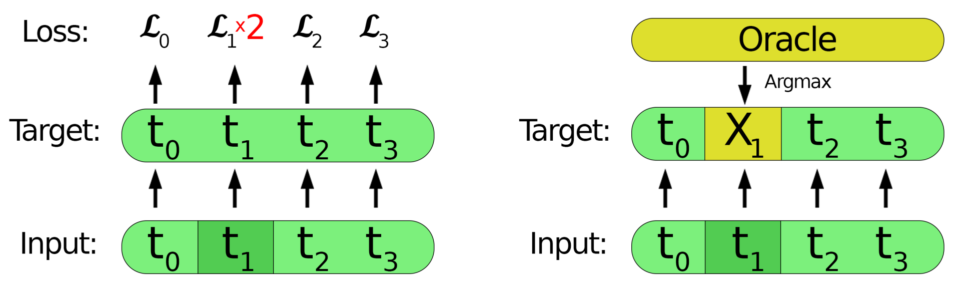

Figure 8: A pedagogical illustration of the CCE loss modifications. For both examples the time step t_1 has an oracle entropy that belongs in the top 25%. **Left:** Entropy Scaling method, where the loss of high entropy states is increased. **Right:** Entropy Consistency method, where the prediction target for high entropy states is changed into the corresponding Oracle argmax for that state.

A.2 TRANSFORMER HYPERPARAMETERS

All transformer experiments utilize the Huggingface transformer (Wolf et al., 2020) implementation of the Llama architectureDubey et al. (2024), and utilize the Adam optimizer with a learning rate of $2.5\text{e}{-5}$. For any non-Hyperfitting training we utilize a batch size of 128, and for Hyperfitting we follow Carlsson et al. (2025) and use a batch size of 8. Table 3 contains a list of the architecture settings for our different models.

Table 3: Model architecture configurations at different parameter scales.

Model	**1M**	**5M**	**10M**
Hidden Size	128	256	512
Intermediate Size	512	1024	2048
#Layers	2	4	4
#Attention Heads	8	8	8

A.3 PROBABILITY GAIN FROM A LOGIT PUSH

Following is a pedagogical analysis of the results in Section 4.3, where we discuss how tokens with higher probability are more sensitive to logit changes. The aim is therefore to investigate how much target token's probability mass increases when we increase its corresponding logit.

We consider the set of logits $Z = \{z_i\}_{i=1}^{V}$ that produces a probability distribution $P = \{p_i\}_{i=1}^{V}$ via the softmax function. Let t denote the target token, and x_t the surgical logit Δ for our target token, so that $z'_t = z_t + x_t$.

Running through the softmax with the updated probability for token t, we get:

$$p'_t = \frac{e^{z'_t}}{\sum_{j=1}^{V} e^{z'_j}} = \frac{e^{z_t+X}}{\sum_{j\neq t} e^{z_j} + e^{z_t+X}}. \tag{5}$$

Let $\mathcal{Z} = \sum_{j=1}^{V} e^{z_j}$ denote the original partition function. This allows us to write:

$$p_t = \frac{e^{z_t}}{\mathcal{Z}} \implies e^{z_t} = p_t\mathcal{Z} \implies \sum_{j\neq t} e^{z_j} = \mathcal{Z} - e^{z_t} = \mathcal{Z}(1-p_t). \tag{6}$$

Substituting back gives:

$$p'_t = \frac{p_t\mathcal{Z}\, e^X}{\mathcal{Z}(1-p_t) + p_t\mathcal{Z}e^X}. \tag{7}$$

Canceling $\mathcal{Z}$, we arrive at an expression depending only on the current probability p_t and the logit push X:

$$p'_t = \frac{p_t e^X}{(1 - p_t) + p_t e^X}. \tag{8}$$

The local sensitivity to a given push is therefore:

$$\frac{dp'_t}{dX} = p'_t(X)\,\bigl(1 - p'_t(X)\bigr). \tag{9}$$

Having established how the updated probability p'_t depends only on the current probability p_t and the logit push X, we now turn to the question of *sensitivity*. Figure 9 shows how this sensitivity changes for a fixed $X = 0$. Clearly, the sensitivity is highest when p_t is ≈ 0.5, and vanishes as $p_t \to 0$ or $p_t \to 1$. We can therefore conclude that the distribution is less responsive to small logit updates when the probability is already very small or very big. This explains why high entropy distributions shifts their probability mass slower compared to low entropy distributions, since even the most likely token has a low probability when the entropy is high.

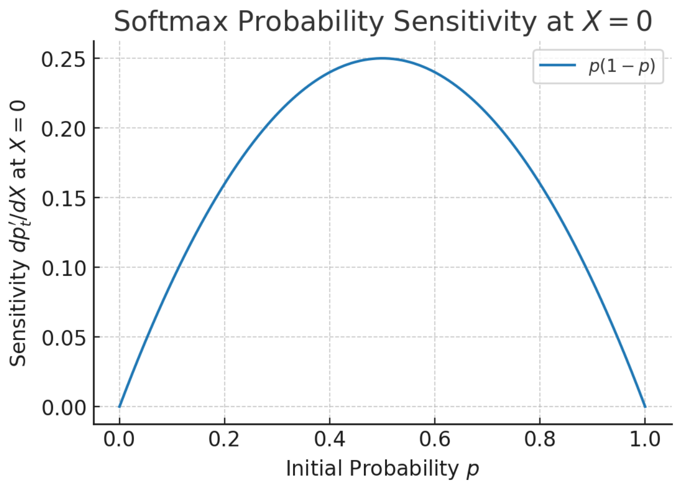

Figure 9: Sensitivity of the probability with respect to a logit push at $X = 0$, showing the local slope $dp'_t/dX = p_t(1 - p_t)$.

A.4 LLM Usage in paper writing

LLMs were used in the creations of this paper primarily for spell checking and proofreading. In addition, they were occasionally employed to help identify relevant sources of information for the related works section. The main model used were GPT-5.