DON'T BET ON SPARSITY: DESIGNING BRAIN-INSPIRED DISTANCE-PRESERVING ENCODER

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-headed self-attention-based Transformers have been a central area of research for quite some time. Albeit showing a significant improvement in understanding short-term and long-term contexts from sequences, encoders of Transformer and its variants fail to preserve layer-wise contextual information. Further, text representations learned by Transformer-based encoders are usually of low entropy with low variance, which contradicts typical human brain functions. In this work, we propose `TransJect`, an encoder model that guarantees a theoretical bound for layer-wise distance preservation between any pair of tokens. We propose a simple alternative to dot product attention to ensure Lipschitz continuity that allows `TransJect` to learn injective mappings to transform token representations to different manifolds and preserve Euclidean distance between every pair of tokens in subsequent layers. Our evaluation on several benchmark short- and long-sequence classification tasks shows a remarkable improvement of 3.1% and 11%, on average, respectively. Furthermore, empirical results suggest that `TransJect` is layer-agnostic; in fact, it prefers shallower architectures than deeper ones and prevents layer-wise incremental learning beyond a threshold. Our empirical analyses also show the generalization capabilities of `TransJect` and the robustness under different hyperparameter configurations. We conduct detailed statistical analysis to confirm the necessity of high-entropic representations to achieve human-like cognition.

1 INTRODUCTION AND RELATED WORK

Over the past few decades, deep neural networks (DNNs) have greatly improved the performance of various downstream applications. Stacking multiple layers has been proven effective in extracting features at different levels of abstraction, thereby learning more complex patterns (Brightwell et al., 1996; Poole et al., 2016). Since then, tremendous efforts have been made to build larger depth models and make them faster (Bachlechner et al., 2020; Xiao et al.). Self-attention-based Transformer model Vaswani et al. (2017) was proposed to parallelize the computation of longer sequences; it has achieved state-of-the-art performance in various sequence modelling tasks. Following this, numerous efforts have been made to reduce computation and make the Transformer suitable even for longer sequences (Katharopoulos et al., 2020; Peng et al., 2020; Kitaev et al., 2020; Beltagy et al., 2020; Press et al., 2021; Choromanski et al., 2021; Tay et al., 2021). However, very few of these studies discuss information propagation in large-depth models. To understand how Transformer encodes contextual similarities between tokens and preserves the similarities layer-wise, we highlight an example in Figure 1. We select three pairs of semantically similar tokens (two of them are present in the same articles with different contexts). We observe both the Euclidean and cosine distances of the representations learned at different layers of pre-trained BERT-base (Devlin et al., 2018) and Transformer trained on the IMDb sentiment classification task. The fine-tuned Transformer model preserves the semantic similarity among the same tokens across layers. However, the Euclidean distance increases in the upper layers, concluding that Transformer projects the representations to different and sparse subspaces, albeit preserving the angle between them. On the other hand, pre-trained BERT demonstrates a more erratic trend in terms of layer-wise preservation of information. Preserving both distance and semantic similarity is important to ensure that the model learns continuously without forgetting any previously learned knowledge, much alike to humans.

Neuroscientists have been working for years to understand how the human brain functions and why the human brain is superior to most other animals. Arguably, the human brain is more capable of ‘associative learning’ and ‘behavioral formation’, which can be attributed to the number of neurons and their inter-connectedness, rather than the size of the brain, or the number of layers through which the information propagates (Dicke & Roth, 2016). Another recent direction in exploring the human brain is to understand its energy state through the lens of entropy (Saxe et al., 2018). Keshmiri (2020) correlated human intelligence with entropy, a measure that quantifies the randomness of a state. Interestingly, the human brain contains both high- and low-entropic neurons. High-entropic neurons are responsible for divergent and creative thinking, whereas low-entropic neurons are responsible for mundane and rigid thinking. Similar attempts have been made to understand the bridge between human brain, thermodynamics and information theory (Collell & Fauquet, 2015). Unfortunately, Transformer and none of its variants consider these viewpoints in their underlying architecture, and thus fail to imitate the euphoric nature of the human brain. This motivates us to develop a complete redesign of self-attention-based **Trans**former with explicit randomness and enforced in**ject**ivitiy, *aka* **`TransJect`**.

To the best of our knowledge, this is the first attempt toward this research direction. A significant effort has been made in reducing the computational complexity of Transformer by introducing sparsity in self-attention (Zaheer et al., 2020; Tay et al., 2020), approximating softmax (Choromanski et al., 2021; Peng et al., 2020), or introducing different kernel tricks (Choromanski et al., 2021; Katharopoulos et al., 2020). With a simple orthogonal parameterization and rearrangement of a non-normalized attention formulation, we show that self-attention can be simplified to simple matrix multiplication. Moreover, by not projecting the embeddings into low-dimensional space, `TransJect` encourages more synapses within the model, resulting in better contextualization. ReZero was proposed by Bachlechner et al. (2020) to felicitate dynamic isometry and faster convergence for large-depth models. With a modified formulation of ReZero, not only our model maintains dynamic isometry but also achieves *Lipschitz* continuity, aiding in preserving layer-wise distances between tokens. By preserving layer-wise information within a fixed bound, `TransJect` ensures incremental learning throughout the encoding layers, similar to how humans learn continually. Kim et al. (2021) concluded that vanilla dot-product self-attention is not Lipschitz. An enforced Lipschitz condition allows us to build layer-agnostic models, where shallower models achieve better performance. Unlike Transformer, which only injects randomness through multi-headed self-attention and dropout, we inject randomness explicitly to every token representation. Due to such a direct injection, our model does not require dropouts for generalization; but it generalizes naturally. To validate our hypotheses and empirically justify the superiority of our model, we use two short and two long sequence classification tasks, in which `TransJect` outperforms Transformer and other benchmark variants with a wide margin of 7% across all four tasks. We report the model’s performances under different hyperparameter configurations to understand the robustness of the model. We further conduct detailed statistical analyses to investigate different components and evaluate their effectiveness.

