Transformers are Universal Predictors

Sourya Basu [1 2] **Moulik Choraria** [1 2] **Lav R. Varshney** [1 2]

Abstract

We find limits to the Transformer architecture for language modeling and show it has a universal prediction property in an information-theoretic sense. We further analyze performance in non-asymptotic data regimes to understand the role of various components of the Transformer architecture, especially in the context of data-efficient training. We validate our theoretical analysis with experiments on both synthetic and real datasets.

1. Introduction

Language models that aim to predict the next token or word to continue/complete a prompt have their origins in the work of Shannon (1948; 1950; 1951). In recent years, neural language models have taken the world by storm, especially the Transformer architecture (Vaswani et al., 2017). Following work on other neural network architectures (Cybenko, 1989), one can show that the Transformer architecture has a *universal approximation* property for sequence-to-sequence functions (Yun et al., 2020).

Transformer architectures have excellent performance and parallelization capability on natural language processing (NLP) tasks, becoming central to several state-of-the-art models including GPT-4 (OpenAI, 2023) and PaLM 2 (Anil et al., 2023). Such large language models (LLMs) are not only very good at the statistical problem of predicting the next token, but also have emergent capabilities in tasks that seemingly require higher-level semantic ability (Wei et al., 2022). Moreover, transformer-based architectures have achieved tremendous attention on domains beyond NLP, such as images (Dosovitskiy et al., 2020), audio (Li et al., 2019), reinforcement learning (Chen et al., 2021), and even multi-modal tasks (Jaegle et al., 2021). Transformers also show cross-domain transfer learning capabilities, i.e.,

models trained on NLP tasks show good performance when fine-tuned for non-NLP tasks such as image processing. In this sense, the Transformer architecture is said to have a *universal computation* property (Lu et al., 2021), reminiscent of predictive coding hypotheses of the brain that posit one basic operation in neurobiological information processing (Golkar et al., 2022).

The basic predictive workings of Transformers and previous findings of universal approximation and computation properties motivate us to ask whether they also have a *universal prediction* property in the information-theoretic sense (Feder et al., 1992; Weissman & Merhav, 2001), which itself is well-known to be intimately related to universal data compression (Merhav & Feder, 1998). As far as we know, the predictive capability of Transformers has not been studied in an information-theoretic sense, cf. Gurevych et al. (2022).

We investigate not only the underlying mathematical principles that govern the performance of Transformers, but also aim to find limitations to their learning capabilities. We show that Transformers are indeed universal predictors, i.e. they can achieve information-theoretic limits asymptotically in the amount of data available. We also analyze their performance in the finite-data regime by understanding the role of various components of the Transformer architecture, providing theoretical explanations wherever applicable.

To summarize our main results, we find the limits to performance of Transformers and show they are optimal predictors. Our limits only assume the Markov nature of data and are otherwise universal. Moreover, we analyze the role of the major components of a Transformer and provide better understanding and directions for their data-efficient training. Finally, we validate our theoretical analysis by performing experiments on both synthetic and real datasets.

2. Definitions and Preliminaries

2.1. Finite-State Markov Processes (FSMPs)

Let $x = \{x_1, \ldots x_n\} \in \mathcal{X}^n$ be sequential data, where $\mathcal{X}$ is some finite set. The state sequence of an FSMP, $s = \{s_1, \ldots, s_{n-1}\}$, is generated recursively according to $s_i = g(x_i, \ldots, x_{(i-k+1)_+})$, where $g(\cdot)$ is the state function for this FSMP, $s_i \in \mathcal{S}$, and $(i)_+ = \max\{i, 1\}$. An FSMP has a predictor function $f(\cdot)$ that outputs a probability distribution

over possible values for x_{i+1}, i.e. $\hat{x}_{i+1} = f(s_i) \in \mathbb{R}^{|\mathcal{X}|}$, $\sum_j f(s_i)_j = 1$. Hence, a FSMP is given by the pair (g, f).

2.2. Transformers

A Transformer architecture consist of three components placed in a series: an input embedding layer, multiple attention layers, and an output projection matrix. Let us describe each of the subcomponents of a Transformer.

Input embedding layer Let the input be sequential data $x = \{x_1, \ldots x_{n-1}\} \in \mathcal{X}^{n-1}$. The embedding layer E processes the input sequence individually to give a sequence $z = \{z_1, \ldots z_{n-1}\} \in \mathbb{R}^{(n-1) \times d_{in}}$.

Attention layer The attention layer further consists of two subcomponents: the self attention mechanism and the position-wise feedforward network.

