LEARNING IS FORGETTING: LLM TRAINING AS LOSSY COMPRESSION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the increasing prevalence of large language models (LLMs), we still have a limited understanding of how their representational spaces are structured. This limits our ability to interpret how and what they learn or relate them to learning in humans. We argue LLMs are best seen as an instance of lossy compression, where over training they learn by retaining only information in their training data relevant to their objective(s). We show pre-training results in models that are optimally compressed for next-sequence prediction, approaching the Information Bottleneck bound on compression. Across an array of open weights models, each compresses differently, likely due to differences in the data and training recipes used. However even across different families of LLMs the optimality of a model's compression, and the information present in it, can predict downstream performance on MMLU-Pro, letting us directly link representational structure to actionable insights about model performance. In the general case the work presented here offers a unified Information-Theoretic framing for how these models learn that is deployable at scale.

1 INTRODUCTION

We still have a limited understanding of *how* Large Language Models (LLMs) achieve impressive results across a wide array of tasks (Devlin et al., 2019; Grattafiori et al., 2024). While a growing body of work interprets LLMs using behavioural experiments, probing, or causal interventions, the scale of these models makes understanding how their representation spaces are distributed a continued challenge. Here we look at an LLM as an instance of lossy compression, offering an account of how models represent information during training and what information matters for performance.

Lossy compression represents data efficiently by preserving only the information from a source relevant to a goal. While audio recordings intended for human listeners can be gigabytes in size, MP3 files save space by discarding frequencies typically outside the range of human hearing (Jayant et al., 1993); similarly, a JPEG file omits subtle colour variations that are difficult for the human eye to perceive. We draw a parallel with LLMs, which are expected to generate responses humans prefer, after being trained on trillions of tokens – more language data than a human hears in 200 lifetimes. More generally, compression is thought to underpin learning in humans and models (see Feldman, 2016), giving a formal account of LLM training in terms of compression allows us to work towards a unified theory of representation learning. We present results showing that over the course of pre-training LLMs optimally compress the information present in their training data for next sequence prediction.

Compression is inherently opinionated – some information from the source is preserved, some is forgotten to save space. Information Theory (Shannon, 1948) provides a formal language to describe this process, letting us both quantify the information present in a representation *and* compute a bound where it is optimally compressed with respect to the data it represents. Our results build on the Information Bottleneck (IB) theory of deep learning (Tishby & Zaslavsky, 2015), showing pre-training follows a two phase trajectory: first increasing mutual information with the training objective, before compressing input information. Across a wide array of LLMs we find each model compresses differently, with the optimality of a model's compression and the information it preserves predicting performance on downstream benchmarks.

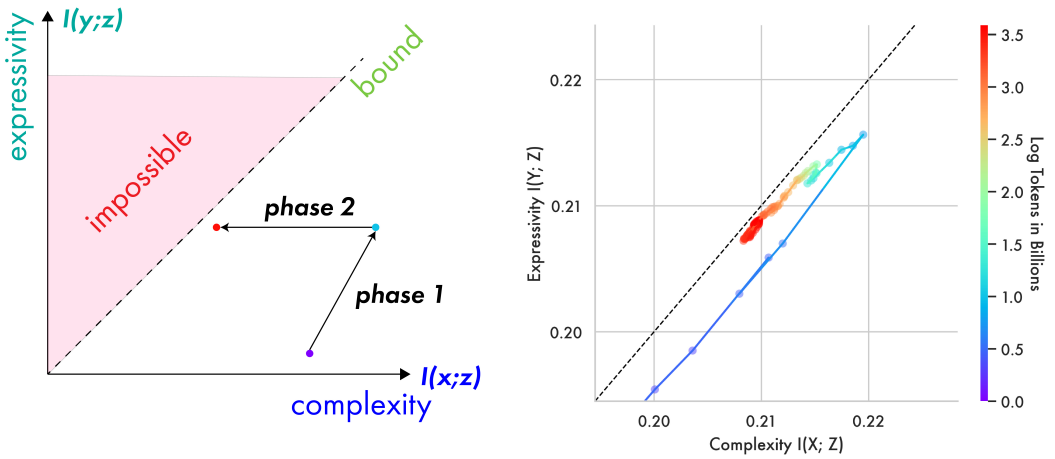


Figure 1: **LLMs Learn an Optimal Compression of the Internet** (Left) An illustration of the information plane, with mutual information of hidden representation z with input x as complexity, and with output y as expressivity. Shown is the training trajectory hypothesised by the Information Bottleneck theory of deep learning (Tishby & Zaslavsky, 2015): phase 1 increases $I(y; z)$ then phase 2 compresses irrelevant $I(x; z)$ approaching the bound on compression. Previously this had only been documented in relatively small neural networks. (Right) The information plane for pre-training of the OLMo2 7B model. The horizontal axis shows mutual information between representations and the input, the vertical axis shows mutual information with the predicted output. Hue indicates timepoint in training in terms of log tokens in billions. Estimates are based on 10,000 examples from the C4 dataset.

A hallmark of large-scale distributed systems, like neural networks, is that they are difficult to understand as a function of their parts alone (Anderson, 1972; Mitchell, 2009). Our approach to interpretability allows us to consider learning and generalisation at the scale of an entire model, rather than studying individual circuits or neurons within it. Additionally it allows us to frame how models do so well at so much in terms of existing theories of learning and compression, while providing actionable insights at LLM scale.

In what follows we focus on offering concrete answers to three questions: Do LLMs optimally compress their representations? What information survives that compression? What representational structures drive performance? In summary, the core findings are:

- Pre-training dynamics for LLMs closely follow theoretical predictions from the Information Bottleneck, with models first expanding representations before slowly approaching optimal compression.
- Scale conditions these dynamics, with smaller models (below 7 billion parameters) struggling to achieve meaningful compression later in training.
- How optimally compressed a model is correlates significantly with performance on MMLU Pro across three families of large language models, letting us directly relate representation structure to behaviour.
- Post-training increases human preference information in a model, with the proportion of preference information also predicting performance on MMLU Pro
- Finally, we compare a wide array of open-weight models across 5 model families, showing they all converge near optimal compression.

2 Background & Related Work

2.1 Learning, Inference, and Compression

Compression has been argued to underpin learning and inference in humans (Chater, 1997; Chater & Vitányi, 2003; Feldman, 2000; Pothos & Chater, 2001) and models (MacKay, 2003; Poggio et al., 2004). Increasingly, probabilistic inference and complexity minimisation are seen as deeply intertwined (Feldman, 2016) – a point perhaps made clearest by Bayesian inference, which implicitly prefers the simplest hypotheses consistent with observed data (Edwards, 1972; Jeffreys, 1939; Vitányi & Li, 2000). Bayesian approaches to human cognition offer accounts of how a broad array of human behaviour can be productively thought of as this kind of inference (see Griffiths et al., 2024, for a review). In machine learning Occam's Razor has long been used as a model selection criterion – where the best model is the simplest one consistent with the data (Burnham & Anderson, 2002; Rissanen, 1978; Wallace & Boulton, 1968). The bias variance trade-off (Geman et al., 1992) makes this explicit in the context of neural networks, showing more complex models may achieve better fit to the training data, but they also generalise worse than their simpler counterparts. While some work has studied whether or not LLMs can match lossless compression algorithms in-context (e.g. Delétang et al., 2023), this is distinct from giving an account of LLM training itself as a process of lossy compression – the object of study here. It is worth noting that there is not universal agreement about how to assess compression (see MacKay, 2003, for discussion), but here we follow in the information-theoretic tradition (Shannon, 1948).

2.2 Rate Distortion Theory

Consider a function E that encodes an input X in a representation Z, $Z = E(X)$. This representation is then decoded by a function D to produce predictions $\hat{Y}$ for an output with true label Y, $\hat{Y} = D(Z)$. Assuming that X and Y are not independent, if E were to losslessly preserve all the information from the input, we would expect D to be able to precisely recover the corresponding output, with $\hat{Y} = Y$. Rate Distortion Theory (RDT) (Shannon, 1948) instead considers the *lossy* case $\hat{Y} \neq Y$, where some amount of error in the prediction – **distortion** – is acceptable. It then becomes a question of how much information about the input – termed the **rate** – the encoder needs to preserve to achieve a given level of distortion.

The Information Bottleneck (IB) Tishby et al. (2000) looks at a particular case, where the **rate** is given as the mutual information between inputs and their representation $I(X;Z)$, and **distortion** as the mutual information between a representation and the corresponding target prediction $I(Y;Z)$ – the 2D space this creates is called the *information plane* (shown in Figure 1). Since $I(X;Z)$ reflects how much information about the input space is preserved it can be referred to as *complexity* (e.g. Zaslavsky et al., 2018). Likewise $I(Y;Z)$ is referred to as *accuracy* given it quantifies how much information a representation has about the target output it will be used to predict. To distinguish this quantity from behavioural accuracy (e.g., exact match on a task) we refer to it as *expressivity* – how uniquely a representation can refer to its target (in line with Kirby et al., 2015). Optimal compression within the IB occurs when an encoding $Z|X$ preserves only the information about X relevant to predicting Y, or when $Z|X$ minimises

$$\mathcal{F}_\beta[p(Z|X)] = I(X;Z) - \beta I(Y;Z) \tag{1}$$

where β is a trade-off parameter controlling the allowable level of distortion. When β approaches 0 all inputs are compressed to a single point, as $\beta \to \infty$ we approach the lossless case, where using X or its encoding Z tells us the same information about Y; $I(Y;Z) = I(Y;X)$. The curve traced by varying β draws a bound, where the encoding $p(Z|X)$ is optimally compressed – everything above the curve is unachievable and everything below it is suboptimal. This bound starts off with a linear relationship where $I(X;Z) = I(Z;Y)$, until Z captures all information shared between X and Y. Intuitively, in an optimal encoding each additional bit of complexity gets you an additional bit of accuracy, until all information shared by X and Y are represented; $I(Y;Z) = I(Y;X)$ – in the cases studied here all models stay well below this saturation point, so for clarity we refer to the bound as the line $I(X;Z) = I(Z;Y)$ (for further discussion of the bound and computing it numerically see Appendix E.6).