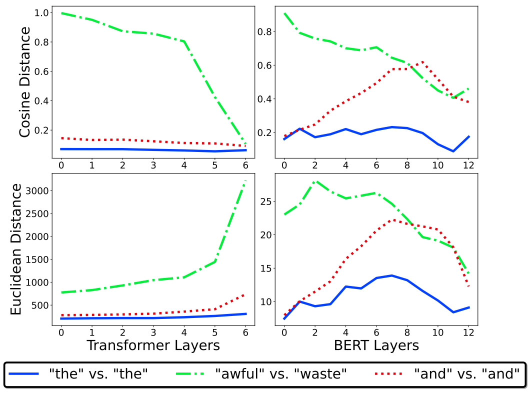


Figure 1: Layer-wise distances between a few selected tokens from the text “*The characters are unlikeable and the script is awful. It’s a waste of the talents of Deneuve and Auteuil.*” We use pre-trained BERT and Transformer models for extracting the token representations from different encoding layers.

2 Background

Activation bound. For any function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$, we define the activation bound K_f as $\sup_{\boldsymbol{x} \neq 0} \frac{||\boldsymbol{f}(\boldsymbol{x})||_p}{||\boldsymbol{x}||_p}$, for a suitable integer p. For a linear map $\boldsymbol{M}$, it is equivalent to the induced matrix norm $||\boldsymbol{M}||_p$. Intuitively, this is the maximum scale factor by which a mapping expands a vector $\boldsymbol{x}$. In the Euclidean space, we usually choose $p = 2$.

Lemma 1 (Activation bound of linear maps). For a matrix $\boldsymbol{M} \in \mathbb{R}^{n \times m}$, $K_{\boldsymbol{M}}$ is same as the largest absolute singular value, under $||.||_2$.

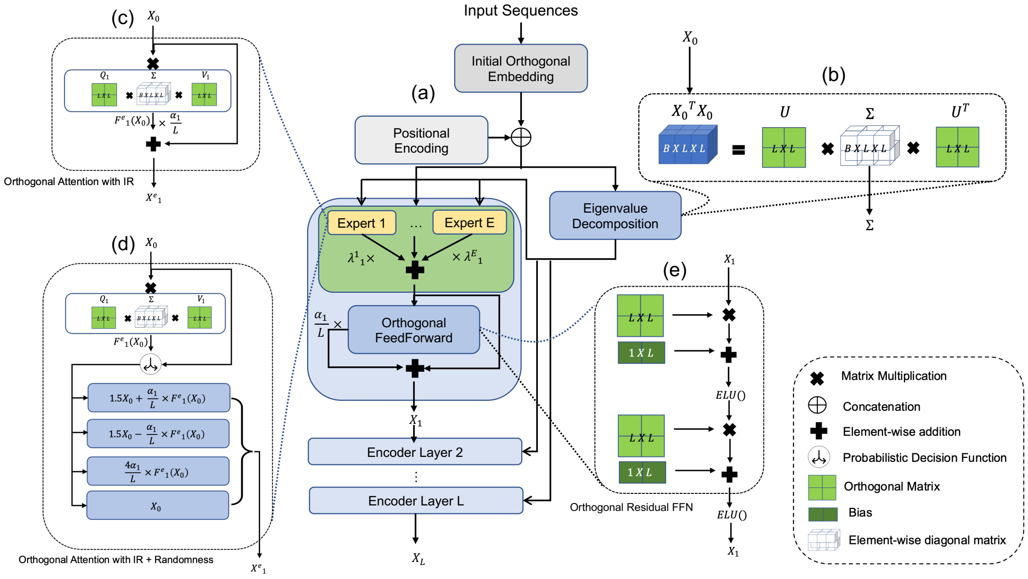


Figure 2: Internals of TransJect with (a) an L-layered encoder with MOE, (b) approximated eigenvalue computation, (c) orthogonal attention with injective residual. For the entropic variant of TransJect, we use a decision function to determine the residual output (d). The intermediate encoder outputs are fed to orthogonal residual FFN (e) layer to extract the final representation at layer l.