The masked self-attention layer takes input $X = [x_1, x_2, \ldots, x_{n-1}]^T \in \mathbb{R}^{(n-1) \times d_{in}}$. Queries (Q), keys (K), and values (V) are computed from X by multiplying with three corresponding matrices as $Q = XW_Q, K = XW_K$, and $V = XW_V$, where each matrix W_Q, W_K, and W_V is of dimension $d_{in} \times d_{model}$. The output of the self-attention layer, $H^{(0)} = [h_1^{(0)}, h_2^{(0)}, \ldots, h_{n-1}^{(0)}]$ is given by

$$H^{(0)}$$
$$= \text{Attention}(Q, K, V) = \text{softmax}(\text{mask}_{-\infty}(\frac{QK^T}{\sqrt{d_{model}}}))V,$$

where $\text{mask}_{-\infty}$ is the causal binary mask used to preserve the auto-regressive property of language modeling by ensuring $h_i^{(0)}$ depends only on $x_{j \leq i}$ by setting all the entries in the matrix QK^T corresponding to connections to $x_{j>i}$ to $-\infty$.

This attention mechanism is extended to multi-head attention with m heads by simply dividing the inputs of dimension d_{model} into m sub-parts and computing attention separately and then concatenating them.

Masked self-attention followed by position-wise feedforward network, FFN, gives a Transformer decoder layer $H^{(1)} = [h_1^{(1)}, h_2^{(1)}, \ldots, h_{n-1}^{(1)}]$.

Output projection matrix The output projection matrix W_p takes as input a sequence of dimension $\mathbb{R}^{(n-1) \times d_{model}}$ and outputs probabilities on the output space of dimension $\mathbb{R}^{(n-1) \times |\mathcal{V}|}$, where $\mathcal{V}$ is the vocabulary space.

An L-layered Transformer decoder consists of an embedding layer, followed by L attention layers, followed by an output projection matrix. A single-layered Transformer decoder is shown in Fig. 1.
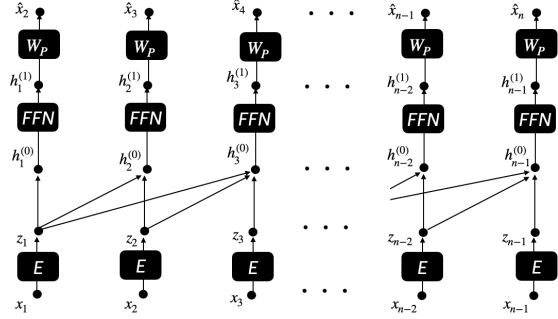

Figure 1. Single-layered Transformer decoder with attention span, $k, = 3$.

3. Performance Limits of Transformers

Here we provide theoretical limits to the performance of the Transformer architecture. First, we show that the Transformer architecture can be viewed as an approximate FSMP.

3.1. Transformers as approximate FSMPs

An FSMP as defined in Sec. 2.1 is given by a function pair (g, f), where g is a state function that first *aggregates* certain past observations $(x_j, \ldots, x_{i-1})$, where $j < (i-1)$ is a choice for the g function and f is a probability function from the states given by g to $\mathbb{R}^{|\mathcal{V}|}$.

In an L-layered Transformer, we model the g function by the embedding layer E followed by a sequence of L attention layers. The lth attention layer computes the weighted sum of past observations $(h_j^{(2l)}, \ldots, h_{i-1}^{(2l)})$ followed by an FFN. Note that the weighted sum in the attention mechanism is performed in the higher dimension d_{model}, which can retain as much information as concatenation in lower dimension if d_{model} is large enough. The output of the embedding layer followed by L-attention layers can be seen as approximating the g function, call it $\bar{g}$.

This is followed by the output projection matrix W_P, which can be seen as approximating the output probability function f, call it $\bar{f}$. Fig. 2 shows a single-layer Transformer, comparing its components with that of an FSMP.

3.2. Theoretical Limits

Here, we use the similarity between Transformers and FSMPs to find the limits of the Transformer architecture. First we provide the training setup and loss criterion for our results. Note that the only assumption we use in the following data generation process is of Markovity and hence, the results obtained are general otherwise. We first describe the dataset and the loss criterion.
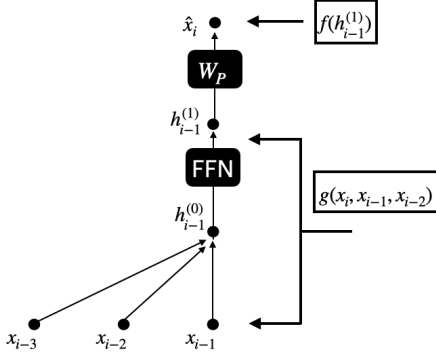
2

Figure 2. A single Transformer decoder layer with attention span equals to three ($k = 3$) and its comparison to a finite-state Markov predictor (FSMP). The output of the Transformer decoder, h_{i-1}^1 can be interpreted as the output of the state function, $g(x_i, x_{i-1}, \ldots, x_{i-k+1})$, of a FSMP. The matrix W_P can be interpreted as the predictor function $f(\cdot)$ of a FSMP that takes the state as input and outputs a probability distribution over the vocabulary space. The embedding layer is ignored for brevity.