Applying the Information Bottleneck to Deep Learning Tishby and Zaslavsky (2015) offered a theoretical characterisation of training a multi-layered neural network as optimising an Information Bottleneck. They theorise two phases of training: first a fitting phase during which representations increase mutual information with the target labels $I(Y;Z)$; and second a compression phase, during which models compress irrelevant information about the input $I(X;Z)$ and in the process begin to approach the optimal bound. It is this latter phase that is hypothesised to result in representations that generalise robustly. Figure 1 shows the theoretically hypothesised trajectory a model takes through the information plane over the course of training.

Shwartz-Ziv and Tishby (2017) confirmed the two-phase predictions from the IB theory of deep-learning empirically in feed forward networks trained on MNIST. Subsequent work has questioned the generality of these findings, showing how – at least in linear networks – the compression phase can be driven by the type of non-linearity used (Saxe, Bansal, et al., 2019), or that compression is not necessarily required for generalisation (Goldfeld et al., 2019). It remains unclear whether deep-learning models in the general case can be expected to follow the phases of expansion and compression predicted by the IB, in particular when it comes to sequence models (e.g. Transformers) trained on complex tasks.

2.3 Interpreting Neural Networks

A broad literature on the theory of deep learning tries to give an accounting of learning dynamics in small multi-layer networks (e.g. Frankle & Carbin, 2018; Saxe, McClelland, & Ganguli, 2019). While there has been some extension of these kinds of representational analyses to larger models – like applying information theoretic methods to transformers (Voita et al., 2019) – much of the work on interpretability in LLMs leverages behavioural or probing evidence. Behavioural approaches treat models as akin to psycholinguistic subjects (Futrell et al., 2018, 2019), taking model outputs as behaviours (Hu et al., 2020; Marvin & Linzen, 2018; Warstadt et al., 2019). Probing (Pimentel et al., 2020; Veldhoen et al., 2016; Voita & Titov, 2020) trains a smaller model – like a linear classifier – to predict labels from a model's latent representations, as evidence that information relevant to those labels is present. While valuable, these approaches are removed from the models' representations themselves – characterising downstream behaviours rather than characterising the representational structures that drive them.

Mechanistic interpretability follows in a similar vein but aims to describe how circuits within a model implement the functions that solve a task. These analyses have given accounts of how two layer linear and non-linear models represent features from synthetic data (Elhage et al., 2021) or how single-layer attention only transformers solve modular addition (Nanda et al., 2023). When deployed at scale, to LLMs, this work often relies on training unsupervised probes termed *sparse auto-encoders* (Elhage et al., 2022) to identify correspondences between parameters and different words or concepts from the training data (Bricken et al., 2023). In the general case this work often focuses on 'mono-semanticity' – looking for lossless, one-to-one correspondences between input features and parts of a model. More recently studies of when features emerge during pre-training have aligned with the expansion/compression pattern described by the IB theory (Ge et al., 2025).

To be sure, there is an abundance of methods for analysing deep-learning models. Here, however, we highlight a disconnect between work on the theory of learning in humans and neural networks, and work on interpretability. Interpretability methods can be deployed at scale on complex models and tasks, but lack clear relationship to existing theoretical work. In the sections that follow we operationalise Rate Distortion Theory, and related work on learning as compression, at LLM scale. This allows us to analyse training dynamics while contextualising our conclusions in existent and well-studied theoretical frameworks. Our approach represents one that is theoretically motivated but can be applied to any model at any scale.

3 Methods

3.1 Entropy Estimation

Let $T \in \mathbb{Z}^{B\times S}$ be a batch of B tokenized samples with sequence length S, drawn from a corpus of text data $\mathcal{T}$, and let E be a model with L layers and representation dimension h; the corresponding encoded representations are $Z \in \mathbb{R}^{L\times B\times S\times h}$. Let $X \in \mathbb{Z}^{B\times S}$ be feature labels for the text in T.

For example, when we look at optimal compression with respect to the IB bound, these labels X are the token ids for the model inputs; however, when analysing representation information more generally, these can be other input features, such as preference label or language id. It is desirable to compute the mutual information $I(X;Z)$ using Shannon entropy as opposed to differential entropy to accomplish this, previous work quantises Z into n bins, to get a discrete encoding $\hat{Z}$ (Shwartz-Ziv & Tishby, 2017; Voita et al., 2019)[1]. Unfortunately the approaches from this previous work have memory and resource requirements that make them difficult to apply at LLM scale. As a result we use the soft-entropy estimator from Conklin (2025) – this is an efficient differentiable relaxation of a binning-based estimate that has been shown to converge to the true entropy of a distribution.

To obtain a soft quantisation $\hat{Z}$, this approach first computes $\bar{Z}$, which is the normalization of Z to lie on the surface of the unit sphere $\mathbb{S}^h$ in $\mathbb{R}^h$. It then samples n points $\{w_i\}_{i=1}^n$ uniformly at random from $\mathbb{S}^h$.[2] Then, for each normalized representation $\bar{z} \in \mathbb{R}^h$, we compute a vector whose i^{th} entry is the cosine between $\bar{z}$ and w_i, then apply softmax to that vector – softly assigning each embedding $\bar{z}$ to the points in W. More formally, for each $(l, b, s) \in [L] \times [B] \times [S]$, tensor $\bar{Z}$ (whose shape coincides with Z) is defined so that $\bar{Z}_{l,b,s,:} = Z_{l,b,s,:}/\|Z_{l,b,s,:}\|$, and we stack the uniform samples $\{w_i\}_{i=1}^n$ into a matrix $W \in \mathbb{R}^{h\times n}$:

$$\{w_i\}_{i=1}^n \sim \text{Unif}(\mathbb{S}^h), \qquad W_{:,i} = w_i. \tag{2}$$

Tensor $\check{Z} \in \mathbb{R}^{L\times B\times S\times n}$ is then defined so that for $(l, b, s) \in [L] \times [B] \times [S]$,

$$\check{Z}_{l,b,s,:} = \text{softmax}\Big(\sum_{j=1}^{h} \bar{Z}_{l,b,s,j} W_{j,:}\Big). \tag{3}$$

Each vector $\check{Z}_{l,b,s,:}$ defined this way is a probability vector. Let $\hat{Z} \in \mathbb{R}^{L\times n}$ be the matrix obtained from tensor $\check{Z}$ by averaging over the batch and sequence dimensions, and let $\hat{z}_l$ be the l-th row of this matrix, a probability vector of length n by construction:

$$\hat{Z} = \frac{1}{BS}\sum_{b=1}^{B}\sum_{s=1}^{S} \check{Z}_{:,b,s,:}, \qquad \hat{z}_l = \hat{Z}_{l,:}, \quad H(\hat{z}_l) = -\sum_{j=1}^{n} \hat{z}_{l,j} \log \hat{z}_{l,j}. \tag{4}$$

Vectors $\hat{z}_\ell$ are probability vectors for each layer $l \in [L]$ describing a categorical distribution over n categories. Therefore we can compute the Shannon entropy $H(\hat{z}_l)$ as above. Due to the normalisation step during quantisation, this distribution approximates the probability that a representation in a layer l lies along a particular angle with respect to the origin. To estimate the entropy in an entire model, denoted $H(Z)$ we average entropy across layers. Efficiency (Wilcox, 1967) normalises H by the entropy of a uniform distribution $\log(n)$, thereby bounding the entropic quantity between 0 and 1 – to aid interpretability here we convert $H(Z)$ to an efficiency $\mathcal{H}(Z)$ by additionally normalising by the entropy of a uniform distribution at each layer. These definitions can also be conditioned on the feature labels X.

$$\mathcal{H}(Z) := \frac{1}{L\log(n)}\sum_{l=1}^{L} H(\hat{z}_l), \qquad \mathcal{H}(Z|X=x) := \frac{1}{L\log n}\sum_{l=1}^{L} H(\hat{z}_l|X=x) \tag{5}$$

This now allows us to efficiently compute the mutual information[3] between input features X and encodings across an entire model, regardless of model size.

$$I(X;Z) := \frac{1}{|X|}\sum_{x\in X} \mathcal{H}(Z) - \mathcal{H}(Z|X=x) \tag{6}$$

[1] For discussion of Shannon entropy and why previous approaches are not scalable see Appendices E.4,E.5.

[2] This is equivalent to sampling from an isometric h-dimensional multivariate normal, $\tilde{w}_i \sim \mathcal{N}(0, Id_h)$, and scaling to unit length, $w_i = \frac{\tilde{w}_i}{||\tilde{w}_i||}$.

[3] This quantity is not strictly the mutual information $I(X;Z)$ because we are weighting each label equally, rather than weighting by $P(X = x)$. Since our main interest is in token ids as labels, this choice avoids weighting highly frequent tokens like "the" and "a" more heavily than less frequent ones.