Activation bound of self-attention. Transformer (Vaswani et al., 2017) relies upon bidirectional multi-headed self-attention for capturing contextual similarity between tokens in the encoder. In each head, the self-attention operation projects the original representation vector $\boldsymbol{x}$ into three difference subspaces with $\boldsymbol{Q}$ (query), $\boldsymbol{K}$ (key) and $\boldsymbol{V}$ (value). Formally, Transformer calculates attention matrix $SelfAttention(\boldsymbol{X}) = Attention(\boldsymbol{X}\boldsymbol{W}^Q, \boldsymbol{X}\boldsymbol{W}^K, \boldsymbol{X}\boldsymbol{W}^V)$ as,

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{D}^{-1}\boldsymbol{A}\boldsymbol{V}, \boldsymbol{A} = exp\Big(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}\Big), \boldsymbol{D} = diag(\boldsymbol{A}\boldsymbol{I}_L) \tag{1}$$

Here d is the hidden size, and L is the length of the sequence. With the normalization, $\boldsymbol{D}^{-1}\boldsymbol{A}$, Transformer ensures a stochastic attention weights (column sum is 1).

Corollary 1 (Largest eigenvalue of a square stochastic matrix). The largest absolute value of any eigenvalue of a square stochastic matrix is equal to 1.

As the activation bound of the matrix $\boldsymbol{D}^{-1}\boldsymbol{A}$ is 1, activation bound of the attention map is same as the activation bound of $\boldsymbol{W}^V$. Usually, $\boldsymbol{K}$ and $\boldsymbol{Q}$ are used to project the original representations to different subspaces. We show that having both $\boldsymbol{Q}$ and $\boldsymbol{K}$ in the same subspace with orthogonal basis can reduce the activation bound of the attention map to singular values of $\boldsymbol{X}$.

Lipschitz Continuity. A function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ under $||.||_p$ norm is called *Lipschitz continuous* if there exists a real number $K \geq 0$ such that

$$||f(\boldsymbol{x}) - f(\boldsymbol{y})||_p \leq K||\boldsymbol{x} - \boldsymbol{y}||_p \tag{2}$$

for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$. Although Lipschitz continuity can be defined over any metric space, in this paper, we restrict its definition to only Euclidean space with $p = 2$. K is called *Lipschitz bound*.

Lemma 2 (Lipschitz bound for continuously differentiable functions). Any $\mathcal{C}^1$ function $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ has Lipschitz bound as $sup_{\boldsymbol{x}}||\nabla_{\boldsymbol{x}} f||$.

3 Designing Injective Transformer

In this section, we formally describe our model, TransJect and its variants. Our model inherits the structure from vanilla Transformer, and achieves a smoother activation plane by utilizing injective maps for transforming token representations across layers. For an L-layered stacked encoder, we aim to learn the representation of a sequence $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_n\}$ at each layer l that preserves the pairwise distance between every pair of words within a theoretical bound. We illustrate the components of `TransJect` in Figure 2.[1]

3.1 Space-Preserving Orthogonal Attention

The backbone of `TransJect` is the space-preserving orthogonal attention.

Theorem 1 (Space-Preserving Orthogonal Attention). Replacing $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V$ with real square orthogonal matrices in self-attention reduces the activation bound to the largest singular value of $\boldsymbol{X}$.

Notice that the activation bound of the modified attention mechanism does not depend on any learnable parameters, rather can be bounded by the largest eigenvalue of $\boldsymbol{X}^T\boldsymbol{X}$. Therefore, having a stochastic $\boldsymbol{X}^T\boldsymbol{X}$ will ensure that the largest eigenvalue is always 1, and the attention operator preserves pairwise distance between any two tokens. In each layer, we learn orthogonal projection matrices, $\boldsymbol{U}$ and $\boldsymbol{V}$, whereas the diagonal matrix containing eigenvalues $\boldsymbol{\Sigma}$ is learned on the initial embedding obtained from the initial embedding layer defined in Section 3.4, also denoted as $l = 0$.

Approximating eigenvalues. Eigenvalue decomposition is computationally expensive with a runtime complexity of $\mathcal{O}(n^3)$. In this work, we use a simple approximation to compute $\tilde{\boldsymbol{\Sigma}}$, the eigenvalues of $\boldsymbol{X}^T\boldsymbol{X}$. Formally, we compute $\tilde{\boldsymbol{U}} = \arg\min_{\boldsymbol{U}} ||\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T||$, and $\tilde{\boldsymbol{\Sigma}} = \arg\min_{\boldsymbol{\Sigma}} ||\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T||$. To learn the approximate eigenvalues, we can minimize the reconstruction loss $||\boldsymbol{X}^T\boldsymbol{X} - \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{U}^T||$ for a learnable orthogonal eigenvectors U. However, we can also initialize a random diagonal matrix $\tilde{\boldsymbol{\Sigma}}$, without any approximation that optimizes the only task-specific training objective, without enforcing the reconstruction. We denote this version of the model as **`Random-TransJect`**. We compute $\tilde{\boldsymbol{\Sigma}}$ only once, only on the initial token embeddings. Further, we standardize the eigenvalues $\boldsymbol{\Sigma}$ to maintain the largest absolute eigenvalue as 1.