Dataset We consider the train dataset $\mathcal{D}_{Train} = \{x_1, \ldots, x_n; y_1, \ldots, y_n\}$ of size n and test dataset $\mathcal{D}_{Test} = \{x_{n+1}, \ldots, x_{n+m}; y_{n+1}, \ldots, y_{n+m}\}$ of size m. In the datasets, $\{x_i\}_{i=1}^{n+m}$ is an arbitrary sequence with distribution $p(x_1, \ldots, x_{n+m})$ and y_i is generated conditioned on $g(x_i, \ldots, x_{(i-l+1)_+})$, where g is a state function. Thus, we consider a sequence of Markov order l. For our theoretical analysis, we use binary vocabulary, i.e. $x_i, y_i \in \{0, 1\}$, and will let $m \to \infty$.

Loss criterion A model is trained on $\mathcal{D}_{Train}$ with some state function $\bar{g}$ to obtain an order k stationary Markov estimator $p_{\theta_k}^n(\cdot)$, which is the f function in the context of an FSMP.

The train loss on $\mathcal{D}_{Train}$ is given by $\mathcal{L}_{Train}^{(l,k,n)}(\mathcal{D}_{Train}) = -\frac{1}{n}\sum_{i=1}^{n} \log\left(p_{\theta_k}^n(y_i|\bar{g}(x_{\leq i}))\right)$.

Denote the loss on $\mathcal{D}_{Test}$ by $\mathcal{L}_{Test}^{(l,k,m,n)}(\mathcal{D}_{Test}) = -\frac{1}{m}\sum_{i=1}^{m} \log\left(p_{\theta_k}^n(y_{i+n}|\bar{g}(x_{\leq i+n}))\right)$. Following (Manning & Schutze, 1999), we consider languages to show stationary, ergodic properties. Now, taking $m \to \infty$, and using stationarity of $p(\cdot)$, we define $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test}) = \lim_{m\to\infty} \mathcal{L}_{Test}^{(l,k,m,n)}(\mathcal{D}_{Test})$ to get

$$\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test}) =$$
$$-\sum_{(y_k, x_k, \ldots, x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1)\log\left(p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))\right). \tag{1}$$

For fixed data Markov order l with known state function g and estimator Markov order k, suppose the minimum loss is obtained by an estimator $p_{\theta_k}^{n,*}$. Then, this minimum loss is

$$\mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test}) =$$
$$-\sum_{(y_k, x_k, \ldots, x_1)\in\{0,1\}^{k+1}} p(y_k, x_k, \ldots, x_1)\log\left(p_{\theta_k}^{n,*}(y_k|g(x_k, \ldots, x_1))\right). \tag{2}$$

Bayesian estimator An order-k Bayesian estimator with a given state function g is of the form

$$p_{\theta_k}^n(Y_{k+i} = y_k|g(X_{k+i}, \ldots, X_{1+i}) = g(x_k, \ldots, x_1))$$
$$= \begin{cases} \frac{N(y_k; g(x_k, \ldots, x_1))}{N(g(x_k, \ldots, x_1))} & \text{if } k > 0, \\ \frac{N(y_0)}{n} & \text{if } k = 0, \end{cases} \tag{3}$$

where $N(y_k; g(x_k, \ldots, x_1))$ denotes the number of counts of $Y_{k+i} = y_k$ and $g(X_{k+i}, \ldots, X_{1+i}) = g(x_k, \ldots, x_1)$ for $0 \leq i \leq (n - k)$ in the train set, $\mathcal{D}_{Train}$. Similarly, $N(g(x_k, \ldots, x_1))$ is the number of occurrences of the subsequence $g(X_{k+i}, \ldots, X_{1+i}) = g(x_k, \ldots, x_1)$, $N(y_0)$ is the number of occurrences of $Y_i = y_0$ for $0 \leq i \leq (n - k)$, and n is the size of the train dataset. Note that the order l and the state function g of the source is unknown to the estimator, hence it is not obvious what k and g should be chosen for best test performance.

Now we state and prove the theorem giving limits to the test loss obtained by any Transformer architecture with attention span k. Thereafter we show the obtained limits are indeed optimal. Proofs are in Appendix A.

Theorem 3.1. *Let $\{\mathcal{D}_{Train}, \mathcal{D}_{Test}\}$ be an order-l Markov dataset with some state function g. Let $\bar{g}$ be the state function of a Transformer such that $H(Y_m|g(X_m, \ldots, X_1)) = H(Y_m|\bar{g}(X_m, \ldots, X_1))$ for $m = k, l$, e.g. choose $\bar{g}$ as the identity function and let $p_{\theta_k}^n$ be the order-k Bayesian estimator in (3) with state function $\bar{g}$. Let $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ be the corresponding test loss of the Transformer.*

Then, if $l \leq k$,

$$\lim_{n\to\infty} \mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test}) = H(Y_l|X_l, \ldots, X_1), \tag{4}$$

whereas, if $l > k$,

$$\lim_{n\to\infty} \mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test}) = H(Y_k|X_k, \ldots, X_1). \tag{5}$$

Here $H(Y_m|X_m, \ldots, X_1)$ is the conditional entropy of the distribution $p(y_m|x_m, \ldots, x_1)$ for $m = l, k$.