3.2 Mutual Informations

To look at whether or not a model is optimally compressed with respect to some data we need to compute mutual informations with respect to input and output labels. LLMs are trained with inputs as preceding context and outputs as trailing context (for discussion of this, and examples of the labelling procedure see Appendix E.3). Maintaining conditional estimates of a token embedding given a preceding context $P(Z|X)$ for every possible context window proves intractable, and many contexts occur only once in the training data. Accordingly, like many other works on language modelling we approximate the distribution over possible sequences using n-grams with a kind of back-off (Katz, 1987). By conditioning on finite widths of preceding context we can tractably approximate $P(Z|X)$; the maximum width we consider here are quad-grams by which point $I(X;Z)$ begins to converge and past which point computation becomes intractable in an LLM setting (see Appendices E.1, E.2 for discussion). By backing off further (e.g. to trigrams, bigrams, and tokens) we can also estimate how much different context widths contribute to information in a model - in practice the majority of results presented here use trigram backoff which takes into account context while being considerably less sparse than quad-grams.

In addition to mutual information with input and output labels, we also consider human preference data. A growing body of work stresses the importance of post-training approaches for aligning models with human preference (Bai et al., 2022; Ouyang et al., 2022; Rafailov et al., 2023). We can quantify this information in a model using preference data, where a prompt has two continuations, one of which is labelled preferred by human raters. Conditioning on this label lets us compute $P(Z|\text{preferred})$ and $I(Z;\text{preferred})$.

Data and Sampling Getting a true estimate of the entropy of a vector space remains a major challenge, with most approaches underestimating the true entropy (Paninski, 2003). As a result we do not claim our experiments estimate the entropy of a model's true latent distribution, but rather an estimate of the entropy with respect to a particular sample of data. By holding the data constant across models and experiments we can compute an estimate that is useful for comparisons, even if it does not exactly match the true entropy. Unless otherwise noted, token bigram, trigram, and quadgram estimates are with respect to 10,000 samples from C4 (Raffel et al., 2020), and preference estimates are based on 10,000 samples from Tulu (Lambert et al., 2024); in both cases we consider a maximum context length of 512.

4 Experiments

In order to study training time-courses our pre-training analyses look at the OLMo2 family of models (OLMo et al., 2025), which makes available intermediate checkpoints [4]. We focus analysis on the 7B model unless otherwise noted, while including results for the 32B and 1B variants to show where conclusions hold or differ across model scales. In addition, to show our conclusions hold outside of this particular family of models we compare a wide array of open-weights LLMs (which do not make intermediate training checkpoints available), showing where they lie on the information plane at the end of training.

4.1 Pre-training Approaches Optimal Compression

The majority of pre-training appears to be a slow compression of a model's training data. The Information Bottleneck theory of deep learning predicts two phases: a *fitting phase* during which output information $I(Y;Z)$ increases, followed by a *compression phase* during which $I(Y;Z)$ remains constant and input information $I(X;Z)$ decreases. The theoretically predicted trajectory of training is visualised in Figure 1. Shown beside it (and reproduced in Figure 2) is the training trajectory for the OLMo2 7B model with respect to data from English C4. Strikingly, the 7B model closely follows the two-phase prediction from the Information Bottleneck, first increasing mutual information with outputs, before compressing input information and progressing towards the bound on optimal compression. This shows how, even at scale, deep-learning models appear to thread a

[4] Appendix D includes additional pre-training analyses of the Smol LM2 (Allal et al., 2025) and Pythia (Biderman et al., 2023) models which also make intermediate checkpoints available. These follow a similar pattern to the results presented here.

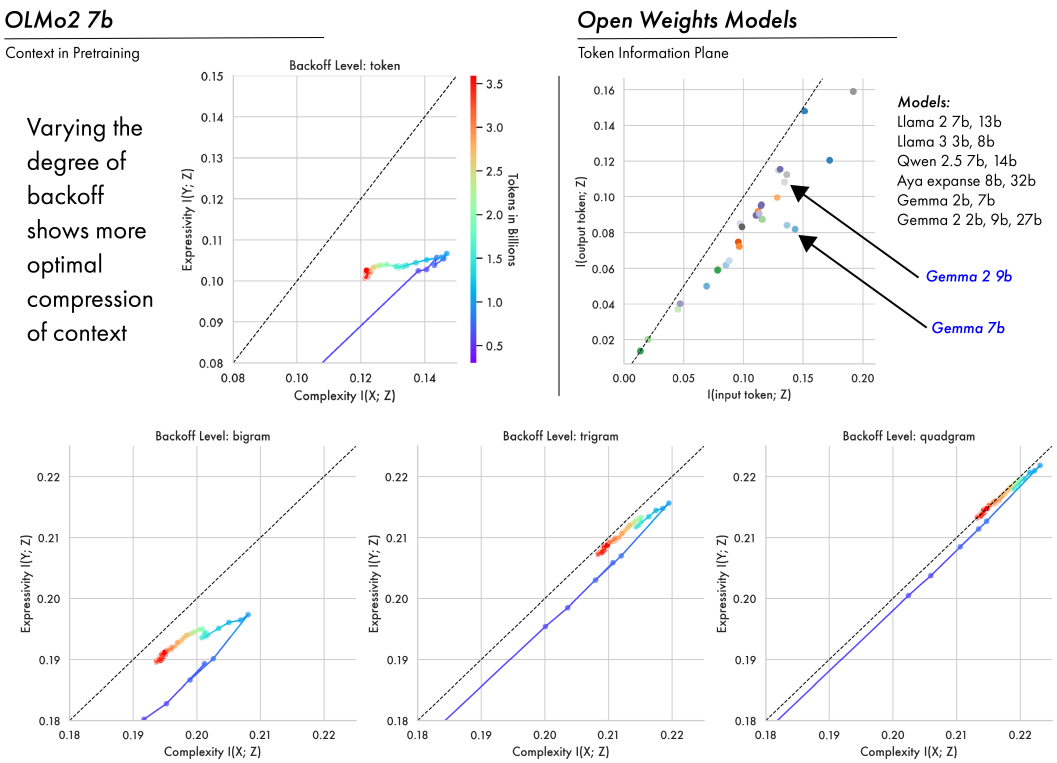


Figure 2: **Pre-Training Trajectories Match Theoretical Predictions**. (Top Left and Bottom Row) The information plane over pre-training for different levels of backoff. By changing how many tokens we condition the mutual information on in the context window, we see how the OLMo2 7B model compresses not just token but also local context information. Across all context windows we see the same two phase pattern predicted by the Information Bottleneck – with more contextual representations approaching greater optimality. (Top Right) The Information Plane with token backoff for a wide array of open-weights models. Models approach the bound for optimal compression, which is shown as a dotted line. Model names are shown at right with two points labelled. A full legend identifying each dot, with additional levels of backoff, is given in Appendix Figures 5 and 6.

needle between representational complexity and expressivity. It also demonstrates how LLMs can be effectively studied from the perspective of Rate Distortion Theory, as they try to converge to an optimal lossy compression of their training data. By varying the degree of backoff in the conditional distribution used to compute mutual information, we can see how contextual information evolves over pre-training at the token, bigram, trigram, and quad-gram levels (Figure 2 bottom). All cases result in a similar two-phase pattern of expansion and compression, with larger conditioning context converging closer to the bound. There is also a pattern of convergence such that quad-grams account for only marginally more information than trigrams – suggesting representations largely encode local context, likely reflecting the information locality of the natural language on which they're trained (Gibson, 1998; Gibson et al., 2000; Hahn et al., 2022). This high degree of optimality in contextual encodings also likely reflects an inherent pressure in the pre-training objective for models to not only develop token representations, but representations of a token *in context*.

The Effect of Scale: Smaller Models Struggle to Compress Parameter count shows a marked effect on the degree of compression achievable by a model. Figure 3, shows pre-training trajectories for the 1B, 7B, and 32B parameter models. The larger models both closely follow the hypothesized Information Bottleneck trajectory, exhibiting phases of expansion and compression, ultimately approaching optimal compression. The 1B parameter model exhibits markedly different behaviour. While it successfully completes the initial expansion phase – increasing output information $I(Y;Z)$ – it fails to approach optimal compression. Instead, in the second phase the smaller model oscillates while moving slowly away from the theoretical frontier (Figure 3 bottom-left). Correlations (Figure

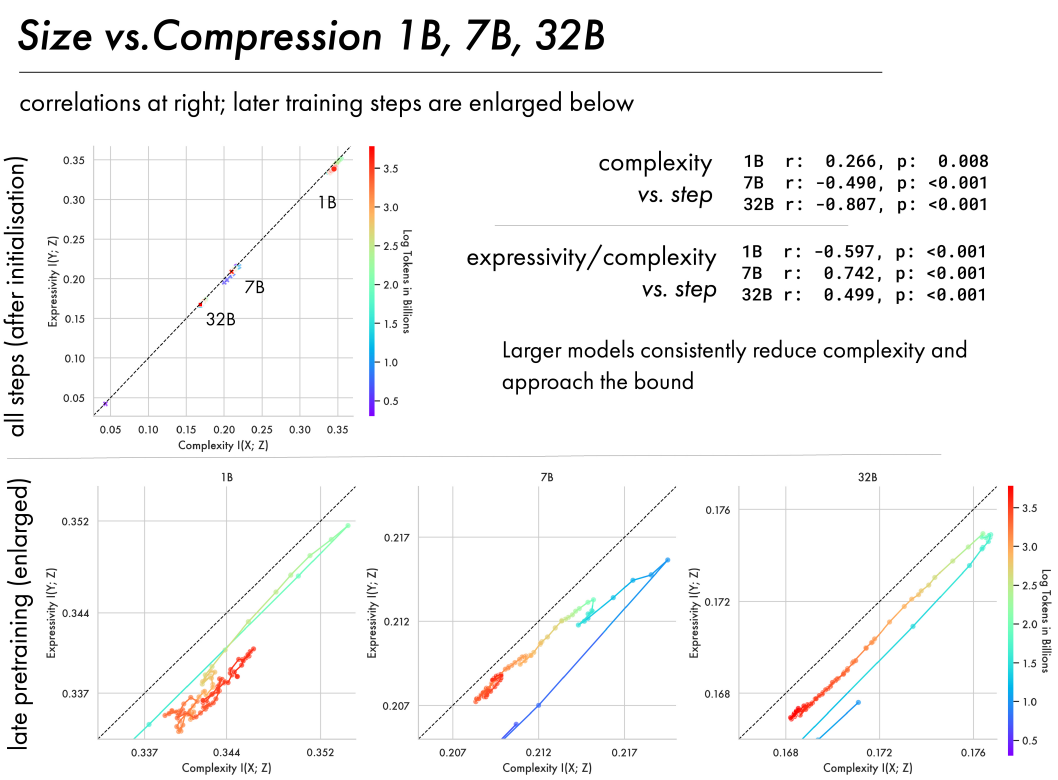


Figure 3: **Smaller Models Struggle to Compress**. (Top Left) Pre-training trajectory for three different model sizes with the 7B and 32B models converging to lower complexity solutions than 1B. (Bottom) Zooming in on later pretraining for each model size the 1B model matches Phase 1 but struggles to achieve meaningful compression later on, oscillating for much of pre-training off the frontier. (Top Right) Spearman correlations between training step and complexity show larger models compress over the course of training with the 32B compressing most. Correlations between step and the expressivity/complexity ratio show only larger models consistently approach the optimal bound (this ratio increases closer to the bound). All results use backoff to the trigram level.