3.2 Injective Residual (IR)

For every layer l, we fine-tune the hidden representation by learning a new attention projection on the hidden state learned in the previous layer. Formally, we define,

$$\boldsymbol{X}^{(l)} = \boldsymbol{X}^{(l-1)} + \frac{\alpha_l}{L} F(\boldsymbol{X}^{(l-1)}) \tag{3}$$

Here, F is the self-attention operator, followed by a suitable non-linear activation function, and $\alpha_i \in (0, 1)$ is the residual weight. In the previous studies, ReLU and GELU (Hendrycks & Gimpel, 2016) have been popular choices for the activation function. In this work, we choose ELU (Clevert et al., 2015), a non-linear $\mathcal{C}^1$ (continuous and differentiable) activation function with a Lipschitz bound of 1. Although ReLU is a Lipschitz function with $K = 1$, it is not everywhere differentiable and injective. Following Bachlechner et al. (2020), we adopt ReZero (residual with zero initialization) to enforce dynamical isometry and a stable convergence.

Lemma 3 (Residual contractions are injective). $f : \boldsymbol{X} \rightarrow \boldsymbol{X} + \frac{\alpha_l}{L} F(\boldsymbol{X})$ is one-to-one.

Injecting randomness in Transformers. In Section 1, we argued how entropy plays a pivotal role in how human brain functions. Motivated by this neuroscientific finding, we artificially inject randomness into `TransJect` to learn complex representations spanning across high and low-entropic regions. Typically, Transformer injects randomness using multi-headed self-attention, where each head aims to learn a different representation. However, Transformer learns this randomness implicitly, where the learning is managed by the training objective. We infuse explicit randomness at the neuron level. As opposed to Transformer (or any other variants of it), our model enforces each neuron (hidden state) to attain to different subspaces to learn complex and more generalized representations. For this randomized variation, named **`Entropic-TransJect`**, we use a randomized variation of Equation 3 to compute hidden representation at each layer l. For each neuron i, we draw a random sample u $\sim U(0, 1)$ and compute,

$$\boldsymbol{X}^{(l)}_{:,i} = \begin{cases} 1.5\boldsymbol{X}^{(l-1)}_{:,i} + \frac{\alpha_l}{L} F(\boldsymbol{X}^{(l-1)})_{:,i}, & \text{if u} \leq 0.25 \\ 1.5\boldsymbol{X}^{(l-1)}_{:,i} - \frac{\alpha_l}{L} F(\boldsymbol{X}^{(l-1)})_{:,i}, & \text{if } 0.25 < \text{u} \leq 0.50 \\ 4\frac{\alpha_l}{L} F(\boldsymbol{X}^{(l-1)})_{:,i}, & \text{if } 0.5 < \text{u} \leq 0.75 \\ \boldsymbol{X}^{(l-1)}_{:,i}, & \text{if } 0.75 < \text{u} \leq 1 \end{cases} \tag{4}$$

[1] All the proofs presented in the paper are supplied in Appendix A.

It is easy to verify that $\mathbb{E}_{\mathsf{u}\sim U(0,1)}[\boldsymbol{X}^{(l)}_{:,i}] = \boldsymbol{X}^{(l-1)}_{:,i} + \frac{\alpha_l}{L}F(\boldsymbol{X}^{(l-1)})_{:,i}$. Also, notice that each neuron performs same computation for all the tokens, without disturbing the distance between them.

To maintain the dimensionality, Transformer projects the representations to a lower-dimensional space, which reduces the total number of synapses among the neurons by a factor of H, the number of heads. As opposed to this, we devise a **M**ixture **o**f **E**xpert (MOE) attention motivated by Shazeer et al. (2017). With this, we compute $\boldsymbol{X}^{(l,e)}$ for each expert $e \in \{1, 2, \cdots, E\}$ in each layer l using Equation 3, learnable expert weights λ_is, and use a convex combination of them to compute,

$$\boldsymbol{X}^{(l)} = \sum_{e=1}^{E} \lambda_e \boldsymbol{X}^{(l,e)} \qquad s.t. \sum_{e=1}^{E} \lambda_e = 1 \tag{5}$$

Corollary 2 (Injectivity of MOE). The mapping function defined in Equation 3.2 is injective.

3.3 Orthogonal Residual FFN (ORF)

We reformulate the position-wise feed-forward networks with orthogonal parameterization. FFN layers in Transformer emulate a key-value memory (Geva et al., 2021). To preserve the layer-wise memory, we enforce Lipschitz continuity on the feed-forward sublayer. Formally, we define,

$$ORF(\boldsymbol{X}^{(l)}) = \boldsymbol{X}^{(l)} + \frac{\alpha_l}{L} ELU\Big(ELU(\boldsymbol{X}^{(l)}\boldsymbol{W}_1 + \boldsymbol{b}_1)\boldsymbol{W}_2 + \boldsymbol{b}_2\Big) \tag{6}$$

where $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are orthogonally parameterized.

Corollary 3 (Injectivity of ORF). Orthogonal residual FFNs are injective.