Moreover, $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test}) \to \mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test})$ with convergence error $\mathcal{O}(\frac{|\mathcal{S}|}{n})$ in expectation in both the cases, (4), (5), where $\mathcal{S}$ is the range of the state function $\bar{g}$.

Now, we show that the limits obtained in Thm. 3.1 are optimal.
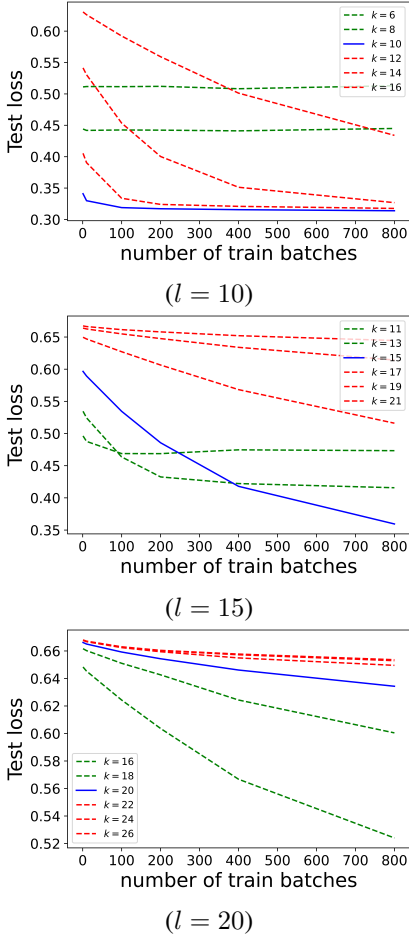
$(l = 10)$

$(l = 15)$

$(l = 20)$

Figure 3. Performance of Finite State Markov Predictor on a Boolean synthetic dataset, where the output is 1 if the sum of previous l observations is greater than a threshold, else 0. Plot illustrates the nature of the obtained limit in Thm. 3.1. Smaller values of k converges faster but to a higher value of test loss compared to larger values.

Theorem 3.2. *Let $\{\mathcal{D}_{Train}, \mathcal{D}_{Test}\}$ be an order-l Markov dataset with some state function g. Then, let h be some arbitrary function taking as input $(X_i, \ldots, X_{i+k})$, and outputs some probability $h(Y_{i+k}|X_i, \ldots, X_{i+k})$ over the space of Y_{i+k}. Then, the optimal cross-entropy loss obtained by h, $\sum_{(y_k, x_k, \ldots, x_1) \in \{0,1\}^k} p(y_k, x_k \ldots, x_1) \log\left(h(y_k|x_k \ldots, x_1)\right)$ is greater than or equal to $H(Y_l|X_l, \ldots, X_1)$ if $l \leq k$, else, greater than or equal to $H(Y_k|X_k, \ldots, X_1)$.*

3.3. Understanding the limits

From the limits in Thm. 3.1, we know that for $k < l$, increasing the value of k improves the obtained limit. This is because conditioning reduces entropy and hence using the stationarity property $H(Y_k|X_k, \ldots, X_1) \leq H(Y_{k-1}|X_{k-1}, \ldots, X_1)$ (Cover & Thomas, 1999). But, for $k > l$, $H(Y_l|X_l, \ldots, X_1) = H(Y_k|X_k, \ldots, X_1)$. To

better understand the limits in Thm. 3.1, we show the performance of an FSMP with the aggregation function g as the identity function with span of the input k, and the output probability function f is set as the Bayesion predictor in (3). We test the performance of the FSMP on a synthetic Boolean dataset, where the input is randomly generated binary values and the output at position i is 1 if the sum of the past l observations is greater than a threshold, else, 0. The main observation to note from Fig. 3 is that for smaller values of k, the convergence is faster compared to larger values, but, the value the losses converge to is much worse than for larger values of k.

The limits in Thm. 3.1 gives us two important design choices for efficiently designing and training Transformers. First, the choice of the state function $\bar{g}$ and its attention span k. Second, the amount of data, n, to train on.

The choice of $\bar{g}$ and k must be such that it does not lose any information important to the output, i.e. $H(Y_m|g(X_m, \ldots, X_1)) = H(Y_m|\bar{g}(X_m, \ldots, X_1))$. One such $\bar{g}$ can be simply the identity function. The corresponding k must be chosen large enough, i.e. $k \geq l$. Note that for all such $\bar{g}$ and k the optimal value remains the same as $n \to \infty$, but, in the finite data regime, one needs to choose appropriate $\bar{g}$ and k to get the best performance. In particular, one needs to choose $\bar{g}$ and k that has the smallest range $\mathcal{S}$ while also satisfying $H(Y_m|g(X_m, \ldots, X_1)) = H(Y_m|\bar{g}(X_m, \ldots, X_1))$. Thus, it indicates the need for sparsity in the design.