3 top right) between training step and complexity show larger models compress representations, with the 1B model significantly expanding. Correlations between step and the ratio of expressivity over complexity – which increases as models approach the bound – show only larger models consistently approach the bound (as indicated by positive correlation coefficients). This suggests that for a given level of data complexity, a certain parameter threshold may be necessary for models to achieve an optimal compression – an observation in line with work on scaling laws (Kaplan et al., 2020).

Convergence Patterns Across Open-Weight Models In addition to looking at the OLMo2 family of models, we compute complexity and expressivity estimates across a diverse array of open-weight models (for tractability here we backoff to the token and bigram levels). A striking convergence pattern emerges: across different model families, hyperparameters, and training methodologies, representations ultimately converge to token and bigram informations clustered near the optimal bound on compression (Figure 2, *Right*; with full model names in Appendix, Figures 5 and 6). This suggests that training as a process of compression is not an artifact of a single LLM's training trajectory, but more fundamentally applies to to deep-learning models as a class, and to the data and the objectives used to train them.

4.2 Relating Representation Structure to Performance

So far we have studied how information in an LLM is structured; we now consider how that structure relates to downstream performance. Figure 4 shows correlations between representational measures

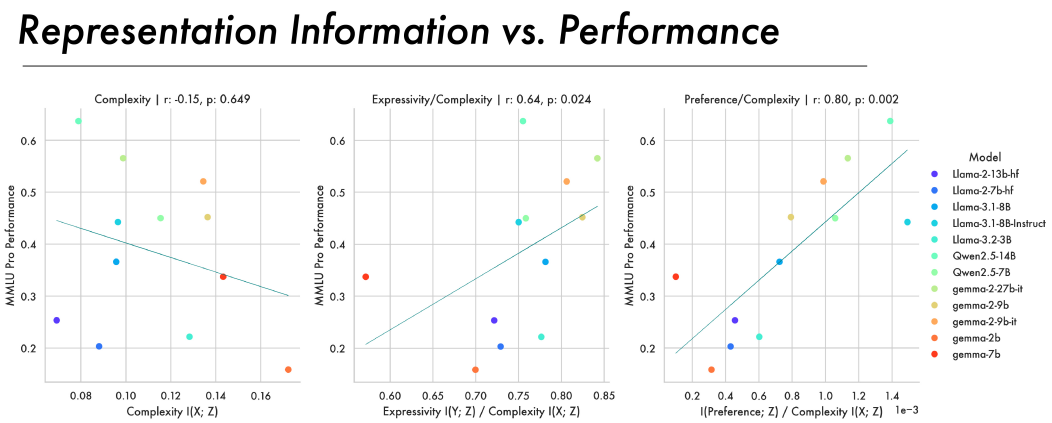


Figure 4: **Optimality of Compression and Preference Information Correlate with Performance** While complexity does not directly correlate with downstream performance as measured by accuracy on MMLU Pro (Left), the ratio of expressivity to complexity, which indicates distance from optimal compression, does (Middle). The ratio between preference information and model complexity also correlates with downstream performance (Right). Expressivity and complexity are calculated using token backoff in all three plots.

and performance on the MMLU Pro benchmark (Wang et al., 2024) for open weights models from 3 different families. Complexity alone proves not to be predictive of performance ($r = -0.15$, $p = 0.649$), however the ratio between expressivity and complexity is a significant predictor ($r = 0.64$, $p = 0.024$). This ratio indicates how close a model is to optimal compression, since it approaches 1.0 as the model approaches the IB bound. Together, these results indicate that compression alone is not a significant predictor of performance, but the *optimality* of that compression is.

While LLMs approach optimal compression for next sequence prediction over pre-training, a large body of work also tries to improve their ability to follow instructions, and generate responses humans prefer (e.g. Ouyang et al., 2022). We use preference data (Lambert et al., 2024) to compute mutual information with preference. As shown in Figure 4, the ratio between a model's complexity and the amount of preference information it contains also proves a significant predictor of downstream performance ($r = 0.8$, $p = 0.002$). This suggests that not only does the optimality of a model's compression matter, but exactly what information survives that compression does too. In Appendix C we include results showing that post-training increases the amount of preference information across different open weights models while minimally changing their complexity. This suggests that pre-training is responsible for the broad compression learned by a model, while post-training edits the information it contains; we leave a more complete assessment of how different phases of training affect representational compression to future work.

5 Conclusion

The work presented here bridges the gap between theoretical accounts of learning and the practical complexities of LLMs. We show that LLMs learn an optimal compression of the data on which they are trained, with a wide array of open-weights models converging near the IB bound – with the optimality of a model's compression predicting downstream performance. Each compression is different; we can account for the information that survives the compressive process, showing how representations encode information about different levels of local context and human preferences.

The approach to interpretability we introduce here interprets a model as a whole – rather than focussing on a particular circuit, or attention head – because complex distributed systems are not best understood in terms of their parts alone. Giving a holistic account of what it means to train an entire model on the entire internet is a challenge, but we argue that LLMs are best understood as lossy compression. In doing so, we place them in the context of a long history of work on representation learning across the sciences.

6 Ethics Statement

All experiments reported here use publicly available datasets and pretrained models obtained under their original licenses; see Appendix B.1 for details. To our knowledge, these datasets contain no personally identifiable information, and we are in compliance with their terms of use. No additional data were collected. More generally, all authors have read and adhered to the ICLR Code of Ethics. To the best of our knowledge, these results and their dissemination do not raise any ethical concerns.

7 Reproducibility Statement

All datasets and pretrained models used in our experiments are publicly available (see Appendix B.1). All code will be released with an open source license upon acceptance. Appendix B.2 contains details of compute resources necessary to reproduce these findings.

References

Allal, L. B., Lozhkov, A., Bakouch, E., Blázquez, G. M., Penedo, G., Tunstall, L., Marafioti, A., Kydlíček, H., Lajarín, A. P., Srivastav, V., Lochner, J., Fahlgren, C., Nguyen, X.-S., Fourrier, C., Burtenshaw, B., Larcher, H., Zhao, H., Zakka, C., Morlon, M., … Wolf, T. (2025). Smollm2: When smol goes big – data-centric training of a small language model.

Anderson, P. W. (1972). More is different: Broken symmetry and the nature of the hierarchical structure of science. *Science*, *177*(4047), 393–396.

Arimoto, S. (1972). An algorithm for computing the capacity of arbitrary discrete memoryless channels. *IEEE Transactions on Information Theory*, *18*(1), 14–20.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. (2023). Pythia: A suite for analyzing large language models across training and scaling. *International Conference on Machine Learning*, 2397–2430.

Blahut, R. (1972). Computation of channel capacity and rate-distortion functions. *IEEE transactions on Information Theory*, *18*(4), 460–473.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., et al. (2023). Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, *2*.

Burnham, K. P., & Anderson, D. R. (2002). *Model selection and multimodel inference: A practical information-theoretic approach*. Springer.

Chater, N. (1997). Simplicity and the mind. *The Psychologist*.

Chater, N., & Vitányi, P. (2003). Simplicity: A unifying principle in cognitive science? *Trends in Cognitive Sciences*, *7*(1), 19–22. https://doi.org/10.1016/S1364-6613(02)00005-0

Conklin, H. C. (2025). Information structure in mappings: An approach to learning, representation and generalisation. *The University of Edinburgh*.

Delétang, G., Ruoss, A., Duquenne, P.-A., Catt, E., Genewein, T., Mattern, C., Grau-Moya, J., Wenliang, L. K., Aitchison, M., Orseau, L., et al. (2023). Language modeling is compression. *arXiv preprint arXiv:2309.10668*.

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 4171–4186.

Edwards, A. W. F. (1972). Likelihood. Springer.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., et al. (2022). Toy models of superposition. *arXiv preprint arXiv:2209.10652*.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., et al. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*, *1*, 1.

Feldman, J. (2000). Minimization of boolean complexity in human concept learning. *Nature*, *407*(6804), 630–633.

Feldman, J. (2016). The simplicity principle in perception and cognition. *Wiley Interdisciplinary Reviews: Cognitive Science*, *7*(5), 330–340.

Frankle, J., & Carbin, M. (2018). The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*.

Futrell, R., Wilcox, E., Morita, T., & Levy, R. (2018). Rnns as psycholinguistic subjects: Syntactic state and grammatical dependency. *arXiv preprint arXiv:1809.01329*.