3.4 Injective Token Embedding

Positional encoding was introduced in Transformer for injecting the relative and absolute positional information of tokens into the self-attention layer. It leverages sinusodial representations of every position and adds to the original token embeddings to infuse the positional information. To ensure injectivity at each layer, we need to ensure that the initialization of the token embeddings is also injective *i.e.*, no two tokens should have exactly same embedding. Unfortunately, the addition operator is not injective. Therefore, we compute the initial embedding of token x_i as ${\boldsymbol{X}_i}^{(0)} = Concat(Emb(x_i), PE_{i,:})$. Here PE is defined similarly to the positional encoding proposed by Vaswani et al. (2017). Concatenation ensures the injectivity of embeddings. However, to maintain the dimensionality, we learn the initial embedding and positional encoding at a lower dimensional space, $\mathbb{R}^{\frac{d}{2}}$, where d is the hidden size in the encoder. As the initial embeddings of all tokens in the sequence are all unique, $\boldsymbol{X}^T\boldsymbol{X}$ is a full-rank matrix, hence ensuring injectivity throughout all the subsequent layers.

We define the final encoder mapping for each sublayer l as a composite mapping defined by,

$$SubLayer^{(l)}(\boldsymbol{X}^{(l-1)}) = ORF \circ MOE \circ IR(\boldsymbol{X}^{(l-1)}) \tag{7}$$

Theorem 2 (TransJect.) The composite map defined in Equation 7 is an injective Lipschitz with an upper bound of e^2.

We further show that the lower bound of activation for a multi-layered `TransJect` encoder with $L > 1$ is 0.25. This makes `TransJect` a *pseudo-isometry* with $k = e^2$. A bounded activation bound ensures that the incremental learning of our encoder model reduces with large depth, which theoretically vouches for a shallower model, rather than a deeper one. This suggests that `TransJect` is layer-agnostic and can perform well even with a shallower encoder. It further enforces the importance of learning better embeddings in the initial embedding layer, which essentially drives the entire encoding. The runtime complexity of our orthogonal non-normalized attention is $\mathcal{O}(nd^2)$, whereas dot-product self-attention has a runtime complexity of $\mathcal{O}(n^2d + nd^2)$. In a comparable settings where $n >> d$, `TransJect` should have a lower runtime complexity than Transformer. However, as Transformer projects the token embeddings onto a lower-dimensional space in multi-headed self-attention, `TransJect` and its variants currently exhibit higher computation time. Similar analysis on number of parameters is furnished in Appendix B.

3.5 TransJect Ablations

Table 1: Comparison of TransJect and its variants. Here *Approx.* denotes approximated eigenvalues by minimizing reconstruction loss.

Model	Eigenvalue	Residual
TransJect	Approx.	IR
Entropic-TransJect	Approx.	IR + Random
Random-TransJect	Random	IR
Random-Entropic-TransJect	Random	IR + Random

We explore total four variants of TransJect. We summarize these ablations in Table 1. However, it is important to notice that the variants of TransJect do not introduce any additional learnable parameters. The variants differ in terms of only the intermediate representation computation. Secondly, the entropic-variants of TransJect do not satisfy Theorem 3.4 as the maximum activation bound of the entropic-residual at each layer is > 2. Hence, for an L-layered encoder, the composite activation function is unbounded. However, these variants still ensure injectivity of representations.

3.6 Evaluations

We evaluate TransJect and its variants on four short and long text classification tasks. In all these classification tasks, we keep the configuration of our models with $L = 6$, $E = 4$, and $d = 512$. For all of these tasks, we use the original train-test split, where the training data is used only for model training, and the test data is used for evaluating TransJect and the other baselines. We furnish all these details in Appendix B. We evaluate all the models in terms of test accuracy.

3.6.1 Short sequence classification

We choose the **IMDb** movie review sentiment classification (Maas et al., 2011) and the **AGnews** topic classification (Zhang et al., 2015) datasets for short text classification – the former one is a binary classification task, whereas the latter one contains four classes. For IMDb and AGnews classifications, we choose a maximum text length of 512 and 256, respectively.

Table 2: Text classification accuracy on IMDb and Agnews datasets. Results for models highlighted with † are taken from Dutta et al. (2021). For TransJect and its variants, we report the mean and s.d. of test accuracies obtained from three different runs with different seed initializations.

Model	IMDb	AGnews
Transformer (Vaswani et al., 2017)†	81.3	88.8
Transformer+ReZero (Bachlechner et al., 2020)	83.4	89.6
Linformer (Wang et al., 2020)†	82.8	86.5
Synthesizer (Tay et al., 2021)†	84.6	89.1
TransEvolve (Dutta et al., 2021)†	87.6	**91.1**
TransJect	**88.1** ± 0.01	88.8 ± 0.00
Entropic-TransJect	87.5 ± 0.00	89.1 ± 0.00
Random-TransJect	86.5 ± 0.00	90.2 ± 0.00
Random-Entropic-TransJect	86.5 ± 0.02	90.8 ± 0.00

For these two tasks, we use a mean pooling on the hidden representation obtained by the final encoder layer before passing it to the final classification layer. We utilize the BERT pre-trained tokenizer to tokenize the texts for this two tasks [2].