For the second choice, n, while it may seem obvious that training on larger data would give better performance, we show that with increase in data, the improvement in performance is significant, even with training for the same number of training steps. Moreover, this also motivates an augmentation technique that helps statistically explain the benefit gained from using relative positional encodings (Shaw et al., 2018).

In Appendix B, we further aim to understand Transformers and data-efficient training. In Appendix C, we provide further experiments on real-world data.

4. Conclusion

Here, we prove that the Transformer architecture yields universal predictors in the information-theoretic sense, which may help explain why this architecture seems to be a kind of state-of-the-art universal computation system. We further analyzed the different components of the Transformer architecture, such as attention weights and positional encodings.

Going forward, it may be of interest to take inspiration from information-theoretic work in universal prediction (Merhav & Feder, 1998) to inspire novel neural architectures.

References

Anil, R., Dai, A. M., Firat, O., Johnson, M., Lepikhin, D., Passos, A., Shakeri, S., Taropa, E., Bailey, P., Chen, Z., Chu, E., Clark, J. H., et al. PaLM 2 technical report. arXiv:2305.10403 [cs.CL]., May 2023.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34, 2021.

Chen, S., Dobriban, E., and Lee, J. A group-theoretic framework for data augmentation. *Advances in Neural Information Processing Systems*, 33:21321–21333, 2020.

Cover, T. M. and Shenhar, A. Compound Bayes predictors for sequences with apparent Markov structure. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(6): 421–424, 1977.

Cover, T. M. and Thomas, J. A. *Elements of information theory*. John Wiley & Sons, 1999.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2:303–314, December 1989.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Feder, M., Merhav, N., and Gutman, M. Universal prediction of individual sequences. *IEEE transactions on Information Theory*, 38(4):1258–1270, 1992.

Golkar, S., Tesileanu, T., Bahroun, Y., Sengupta, A., and Chklovskii, D. Constrained predictive coding as a biologically plausible model of the cortical hierarchy. In Koyejo, S., Mohamed, S., Agarwal, A., Belgrave, D., Cho, K., and Oh, A. (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 14155–14169. Curran Associates, Inc., 2022.

Gurevych, I., Kohler, M., and Şahin, G. G. On the rate of convergence of a classifier based on a transformer encoder. *IEEE Transactions on Information Theory*, 68 (12):8139–8155, December 2022.

Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., and Carreira, J. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pp. 4651–4664. PMLR, 2021.

Li, N., Liu, S., Liu, Y., Zhao, S., and Liu, M. Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 6706–6713, 2019.

Lu, K., Grover, A., Abbeel, P., and Mordatch, I. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 2021.

Manning, C. and Schutze, H. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

Merhav, N. and Feder, M. Universal prediction. *IEEE Transactions on Information Theory*, 44(6):2124–2147, October 1998.

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

OpenAI. GPT-4 technical report. arXiv:2303.08774 [cs.CL]., March 2023.

Romero, D. W. and Cordonnier, J.-B. Group equivariant stand-alone self-attention for vision. *arXiv preprint arXiv:2010.00977*, 2020.

Shannon, C. E. A mathematical theory of communication. *Bell Syst. Tech. J.*, 27:379–423, 623–656, July/Oct. 1948.

Shannon, C. E. The redundancy of English. In *Trans. 7th Conf. Cybern.*, pp. 123–158, March 1950.

Shannon, C. E. Prediction and entropy of printed English. *Bell Syst. Tech. J.*, 30(1):50–64, January 1951.

Shaw, P., Uszkoreit, J., and Vaswani, A. Self-attention with relative position representations. *arXiv preprint arXiv:1803.02155*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., and Fedus, W. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Weissman, T. and Merhav, N. Universal prediction of individual binary sequences in the presence of noise. *IEEE Transactions on Information Theory*, 47(6):2151–2173, September 2001.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S., and Kumar, S. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations (ICLR)*, 2020.

A. Proofs

Lemma A.1. $H(Y_l|g(X_l, \ldots, X_1)) = H(Y_l|X_l, \ldots, X_1)$ *for the data generation process in Sec. 3.*

Proof. We know $Y_l - g(X_l, \ldots, X_1) - (X_l, \ldots, X_1)$ forms a Markov chain, hence, $H(Y_l|g(X_l, \ldots, X_1), X_l, \ldots, X_1) = H(Y_l|g(X_l, \ldots, X_1))$, But, since $g(X_l, \ldots, X_1)$ is a function of $(X_l, \ldots, X_1)$, we have $H(Y_l|g(X_l, \ldots, X_1), X_l, \ldots, X_1) = H(Y_l|X_l, \ldots, X_1)$. $\qquad\square$

Thm. 3.1. First we consider the case where $l = 0, k = 0$ and understand the behavior of $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ and $\mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test})$ as a function of n. Then, we will use the results from these cases to understand more general cases varying both l and k.