Futrell, R., Wilcox, E., Morita, T., Qian, P., Ballesteros, M., & Levy, R. (2019). Neural language models as psycholinguistic subjects: Representations of syntactic state. *arXiv preprint arXiv:1903.03260*.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. (2020). The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Gao, T., Yao, X., & Chen, D. (2021). Simcse: Simple contrastive learning of sentence embeddings. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6894–6910. https://doi.org/10.18653/v1/2021.emnlp-main.552

Ge, X., Shu, W., Wu, J., Zhou, Y., He, Z., & Qiu, X. (2025). Evolution of concepts in language model pre-training.

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, *4*(1), 1–58.

Gibson, E. (1998). Linguistic complexity: Locality of syntactic dependencies. *Cognition*, *68*(1), 1–76.

Gibson, E., et al. (2000). The dependency locality theory: A distance-based theory of linguistic complexity. *Image, language, brain*, *2000*, 95–126.

Goldfeld, Z., Berg, E. v. d., Greenewald, K., Melnyk, I., Nguyen, N., Kingsbury, B., & Polyanskiy, Y. (2019, May). Estimating Information Flow in Deep Neural Networks [arXiv:1810.05728 [cs, stat]]. Retrieved April 18, 2024, from http://arxiv.org/abs/1810.05728

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. (2024). The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Griffiths, T. L., Chater, N., & Tenenbaum, J. B. (2024). *Bayesian models of cognition: Reverse engineering the mind*. MIT Press.

Hahn, M., Futrell, R., Levy, R., & Gibson, E. (2022). A resource-rational model of human processing of recursive linguistic structure. *Proceedings of the National Academy of Sciences*, *119*(43), e2122602119.

Hu, J., Gauthier, J., Qian, P., Wilcox, E., & Levy, R. P. (2020). A Systematic Assessment of Syntactic Generalization in Neural Language Models [arXiv: 2005.03692]. *arXiv:2005.03692 [cs]*. Retrieved June 23, 2020, from http://arxiv.org/abs/2005.03692

Jayant, N., Johnston, J., & Safranek, R. (1993). Signal compression based on models of human perception. *Proceedings of the IEEE*, *81*(10), 1385–1422. https://doi.org/10.1109/5.241504

Jaynes, E. T. (1957). Information theory and statistical mechanics. *Physical Review*, *106*, 620–630. https://api.semanticscholar.org/CorpusID:17870175

Jeffreys, H. (1939). *The theory of probability*. OuP Oxford.

Jurafsky, D., & Martin, J. H. (2000). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition, 2nd edition. *Prentice Hall series in artificial intelligence*. https://api.semanticscholar.org/CorpusID:5073927

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Katz, S. (1987). Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE transactions on acoustics, speech, and signal processing*, *35*(3), 400–401.

Kirby, S., Tamariz, M., Cornish, H., & Smith, K. (2015). Compression and communication in the cultural evolution of linguistic structure. *Cognition*, *141*, 87–102.

Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., Gu, Y., Malik, S., Graf, V., Hwang, J. D., Yang, J., Bras, R. L., Tafjord, O., Wilhelm, C., Soldaini, L., … Hajishirzi, H. (2024). Tülu 3: Pushing frontiers in open language model post-training.

MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.

Marvin, R., & Linzen, T. (2018). Targeted Syntactic Evaluation of Language Models [arXiv: 1808.09031]. *arXiv:1808.09031 [cs]*, 1192–1202. https://doi.org/10.18653/v1/D18-1151

Mitchell, M. (2009). *Complexity: A guided tour*. Oxford University Press.

Nanda, N., Chan, L., Lieberum, T., Smith, J., & Steinhardt, J. (2023). Progress Measures for Grokking via Mechanistic Interpretability.

OLMo, T., Walsh, P., Soldaini, L., Groeneveld, D., Lo, K., Arora, S., Bhagia, A., Gu, Y., Huang, S., Jordan, M., Lambert, N., Schwenk, D., Tafjord, O., Anderson, T., Atkinson, D., Brahman, F., Clark, C., Dasigi, P., Dziri, N., … Hajishirzi, H. (2025, January). 2 OLMo 2 Furious [arXiv:2501.00656 [cs]]. https://doi.org/10.48550/arXiv.2501.00656

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P. F., Leike, J., & Lowe, R. (2022). Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, & A. Oh (Eds.), *Advances in neural information processing systems* (pp. 27730–27744, Vol. 35). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf

Oyama, M., Yokoi, S., & Shimodaira, H. (2023). Norm of word embedding encodes information gain. *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, *1*, 2108–2130.

Paninski, L. (2003). Estimation of Entropy and Mutual Information. *Neural Computation*, *15*(6), 1191–1253. https://doi.org/10.1162/089976603321780272

Pimentel, T., Valvoda, J., Maudslay, R. H., Zmigrod, R., Williams, A., & Cotterell, R. (2020). Information-theoretic probing for linguistic structure. *arXiv preprint arXiv:2004.03061*.

Poggio, T., Rifkin, R., Mukherjee, S., & Niyogi, P. (2004). General conditions for predictivity in learning theory. *Nature*, *428*(6981), 419–422.

Pothos, E. M., & Chater, N. (2001). 4 categorization by simplicity: A minimum description length approach to un supervised clustering.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., & Finn, C. (2023). Direct preference optimization: Your language model is secretly a reward model. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, & S. Levine (Eds.), *Advances in neural information processing systems* (pp. 53728–53741, Vol. 36). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2023/file/a85b405ed65c6477a4fe8302b5e06ce7-Paper-Conference.pdf

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, *21*(140), 1–67.

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP) and the 9th International Joint Conference on Natural Language Processing (IJCNLP)*, 3982–3992. https://doi.org/10.18653/v1/D19-1410

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

Sajjadi, M. S., Bachem, O., Lucic, M., Bousquet, O., & Gelly, S. (2018). Assessing generative models via precision and recall. *Advances in neural information processing systems*, *31*.

Saxe, A. M., Bansal, Y., Dapello, J., Advani, M., Kolchinsky, A., Tracey, B. D., & Cox, D. D. (2019). On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, *2019*(12), 124020. https://doi.org/10.1088/1742-5468/ab3985

Saxe, A. M., McClelland, J. L., & Ganguli, S. (2019). A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, *116*(23), 11537–11546.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, *27*(3), 379–423.

Shwartz-Ziv, R., & Tishby, N. (2017). Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*.

Tishby, N., Pereira, F. C., & Bialek, W. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Tishby, N., & Zaslavsky, N. (2015). Deep learning and the information bottleneck principle. *2015 ieee information theory workshop (itw)*, 1–5.

Veldhoen, S., Hupkes, D., & Zuidema, W. (2016). Diagnostic classifiers: Revealing how neural networks process hierarchical structure, 10.

Vitányi, P. M., & Li, M. (2000). Minimum description length induction, bayesianism, and kolmogorov complexity. *IEEE Transactions on information theory*, *46*(2), 446–464.

Voita, E., Sennrich, R., & Titov, I. (2019, September). The Bottom-up Evolution of Representations in the Transformer: A Study with Machine Translation and Language Modeling Objectives [arXiv:1909.01380 [cs]]. Retrieved March 26, 2024, from http://arxiv.org/abs/1909.01380

Voita, E., & Titov, I. (2020, March). Information-Theoretic Probing with Minimum Description Length [arXiv:2003.12298 [cs]]. Retrieved November 21, 2023, from http://arxiv.org/abs/2003.12298

Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification. *The Computer Journal*, *11*(2), 185–194.

Wang, Y., Ma, X., Zhang, G., Ni, Y., Chandra, A., Guo, S., Ren, W., Arulraj, A., He, X., Jiang, Z., et al. (2024). Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Warstadt, A., Parrish, A., Liu, H., Mohananey, A., Peng, W., Wang, S.-F., & Bowman, S. R. (2019). BLiMP: A Benchmark of Linguistic Minimal Pairs for English [arXiv: 1912.00582]. *arXiv:1912.00582 [cs]*. Retrieved January 13, 2020, from http://arxiv.org/abs/1912.00582

Wilcox, A. R. (1967). *Indices of qualitative variation.* (tech. rep.). Oak Ridge National Lab.(ORNL), Oak Ridge, TN (United States).

Zaslavsky, N., Kemp, C., Regier, T., & Tishby, N. (2018). Efficient compression in color naming and its evolution. *Proceedings of the National Academy of Sciences*, *115*(31), 7937–7942.