Table 2 shows the performance of the competing models. On IMDb classification, TransJect outperforms Transformer with a whooping 6.8% margin. TransJect achieves 0.5% better accuracy than TransEvolve, the best baseline. On the AGnews topic classification task, TransJect and its variants fall short of 0.3% from the best baseline, TransEvolve. However, TransJect with randomly initialized eigenvalues achieves 1% better accuracy than Transformer and Synthesizer. Albeit having a high-entropic component, the entropic-variants of TransJect display very high robustness in terms of final predictions, with a nearly zero average standard deviation. Another interesting observation on the AGnews classification task is the superiority of randomly-initialized eigenvalues over actual eigenvalues, whereby injecting randomness through randomly initialized eigenvalues aids in 2% performance improvement. Limited contextual information can create difficulty in reconstructing the $\boldsymbol{X}^T\boldsymbol{X}$ from the approximate eigenvalues. Therefore, having randomly-initialized eigenvalues can aid in learning better context when the context itself is limited. Similarly, on AGnews classification, entropic-variants exhibit better performance than non-entropic variants.

We further evaluate Transformer with zero residual initialization (Transformer+ReZero) model to understand the weighted residual's effect in achieving dynamical isometry. For both IMDb and AGnews classifications, Transformer+ReZero achieves $\sim 1.5\%$ better performance than vanilla

[2] https://huggingface.co/bert-base-uncased

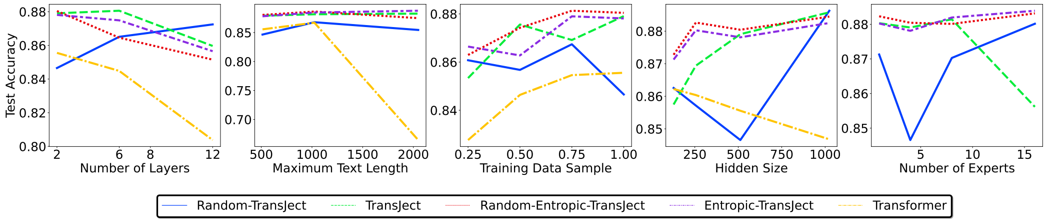


Figure 3: Performance of `TransJect` and Transformer under different configurations for the IMDb classification task.

Transformer. However, `TransJect` consistently outperforms Transformer+ReZero, showing that dynamical isometry may not be sufficient for better generalization.

3.6.2 Long sequence classification

To further highlight the effectiveness of `TransJect` on longer sequences, we evaluate our model on the **ListOps** dataset (Nangia & Bowman, 2018), containing input sequences of arithmetic operations and operands with the predicted output being a single digit between 0 to 9. For this task, we choose a maximum sequence length of 1024. As opposed to the other classification tasks, for ListOps classification, we use max pooling on the hidden representation extracted by the final encoder layer, followed by the final multi-class classification layer. Another task we choose under long text classification is the character-level IMDb (CharIMDb) review sentiment classification. For CharIMDb classification, we use a maximum sequence length of 1024.

Table 3: Classification accuracy on long range sequence classification for the ListOps and character-level IMDb datasets. Results of models highlighted with † are taken from Dutta et al. (2021). For `TransJect` and its variants, we report the mean and s.d. of test accuracies obtained from three different runs with different seed initializations.

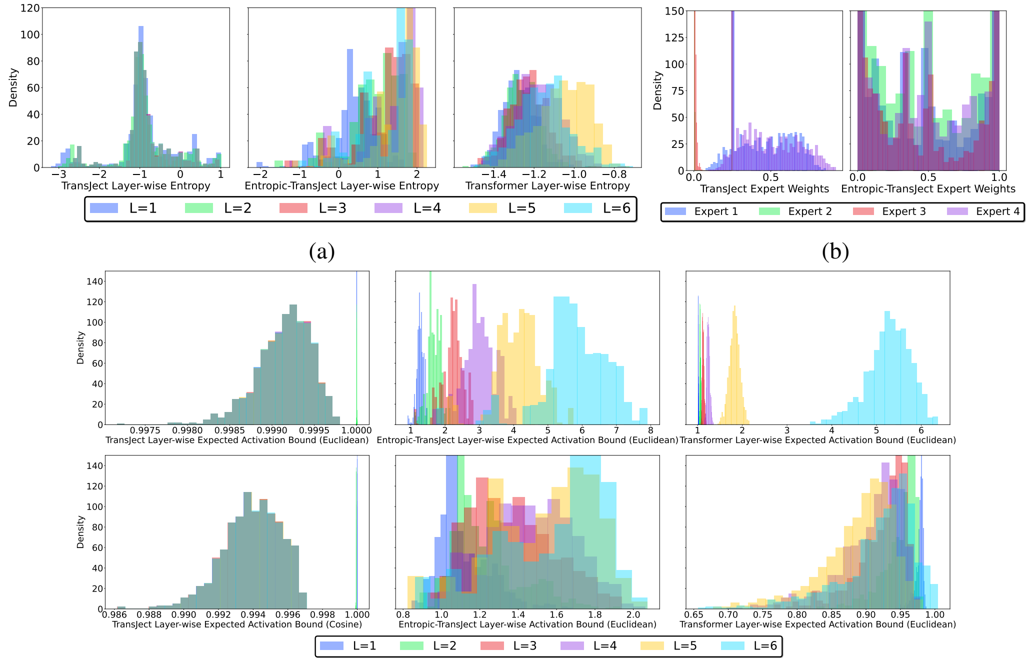
Model	ListOps	CharIMDb
Transformer (Vaswani et al., 2017)†	36.4	64.3
Transformer+ReZero (Bachlechner et al., 2020)	38.9	65.6
Linformer (Wang et al., 2020)†	35.7	53.9
Synthesizer (Tay et al., 2021)†	37.0	61.7
Reformer (Kitaev et al., 2020)†	37.3	56.1
Sinkhorn (Tay et al., 2020)†	17.1	63.6
Big Bird (Zaheer et al., 2020)†	36.0	64.0
Linear Attention (Katharopoulos et al., 2020)†	16.1	65.9
Performers (Choromanski et al., 2021)†	18.0	65.4
Random Feature Attention (Peng et al., 2020)†	36.8	66.0
TransEvolve (Dutta et al., 2021)†	43.2	66.1
`TransJect`	**48.3** ± 0.00	**68.8** ± 0.01
`Entropic-TransJect`	40.7 ± 0.02	64.7 ± 0.03
`Random-TransJect`	43.9 ± 0.00	**68.8** ± 0.03
`Random-Entropic-TransJect`	39.5 ± 0.01	65.6 ± 0.01