Case $l = 0, k = 0$ Consider order 0 estimator in (3). We first compute $\mathcal{L}_{Test}^{(0,0,n),*}(\mathcal{D}_{Test})$ by writing

$$\mathcal{L}_{Test}^{(0,0,n)}(\mathcal{D}_{Test}) = \sum_{y_0 \in \{0,1\}} p(y_0) \log \frac{1}{p(y_0)} + D(p||p_{\theta_0^n}), \tag{6}$$

where $D(p||p_{\theta_0^n}) = \sum_{y_0 \in \{0,1\}} p(y_0) \log \frac{p(y_0)}{p_{\theta_0^n}(y_0)}$ is the KL divergence between p and $p_{\theta_0^n}$. Further, from basic information-theoretic inequalities we know that $D(p||q) \geq 0$ with the equality holding only when $p = q$ (Cover & Thomas, 1999). Thus, we have $\mathcal{L}_{Test}^{(0,0,n),*}(\mathcal{D}_{Test}) = H(Y_0)$, where $H(Y_0) = \sum_{y_0 \in \{0,1\}} p(y_0) \log \frac{1}{p(y_0)}$ is the entropy of the distribution $\{p(y_0) : y_0 \in \{0,1\}\}$. Moreover, we have $p_{\theta_0}^{n,*}(y_0) = p(y_0)$ for $y_0 \in \{0,1\}$.

Now, we show that $D(p||p_{\theta_0^n}) \to 0$ as $n \to \infty$ and further analyze its convergence rate. Using concentration inequalities, it follows that $p_{\theta_0^n} \to p(y_0)$ with rate $\mathcal{O}(\frac{1}{\sqrt{n}})$ with high probability. For brevity, we do not provide the calculations and use results from Cover & Shenhar (1977) that show the convergence rate to be $\mathcal{O}(\frac{1}{\sqrt{n}})$ in expectation. Now we show that this implies $D(p(y_0)||p_{\theta_0^n}) \to 0$ with an approximate rate $\mathcal{O}(\frac{1}{n})$. To this end, take $p_{\theta_0^n} = p(y_0) + t$, where $t = \mathcal{O}(\frac{1}{\sqrt{n}})$. Say, $p(1) = p = p(y_0)$ and hence, $p(0) = 1 - p$. Then, we have

$$D(p||p+t) = p \log \frac{p}{p+t} + (1-p) \log \frac{(1-p)}{(1-p-t)}$$
$$= -\frac{1}{\ln 2} \left(p \ln\left(1 + \frac{t}{p}\right) + (1-p) \ln\left(1 - p - t\right) \right).$$

Assuming n to be large enough such that $t \ll \min\{p, 1-p\}$ and using the Taylor expansion of $\ln(1-x)$ up to second-order terms, we have

$$D(p||p+t) \approx \frac{t^2}{\ln 2} \left(\frac{1}{p} + \frac{1}{1-p} \right).$$

Thus, $D(p||p_{\theta_0^n}) \to 0$ with $\mathcal{O}(\frac{1}{n})$ in expectation.

Next, we generalize these optimal loss and convergence results for general l and k using results from the current case.

Case $l \leq k$ Here we have the evaluation loss $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ as

$$\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test}) = H(Y_l|\bar{g}(X_l, \ldots, X_1)) +$$
$$\sum_{(x_k, \ldots, x_1)} p(x_k, \ldots, x_1) \times \tag{7}$$
$$D(p(y_k|\bar{g}(x_k, \ldots, x_1))||p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))), \tag{8}$$

where $p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))$ is as defined in (3). Since $D(p||q)) \geq 0$ with equality holding only when $p = q$, we have $\mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test}) = H(Y_l|\bar{g}(X_l, \ldots, X_1)) = H(Y_l|g(X_l, \ldots, X_1)) = H(Y_l|X_l, \ldots, X_1)$, where $H(Y_l|g(X_l, \ldots, X_1))$ is

the conditional entropy of the distribution $p(y_k|g(x_k, \ldots, x_1))$. The equality $H(Y_l|g(X_l, \ldots, X_1)) = H(Y_l|X_l, \ldots, X_1)$ follows directly and is proved in Lem. A.1 for completeness.

Moreover, we have $p_{\theta_k}^{n,*}(y_k|\bar{g}(x_k, \ldots, x_1)) = p(y_k|\bar{g}(x_k, \ldots, x_1))$ for $(y_k, x_k, \ldots, x_1) \in \{0,1\}^{k+1}$. Note that here, for $l \leq k$, $\mathcal{L}_{Test}^{(l,k,n),*}(\mathcal{D}_{Test}) = H(Y_l|\bar{g}(X_{l-1}, \ldots, X_0))$ is independent of k.