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). Bertscore: Evaluating text generation with bert. *Proceedings of the International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=SkeHuCVFDr

A Open-Weights Models, Detailed Visuals

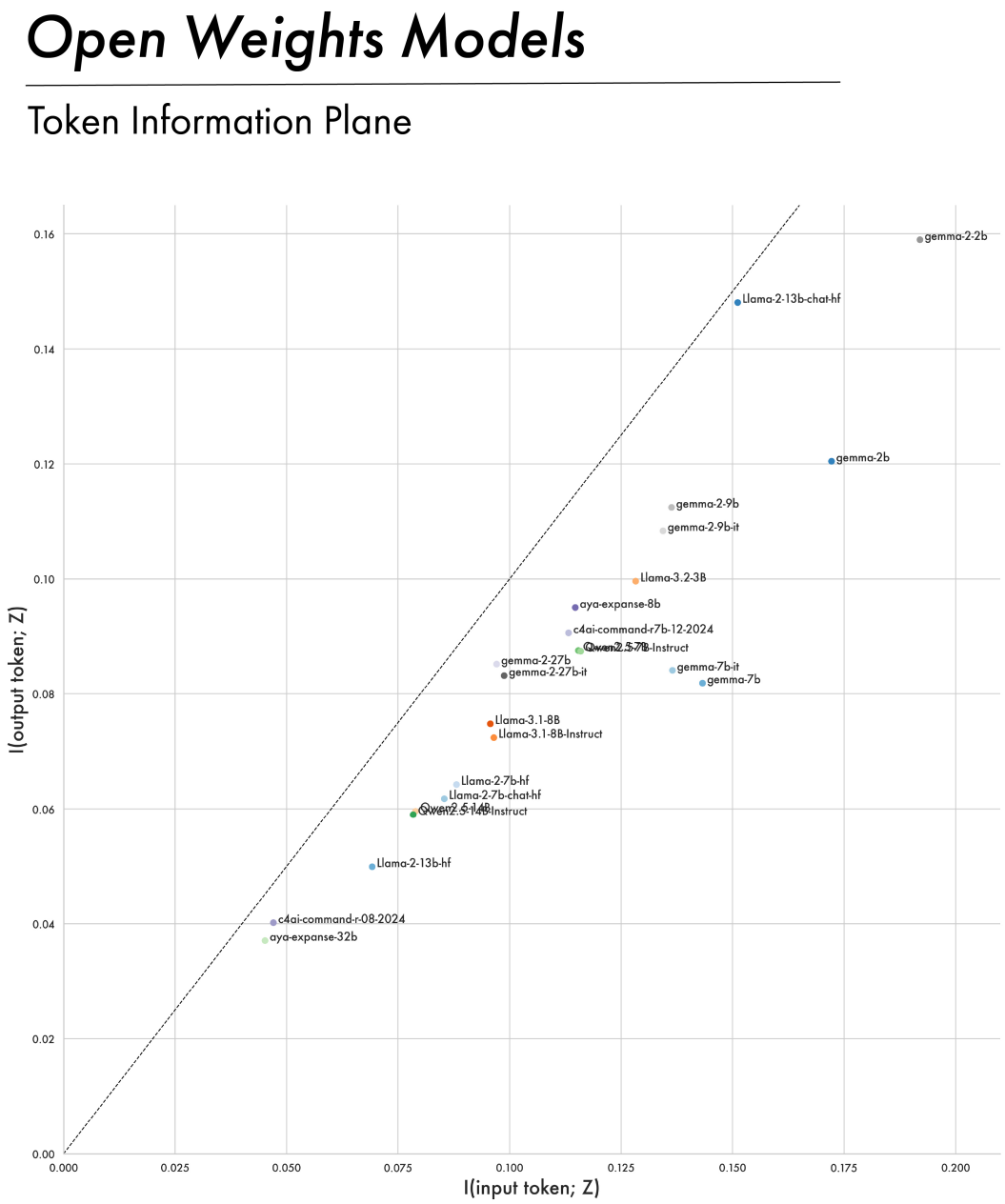


Figure 5: **The Token Information Plane for Open Weights Models** Shown here is the full, labelled token information plane for 23 different open-weights models. Overall while model lie at different levels of complexity and expressivity they broadly approach the IB Bound on optimal compression. The labels which are superimposed and so difficult to read are for the base and post-trained variants of Qwen2.5 (7B &14B). The bigram information plane is shown on the following page.

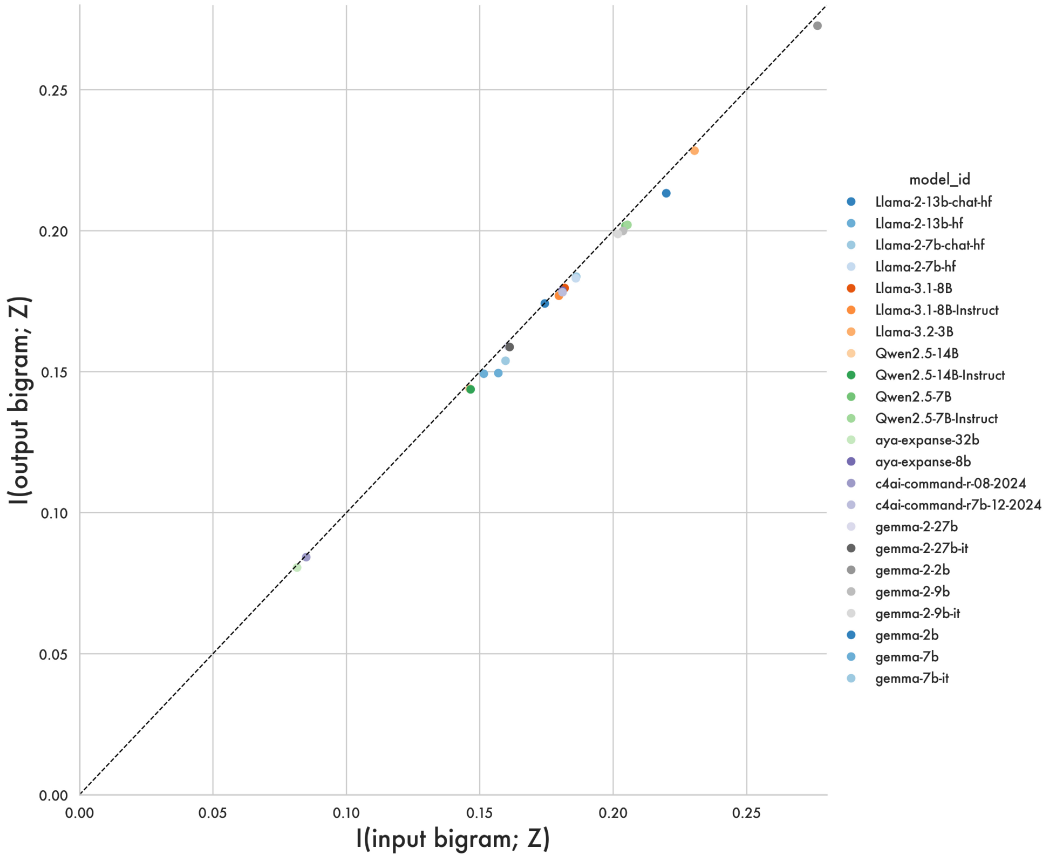


Figure 6: **The Bigram Information Plane for Open Weights Models** Shown here is the full, labelled bigram information plane for 23 different open-weights models. Compared with the token case above, here models lie even closer to the frontier. In part because many of the models are essentially superimposed a legend is provided at right, rather than labels adjacent to each point.

B Datasets, Models, and Compute

B.1 Licenses for Models and Datasets

As noted in section 3, we use two datasets for estimation - Tulu (Lambert et al., 2024) and C4 (Raffel et al., 2020) both of which fall under the Open Data Commons Attribution License (ODC-By) v1.0. Later we use MMLU Pro for behavioural evaluation (Wang et al., 2024) which falls under the Apache License (Version 2.0).

We study a wide array of models, below is license information grouped by model family:

- **OLMo:** The code and model are released under Apache 2.0.
- **Gemma:** Released under the gemma license stated here: https://ai.google.dev/gemma/terms
- **Llama:** Released under the llama license found here: https://www.llama.com/llama3/license/
- **Qwen:** The code and model are released under Apache 2.0.

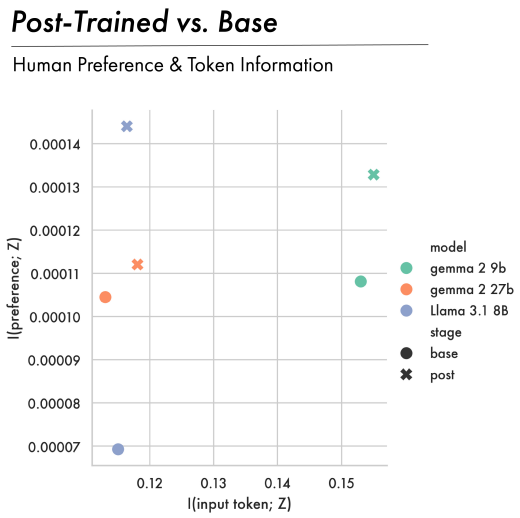


Figure 7: **Post-Training Increases Preference Information and Preference Information Correlates with MMLU Pro.** The vertical axis shows mutual information with human preference, before and after post-training (indicated by shape). The horizontal axis shows mutual information with input token.

- **Aya/Command:** Released under the Creative Commons Attribution Non Commercial 4.0
- **Pythia:** The code and model are released under Apache 2.0.

B.2 Compute Resource and Complexity

The estimation procedure used here has low complexity for an entropy estimator, requiring only a dot-product and softmax. The majority of compute expense comes from the model's forward pass required to compute the estimate. The complexity of this depends on the size of the model. In experiments here estimates required encoding 10,000 samples from C4 and Tulu. This process takes approximately 10, 40, or 70 minutes on either 2, 4, or 8 H100 GPUs respectively (number required depending on model size). Given this we estimate the total number of GPU hours required for the results in this paper at approximately 3,600 H100 hours.

C Post-Training Increases Preference Information

While LLMs become optimally compressed for next sequence prediction over pre-training, the final phase of the training pipeline often introduces other kinds of information. In the general case, post-training is designed to improve a model's ability to follow instructions and better align it with human preferences; we look at how this changes the information content of a model, and how it affects the representations from pre-training. Figure 7 (Left) shows preference information across two different families of open weights models, Llama 3 and Gemma 2, which release a checkpoint at the end of pre-training and one at the end of post-training. Post-trained models show higher degrees of preference information than their pre- and mid-trained counterparts, with minimal change to token information. This supports a framing of pre-training as imbuing the model with core semantic information, which is later augmented with task-specific and preference information.