We evaluate `TransJect` and its variants against Transformer along with several of its recent variants and report the test accuracy in Table 3. As opposed to short sequence classification tasks, for long sequences, `TransJect` turns out to be very effective and consistently outperforms all its own variants as well as the baselines. It achieves 2.7% and 5.1% better accuracy on CharIMDb classification and ListOps classification, respectively, than the best baseline TransEvolve.

Another interesting observation is the superiority of non-entropic models for long sequence classification. In both the datasets, we observe that both `TransJect` and `Random-TransJect` outperform all other baselines as well as the entropic variants of `TransJect`. The margin between random and non-random variants is 4.8% on CharIMDb and ListOps classification tasks, on average. This certainly encourages non-entropic representations for capturing longer contextual information.

3.7 Robustness of `TransJect` under different configurations

We evaluate `TransJect` under different hyperparameter settings to understand its robustness under different configurations. For this purpose, we choose IMDb classification and highlight the test accuracy in Figure 3. Similar analysis is conducted on the CharIMDb classification task, which we report in Appendix C. For IMDb classification, we observe that shallower `TransJect` is preferred over deeper model; however, the standard deviation is a meagre 1.7% and 0.8%, respectively, as compared to 2.7% and 6.6% of Transformer. Similarly, Transformer is heavily influenced under different sequence length, whereas both `TransJect` and `Entropic-TransJect` are very robust and achieve very similar performance irrespective of the sequence length. With large hidden size,



(a)

(b)

(c)

Figure 5: Distribution of (a) layer-wise entropy of token representations, (b) expert weights, (c) layer-wise activation bounds.

Transformer tends to overfit, whereas TransJect remains stable even under varying hidden sizes. Another interesting observation is the generalization capability of our model and its variants. Even with 50% of the training data, TransJect and its entropic-variant can attain to the best test results.

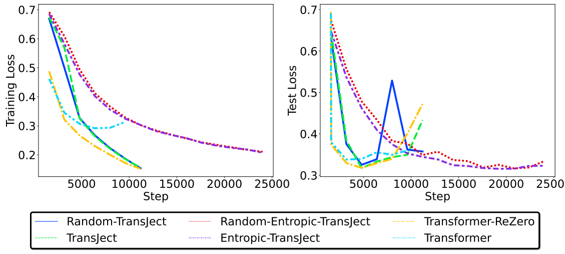


Figure 4: Training and test loss obtained by TransJect, Transformer and Transformer+ReZero for the IMDb classification task.

We further compare the stepwise training and test loss in Figure 4. For IMDb classification, Transformer starts diverging after 7000 training steps. With zero redisual initialization, however, it convergences faster, similar to TransJect; however, it starts overfitting the training data after 5000 steps. On the contrary, Entropic-TransJect learns slowly and eventually generalizes better on the test data with overcoming ReZero after 18000 steps.

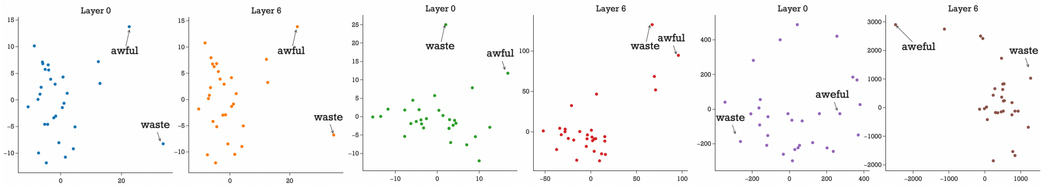
4 Analysis

We conduct a detailed statistical analysis of our model on the IMDb classification task.