Now we look at the convergence rate of $D(p(y_k|\bar{g}(x_k, \ldots, x_1))||p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))) \to 0$ as $n \to \infty$. Here we only perform approximate calculations as this is sufficient for present purposes. From the previous case with $l = 0, k = 0$, we know $D(p(y_0)||p_{\theta_0}^n(y_0)) \to 0$ as $\mathcal{O}(\frac{1}{n})$. For the convergence of $D(p(y_k|\bar{g}(x_k, \ldots, x_1))||p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1)))$, we will only look at the indices y_k that have prefix $\bar{g}(x_k, \ldots, x_1)$. We know that the number of such prefixes is approximately equal to $np(\bar{g}(x_k, \ldots, x_1))$ in expectation. Thus $D(p(y_k|\bar{g}(x_k, \ldots, x_1))||p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))) \to 0$ as $\mathcal{O}(\frac{1}{np(\bar{g}(x_k,\ldots,x_1))})$. Thus, from (7) $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ converges to $H(Y_l|\bar{g}(X_l, \ldots, X_1))$ approximately as $\sum_{\bar{g}(x_k,\ldots,x_1)} p(\bar{g}(x_k, \ldots, x_1))\mathcal{O}(\frac{1}{np(\bar{g}(x_k,\ldots,x_1))}) \approx \mathcal{O}(\frac{|\mathcal{S}|}{n})$.

Case $l > k$ Here the evaluation loss $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ is given by

$$\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test}) = H(Y_k|\bar{g}(X_k, \ldots, X_1))$$
$$+ \sum_{(x_k,\ldots,x_0)} p(x_k, \ldots, x_1)\times \tag{9}$$
$$D(p(y_k|\bar{g}(x_k, \ldots, x_1))||p_{\theta_k}^n(y_k|\bar{g}(x_k, \ldots, x_1))), \tag{10}$$

The optimal value of $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ is clearly $H(Y_k|\bar{g}(X_k, \ldots, X_1)) = H(Y_k|g(X_k, \ldots, X_1)) = H(Y_k|X_k, \ldots, X_1)$ from non-negativity of KL divergence and the data generation process. Moreover, the convergence rate of $\mathcal{L}_{Test}^{(l,k,n)}(\mathcal{D}_{Test})$ to $H(Y_k|X_k, \ldots, X_1)$ approximately with rate $\mathcal{O}(\frac{|\mathcal{S}|}{n})$, the same way as in the previous case. $\quad\square$

Thm. 3.2. The proof follows directly from the non-negativity of KL divergence.

$l \leq k$ Here, we have

$$\sum_{(y_k,x_k,\ldots,x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1) \log\left(h(y_k|x_k \ldots, x_1)\right)$$
$$= H(Y_k|X_k, \ldots, X_1) + \sum_{(x_k,\ldots,x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1)\times$$
$$D(p(y_k|x_k \ldots, x_1)||h(y_k|x_k \ldots, x_1))$$
$$= H(Y_l|X_l, \ldots, X_1) + \sum_{(x_k,\ldots,x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1)$$
$$D(p(y_k|x_k \ldots, x_1)||h(y_k|x_k \ldots, x_1))$$

Note that $H(Y_k|X_k, \ldots, X_1) = H(Y_l|X_l, \ldots, X_1)$ because Y_k only depends on the past l observations and the stationarity assumption. Further, we have $D(\cdot||\cdot) \geq 0$ to conclude the proof.

$l > k$ Here, we have

$$\sum_{(y_k,x_k,\ldots,x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1) \log\left(h(y_k|x_k \ldots, x_1)\right)$$
$$= H(Y_k|X_k, \ldots, X_1) + \sum_{(x_k,\ldots,x_1)\in\{0,1\}^k} p(y_k, x_k \ldots, x_1)$$
$$D(p(y_k|x_k \ldots, X_1)||h(y_k|x_k \ldots, x_1)),$$
$$\geq H(Y_k|X_k, \ldots, X_1)$$

where we have $D(\cdot||\cdot) \geq 0$. $\quad\square$

B. Understanding Transformers and Data-Efficient Training

First, we understand the role of various components of a Transformer in the prediction process. Then, we show the importance of relative position encodings (Shaw et al., 2018) and group-equivariant positional encodings (Romero & Cordonnier, 2020) statistically in resource-constrained datasets. In particular, our results show that these methods are beneficial only when the available data is small, but their effect is negligible for larger datasets. For this analysis, we use position augmentations and group augmentations instead of equivariance for the ease of analysis. In practice, both equivariance and augmentation enforce the same constraint on the model. While equivariance applies the constraints by design in the model, augmentation does it more implicitly when training.

B.1. Understanding Transformers

Here we discuss the role of various components of a Transformer.