D Additional Model Timecourses

A major challenge in studying pre-training is the limited availability of checkpoints. While there are a huge number of checkpoints available for final trained models, intermediate checkpoints over the course of pre-training are relatively rare. We focus analysis in the main paper on the OLMo2 models as they offer comprehensive check pointing – and comparatively strong performance. Here

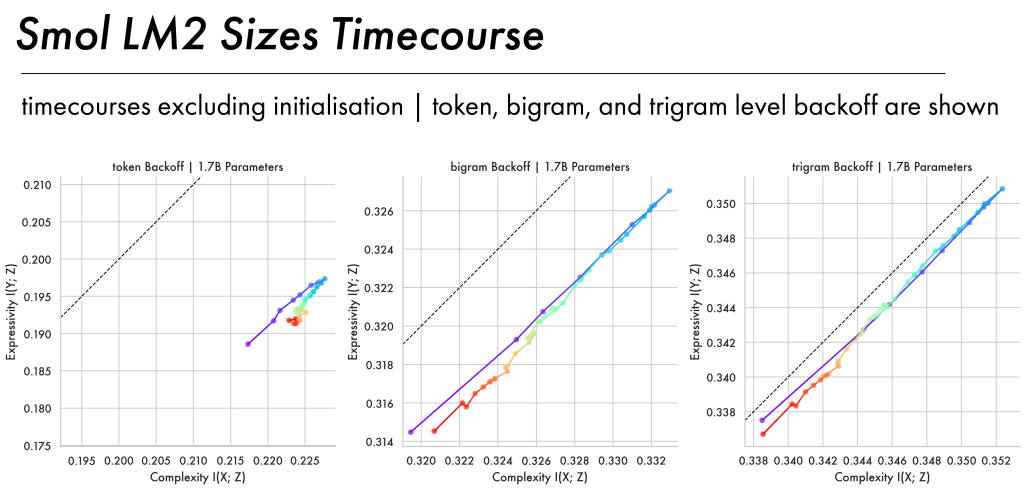


Figure 8: **Smol LM2 Timecourses**

we look at two other families of models which make available some pre-training checkpoints. The Smol LM2 models (Allal et al., 2025) released this year are models with 1.7B parameters or smaller that achieve competitive performance. The 1.7B Smol model was trained on 11 Trillion tokens and performs comparably to the 1B OLMo2 model which was trained on 4 Trillion Tokens. Broadly the 1.7B Smol model follows a similar training trajectory to the OLMo2 1B model having phases of expansion and compression but failing to approach the bound like the OLMo2 7B and 32B models. Pretraining timecourses for the Smol 1.7B model are shown in Figure 8 with token, bigram, and trigram backoff.

The other family of models we analyse are the Pythia models (Biderman et al., 2023). Timecourses for two Pythia models are shown in Figure 9. Included are analyses of the 1.4B and 6.9B models. In terms of parametrisation these are roughly comparable to the 1B and 7B OLMo2 models analysed in the main paper. However it's worth noting that the methodology for training these models is substantially different, and that their performance is substantially lower than the OLMo2 models, and other more recent open-weights models analysed above. In terms of training, Pythia models are intended for scientific analysis, as a result they use the same amount of data, batch size, and number of training steps across model sizes. Perhaps most importantly these models are trained on the Pile dataset (L. Gao et al., 2020). This contains roughly 299,892,736,000, by contrast the 1B OLMo2 model is trained on 4,000,000,000,000 tokens - meaning the Pythia models see 7.5% of that data. Accordingly the 1.4B Pythia model appears to achieve better compression later in training than its OLMo counterpart. As discussed in the main paper there may be a relationship between data complexity and the model complexity needed in order to achieve substantive compression of it. By contrast the 6.9B Pythia model is still compressing representations late into pretraining; this would appear to indicate it is under-trained.

E Entropy Estimation

E.1 Approximating the Input Distribution

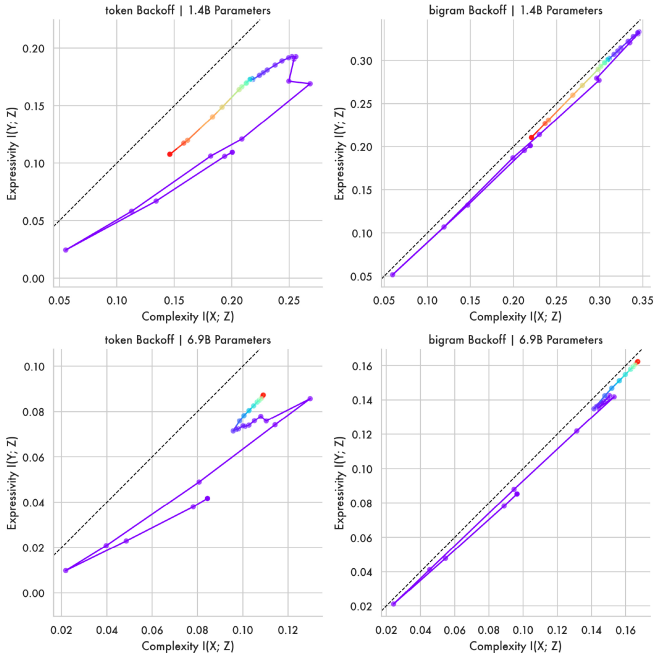
We estimate the mutual information between model inputs and outputs. In an auto-regressive decoder-only LLM the input to a model is the preceding context up to the current token. We view the input as n-grams of tokens where the input at timestep x_t is an ngram of width t containing all tokens $x_0 ... x_t$. Maintaining probability distributions for every possible context proves intractable due to the combinatorial complexity of natural language. Additionally, ngrams greater than 3 tokens become sparsely distributed in the data making reliable estimation of their probabilities a challenge. As a result we condition estimates on ngrams of fixed-widths 1, 2, 3, 4 - referred to in the paper as token, bigram, trigram, quadgram. This is related to Backoff (Katz, 1987) which reduces n-gram size until the n-gram has non-zero probability in a corpus. Here though we do not interpolate dif-

Pythia Sizes Timecourse

timecourses excluding initialisation | token and bigram level backoff are shown

Pythia models are trained on orders of magnitude fewer tokens than the OLMo2 Models shown in the main paper - only 7% of the OLMo2 data. Additionally unlike other LLM families all sizes see the same amount of data, in the same batch size, for the same number of steps.

Accordingly here, the Pythia 6.9B model appears undertrained, still expanding representations even late in training. Having seen the same amount of data the smaller model follows the expected two phase trajectory. Here the 1.4B pythia model may achieve better compression given it is trained on only a small fraction of the data OLMo2 1B is expected to compress.



Computing a ratio of output information over input information gives us a heuristic for describing how far a model is from the bound. As this approaches 1.0 a model approaches the bound. Comparing these ratios for the pythia and OLMo2 models shows pythia ends pretraining further from optimality.

	OLMo2		Pythia	
token	1B	0.793	1.4B	0.740
	7B	0.842	6.9B	0.801
	32B	0.817		
bigram	1B	0.960	1.4B	0.953
	7B	0.981	6.9B	0.971
	32B	0.976		
trigram	1B	0.981	1.4B	0.979
	7B	0.995	6.9B	0.992
	32B	0.993		

Figure 9: **Pythia Model Timecourses**

ferent n-gram widths, instead maintaining separate aggregate estimates for each width – in part to be able to study how different levels of contextual information are represented in the model. Where a given n-gram, like a quadgram, does not have non-zero probability in the data it is omitted from overall quadgram mutual information estimate.

In practice this means estimates for smaller n-gram widths are more reliable - a classical issue in language modelling (see Jurafsky & Martin, 2000, p.32). Token, bigram, and trigram estimates can be estimated reliably from a relatively small sample of data. We judge this by looking at how estimates change as a function of the number of samples during the estimation procedure, by 5,000 samples these estimates relaibly begin to converge. Quadgrams, due to their sparsity, tend to have less robust estimates - additionally the number of labels grows quadratically with each additional ngram width making quadgrams challenging to estimate for larger models with larger vocabularies. As a result our broad comparison of open-weights models uses token and bigram estimates. Additionally the pre-training model size analysis here focusses on trigram estimates (figure 3) as this widest context

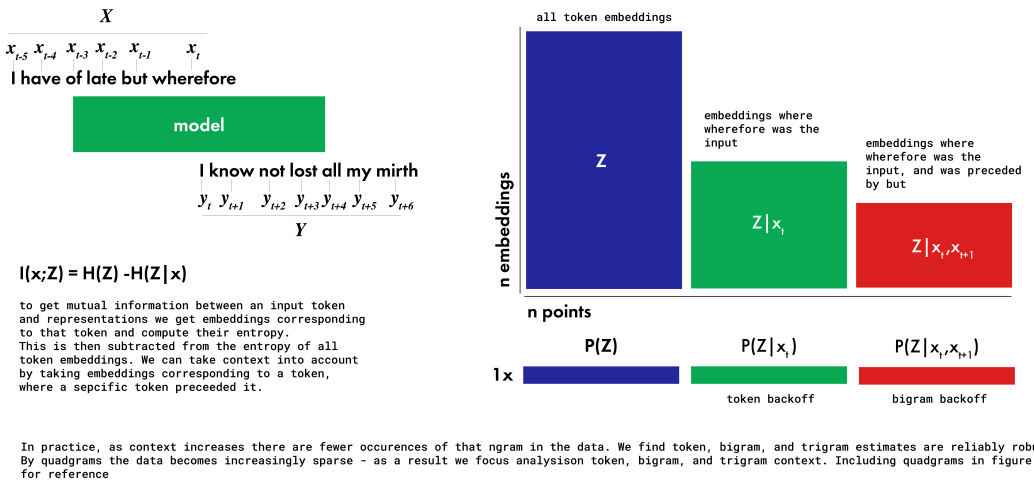


Figure 10: **Illustration of conditional probability estimates.** An example sentence is provided, assuming word-level tokenization for simplicity. At left are the indices for the input and output tokens when the current input word is *wherefore*. At right is shown the sub-setting procedure for estimating conditional probabilities. This illustrates how bigram estimates do not compute entropy of two token embeddings, rather the embedding for the current token embedding conditioned on preceding context.

that still reflects a reliable estimate. The analysis of how context is represented over pretraining (figure 2) includes quadgram estimates for reference.

E.2 Approximating the Output Distribution

While during inference models predict the next token given preceding context, this is distinct from how they are trained. During training of an auto-regressive decoder-only LLM causal masking means a token can only attend to preceding context, not trailing context. However transformer decoders are trained using teacher forcing, where predictions are generated for the entire sequence in parallel by assuming predictions are made correctly. This is instead of having training operate on one token at a time with a separate forward pass for each - which is how predictions are generated during inference. The result of this is that for an embedding e_t at timestep t, following embeddings e_{t+1} can attend to e_t. This means embeddings get gradient information from the trailing context. Put another way, the prediction for output y_{t+1} is written in terms of e_t. As a result the gradient information from the next token(s) in a sequence $\nabla\mathcal{L}_\theta(y_{t+1})$ affect the embedding at the current timestep.

Given that our analysis computes embedding mutual informations over training with respect to a model's input and outputs this fact has implications for us. It means that the output for e_t is not just y_{t+1} but all following output tokens $y_{t+1}...y_n$ where n is the sequence length. This is because e_t receives gradient information from the loss with respect to predicting all following tokens in a sequence. As a result we consider X to be the entire preceding context in the input (as mentioned above), and Y to be the entire trailing context after the current point in the sequence. This means when we compute mutual informations for different n-gram widths we match the width for X and Y - conditioning the estimates on the same width of preceding and trailing context respectively.

E.3 Estimating Mutual Informations

To compute mutual informations between the input X and representations Z, we need two quantities: the entropy of representations $\mathcal{H}(Z)$ and the conditional entropy given the input $\mathcal{H}(Z|X)$. To compute $\mathcal{H}(Z)$ we use the quantisation procure described in section 3.1 applied to all token embeddings which gives $\hat{Z}$ - by summing over the each embedding and renormalising we get a categorical distribution $P(Z)$ that describes the embedding space. To get a conditional estimate $P(Z|X)$ we

simply take $\hat{Z}$ and compute a subset, containing the embeddings corresponding to the input X, $\hat{Z}|X$. Summing and renormalising gives us the distribution $P(\hat{Z}|X)$.

This brings us to an important distinction, our analysis discusses mutual informations with respect to tokens, bigrams, trigrams, and quadgrams. These are not computed over different widths of embeddings, but rather over single token embeddings conditioned on the preceding context. In the same way $P(\hat{Z}|X)$ is computed as a subset of $P(Z)$, as we condition on further context we can subset the embeddings further. Figure 10 gives a high-level illustration of this process. It means that $Z|$token is a subset of Z, and $Z|$bigram is a subset of $Z|$token, $Z|$trigram is a subset of $Z|$bigram etc. This means the terms token, or bigram mutual informations refer to the width of the conditioning context, not the width of the embeddings over which entropy is computed.

E.4 On The Use of Shannon Entropy

In this paper we compute the entropy of continuous latent variables. As a result it is natural to ask why we - in line with previous work (Sajjadi et al., 2018; Shwartz-Ziv & Tishby, 2017; Voita et al., 2019) - opt instead to discretise representations and compute their Shannon entropy (Shannon, 1948). There are two major reasons for this; first, differential entropy is not the true continuous analogue of Shannon Entropy (Jaynes, 1957). This is shown by the fact that differential entropy $D(X)$is unbounded $-\infty \leq D(X) \leq \infty$, and variant under linear transformations. This is the main motivator for an information theoretic analysis to discretise and use Shannon entropy directly. A secondary consideration is that we don't know how embeddings are distributed, so in order to get a differential entropy estimate we would first need to fit a distribution to the data. At scale this fitting step can be expensive, and introduce topographic assumptions. While discretisation is imperfect it enables the use of Shannon entropy, and makes minimal topographic assumptions.

E.5 Scalability of Prior Work

Notably Shwartz-Ziv and Tishby (2017) perform an empirical information theoretic analysis of neural-networks trained on MNIST. To do so they perform dimension-wise discretisation of model embeddings. This turns a 16-dimensional vector into a 16 character string. They then convert this to a categorical distribution over all possible strings. This gives a single categorical distribution that can describe representations at a particular layer in a network. This approach to discretisation works well on small problems - they study feed-forward networks trained on MNIST. However the dimension-wise discretisation requires taking a hidden representation with dimensions *batch* $\times$ *hidden* and transforming it to *batch* $\times$ *hidden* $\times$ *n bins*. If using 50 bins, in practice this means using 50 times the memory of not discretising. For the OLMo2 32B model used in this paper which has a hidden dimension of 5120 and 64 layers, and where we have a context window of 512 tokens, this would require holding in memory a tensor of dimensions *batch* $\times 512 \times 5120 \times 44 \times$ *n bins*. The memory use of this approach makes it intractable to apply to contemporary models and the problems studied here.

Voita et al. (2019) studied the transformer base model which has only 6 layers with a hidden dimension of 512. Despite this they note the approach from Shwartz-Ziv and Tishby (2017) was not tractable to apply to the model. They opt instead for quantising representations via clustering, based on related work from (Sajjadi et al., 2018). This method runs a clustering algorithm (Voita et al. (2019) use mini-batch k-means), then treats each cluster as an event in a categorical distribution - where density is assigned proportional to cluster membership. While this method provides robust entropy estimates, and dramatically less memory usage than the approach from Shwartz-Ziv and Tishby (2017) it still has relatively high computational complexity. It requires running a clustering algorithm to convergence, before performing quantisation prohibiting its use in an online setting - you need the cluster centroids before you can assign embeddings to them. Again thinking of the OLMo2 32B model used here, this would require running a clustering algorithm on 5120 dimensional spaces, at all 44 layers separately, for each of the 150 pre-training checkpoints. This would provide the 'bins' for the quantisation, then embeddings would need to be assigned to bins, requiring a second forward pass.

In practice an information-theoretic analysis of an LLM requires an entropy estimation method that is memory efficient, fast to compute, and can be applied in an online setting - requiring a single forward pass and no caching of the embeddings. The only estimator we're aware of that meets these

criteria is the soft-entropy estimator (Conklin, 2025). Here the quantisation requires only a cosine-similarity and a softmax making it fast and memory efficient. Additionally the normalisation step means 'bins' can be computed once at the start of the analysis, rather than needing a pass through the data to fit clusters or fit the support of the model's distribution. Conklin (2025) notes that the use of cosine similarities means this method considers only angular information in the representation space. While euclidean distances can be used instead, this would require first estimating the support of the distribution to fit the 'bins' making online estimation challenging. However use of cosine-based methods is standard practice in NLP (T. Gao et al., 2021; Reimers & Gurevych, 2019; Zhang et al., 2020), with some work suggesting vector norms in LLMs predominantly encode frequency information (e.g. Oyama et al., 2023).

E.6 The Information Bottleneck Bound

The Information Bottleneck bound is the curve traced by varying the trade-off parameter β in:

$$\mathcal{F}_\beta[p(Z|X)] = I(X;Z) - \beta I(Y;Z) \tag{7}$$

The curve this traces is where representations are optimally compressed. Along this bound $p(Z|X)$ is an optimal encoder, from inputs to representations, preserving only the information in X relevant to Y. For a given dataset this optimal encoder can be found numerically via a version of the Blaut-Arimoto (Arimoto, 1972; Blahut, 1972) method for computing channel capacity. Introduced in Tishby et al. (2000), the information bottleneck method for determining channel capacity relies on three equations:

$$p_\beta(z|x) = \frac{p_\beta(z)}{Z_\beta(x)} \exp\Big(-\beta D[p(y|x)||p_\beta(y|Z)]\Big) \tag{8}$$

$$p_\beta(z) = \sum_{x\in X} p(x) p_\beta(z|x) \tag{9}$$

$$p_\beta(y|z) = \sum_{x\in X} p_\beta(x|z) p(y|x) \tag{10}$$

These equations are satisfied self-consistently at the bound. As these three equations rely on each other one can learn an optimal encoder by starting with a randomly initialised one. Then iteratively computing each of these equations in turn.

In the general case the shape of this bound follows a linear relationship, until all mutual information between x and y is captured. At this point the curve saturates — additional complexity doesn't result in additional accuracy, as there's no more predictive information in x. This means numerical computational bound is largely important for computing where it saturates.

However numerical computation of the bound in our setting proves intractable. Here the optimal encoder $p(z|x)$ needs to map all of natural language to representations that optimally predict the next token. This is an exceedingly challenging problem for an iterative numerical optimizer – it's a problem that ordinarily requires a large language model. In experiments we are able to compute a bound for tokenizers up to 50,000 tokens, however past this point convergence begins to fail. In our setting this process takes a tokenizer and 300,000 sentences from c4 to get a maximum likelihood estimate of $P(x), P(y), P(y|x)$ on data representative of a model's training data. We can then iteratively compute $P(z|x)$ until convergence. In our experiments this iterative procedure converges to the expected saturation point - where $I(Z;Y) = I(X;Y)$.

Given that we would like to have a bound for problems where numerical computation of it proves intractable, we leverage this pattern by assuming the bound follows a linear relationship until the saturation point where $I(Z;Y) = I(X;Y)$. The largest tokenizer for which we can tractably compute this quantity has a normalised $I(X;Y)$ of 0.7 (where 1.0 is the maximum possible value). Across all open weights models the highest token complexity converged to is 0.15, well below the saturation point. This is in line with results from (Shwartz-Ziv & Tishby, 2017), which shows FFNs on MNIST only converge near the saturation point when over-fitting.