Attention weights Attention weights play an important role in filtering out unimportant data points and hence help make the range of the state function $\bar{g}$ as small as possible while also ensuring that the important information is retained. We validate this claim by showing that when attention weights are used, additional irrelevant information does not affect the loss values. Whereas, without attention, the loss values are negatively affected.

On the other hand, in the extreme case where all the input variables are important, we show, perhaps surprisingly, that attention performs worse than a network without it.

Absolute positional encoding and FFN Absolute positional encoding introduced by Vaswani et al. (2017) helps identify the order of the words in a sentence for the Transformer. Without positional encoding, the output of a Transformer is equivariant to the permutation of words in the input. This is not desirable since changing the order of words in a sentence can completely change its meaning or may even make it gibberish. Hence, the understanding of the order of the words is important to a Transformer.

In our experiments, we find that FFN does not effectively filter out unnecessary information like attention. But it does improve the results when used with attention, indicating that it helps produce a better representation of the state of the Transformer.

B.2. Data-Efficient Training: Positional Augmentation

Equivariance and augmentations are considered to be two sides of the same coin. Whereas one applies constraints on the model, the other transforms the data during the training process to implicitly introduce those constraints in the learnt functions. Here, we use group augmentations (Chen et al., 2020) to study its impact on the performance of Transformers as it helps our theoretical analysis. Augmentations do not change the asymptotic performance limits, but, improves finite data performance as we find next.

Positional augmentation We study the role of translation augmentation in the performance of Transformers. The key to our analysis is the following: we transform the data while keeping the position encodings the same and since the input to the Transformer is a summation of the data and the absolute encodings, the input appears as new data. This effectively increases the number of training data points in the dataset. Thus, for each augmentation, we count the effective size of the augmented dataset and analyze the performance from the finite data loss expression obtained in Thm. 3.1.

For translation augmentation for a dataset of size n and each batch processing n_{pos} tokens at once. A translation augmentation parametrized by t_0, can be obtained by shifting the data by t, where t goes from t_0 to $n_{pos} - 1$. This way, the semantics of text does not change, but we generate more data points. From the finitary analysis in Thm. 3.1, we know the loss value converges with rate $O(1/n)$. Hence, with augmentation t_0, we have the following gain in the convergence rate. The proof is direct from the assumption that there are no repetitions in the dataset and counting the size of the augmented dataset.

Proposition B.1. *Let a dataset of size n be augmented by t_0 with n_{pos} tokens processed per batch. Then the gain convergence rate is $O(\frac{1}{n}(1 - \frac{1}{(n_{pos}-t_0)}))$.*

Two important consequences to note from Prop. B.1 are: a) the gain decreases proportionally with increase in n, b) it increases with decrease in t_0.

Table 1. Attention

Dataset	Attention			Aggregation		
k	5	10	15	5	10	15
MarkovBoolSum	0.016	0.34	0.36	0.011	0.375	0.421
MarkovBin2Dec	0.036	0.87	0.99	0.015	0.91	1.08
WikiText2	6.17	6.18	6.19	6.49	6.66	6.70

Table 2. Position augmentation

n	1		1000	
t_0	0	99	0	99
loss values	9.68	0.87	0.628	0.624

C. Further Experiments

Here we validate our findings using various experiments.

C.1. Datasets

We work with three sequential datasets: two are synthetic and one is a real language dataset, WikiText2 (Merity et al., 2016). The synthetic datasets are MarkovBoolSum and MarkovBin2Dec. MarkovBoolSum with Markovity l, is constructed by first uniformly generating a binary input, then the labels are set as 1 if the sum of previous l inputs sum to greater than or equal to $l/2$, else 0. For MarkovBin2Dec, the input is generated in a similar way as MarkovBoolSum, but, the output is the decimal value corresponding to the binary observation of the l previous observations. The main difference between the two datasets is that in MarkovBoolSum, not all the inputs contribute to the output, hence attention should help performance. Whereas, in MarkovBin2Dec, all the input values contribute to the output, hence, attention might not perform as well. All experiments were run on one fixed seed.

C.2. Model Architecture

We use a single attention layer Transformer with varying attention span. The model dimension is fixed to 64, and the hidden dimension for FFN is set to 128. For our experiments, we keep three design choices: a) attention or aggregation; b) FFN or no FFN; c) translation augmentation.

C.3. Attention

In Tab. 1, we show results that indicate the benefit of attention. The synthetic datasets used $l = 5$, and were trained for 10000 train batches for 2000 steps of batch size 20. It is found that using attention, the difference between varying lengths of span k is less than without attention. Also, in the case $k = 5$ for MarkovBin2Dec, we find that attention performs poorly compared to no attention. This is because, for $k = 5$, all points in the input are important and attention may have neglected some of the inputs.

C.4. Positional Encodings

The results for position augmentation are given in Tab. 2, where we find that augmentation helps improve test loss performance. But, the loss is significant only when the number of train batches are very small, corresponding to our theoretical analysis.