Entropy states of hidden representations. We compare the *differential entropy* of the hidden representations learned by TransJect at each layer to understand how the energy state changes over the layers. We calculate differential entropy as,

$$\textit{entropy}^{(l)}\left(X^{(l)}\right) = \mathbb{E}_{b,j,h}\left[-\log X^{(l)}_{b,j,h}\right] = -\mathbb{E}_{b,j}\left[\int_{\mathbb{H}} X_{b,j,h}\log X_{b,j,h}\, d\boldsymbol{h}\right] \tag{8}$$

We highlight the distribution of entropy of token embeddings at every layer of TransJect, Entropic-TransJect and Transformer in Figure 5a. For all of these analyses, we use the outputs inferred by our models on test data. As argued in Section 1, Transformer usually embeds onto a low-entropic region where each hidden state learns similar representations. Entropy distribution displayed by this model follows a normal distribution within the range of $(-1.5, -0.7)$, where the negative values suggest low-entropic states, and low inter-quartile range suggests low variability. On the other hand, TransJect follows a multi-modal distribution where the entropy lies between



(a) `TransJect` (b) `Entropic-TransJect` (c) Transformer

Figure 6: Isomap plot of layer-wise token embeddings learned by (a) `TransJect`, (b) `Entropic-TransJect`, and (c) Transformer on the text "*The characters are unlikeable and the script is awful. It's a waste of the talents of Deneuve and Auteuil.*"

$(-3, 1)$. The highest entropic behavior is shown by `Entropic-TransJect`, where the entropy distribution is left-skewed with more positive entropy values. Further, entropy does not depend on the encoder layer in `TransJect`, suggesting that the neurons learn different representations with different entropic state, irrespective of the layer. On the contrary, in Transformer, higher layers have higher entropy.

Effectiveness of expert model. We observe the individual importance of experts and how they interplay in the mixture model. We illustrate the expert weight distribution in Figure 5b, which confirms that the experts are well-balanced, and an enforced load balancing is not required for our model. Further, we calculate the entropy of each expert representation to understand how it constituents the final representation at every layer. The expert entropy values of `TransJect` and `Entropic-TransJect` are -2.2 and -1.56, respectively. Computing an equivalent entropy of different attention heads for Transformer gives us an entropy of 0.67. This suggests that at the individual level, the experts learn similar projection functions; however, at the neuron level, they learn differently. As opposed to this, Transformer enforces all the attention heads to learn embeddings from different subspaces, leading to a high entropy at the level of attention head.

Preserving distances between tokens. Continuing our initial discussion on preserving layer-wise distances between tokens, we calculate the distribution activation bounds at different encoding layers in Figure 5c. The empirical activation bound of `TransJect` turns out to be < 1, which is much lesser than the theoretical bound of $e^2 \approx 7.39$. Although we do not claim for Lipschitz continuity in the dot-product space, the empirical activation bound under dot-product is also ≈ 1. As opposed to `TransJect`, both Transformer and the high entropic variant of `TransJect` have much higher empirical activation bounds. Interestingly, both Transformer and `Entropic-TransJect` aim to preserve the semantic similarity at the later layers, at the expense of distance; however, `TransJect` can preserve both of them with a tighter bound, leading to a more robust representations for each token. Figure 6 shows representations obtained on tokens of a sample text at different encoder layers, projected onto $2-D$. We use isometric mapping (Tenenbaum et al., 2000) for projecting the high dimensional vectors to the $2-$dimensional space. `TransJect` maintains the initial embedding space throughout the layers, showing the robustness in learning initial embeddings. On the other hand, `Entropic-TransJect` and Transformer expand the projection subspaces to a more sparse subspaces, even though they project semantically similar tokens closer.

5 Conclusion

In this work, we introduced a new learning paradigm in language understanding by enforcing a distance-preserving criterion in the multi-layered encoder models. We proposed `TransJect` and its entropic-variants, motivated by neurological and cognitive sciences findings. We derived that by enforcing orthogonal parameterization and utilizing smooth activation maps, Transformers can preserve layer-wise information propagation within a theoretical bound, allowing the models to generalize well, even at a smaller depth. We further argued against implicit regularization techniques like dropout, and rather preferred explicit randomness for better generalization. Our empirical analyses suggested a superior performance of `TransJect` and its variants over other Transformer and self-attention-based baselines. Additionally, we observed that the non-entropic models tend to perform well on long sequence classification, whereas entropic models tend to perform better for short sequence classification in which the context is limited. These findings encourage us to explore the natural laws of science for building better, intuitive and more cognitive artificial intelligence.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Huanru Henry Mao, Garrison W. Cottrell, and Julian McAuley. ReZero is All You Need: Fast Convergence at Large Depth, June 2020. URL `http://arxiv.org/abs/2003.04887`. Number: arXiv:2003.04887 arXiv:2003.04887 [cs, stat].

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The Long-Document Transformer, December 2020. URL `http://arxiv.org/abs/2004.05150`. Number: arXiv:2004.05150 arXiv:2004.05150 [cs].

Graham Brightwell, Claire Kenyon, and Hélène Paugam-Moisy. Multilayer neural networks: one or two hidden layers? *Advances in Neural Information Processing Systems*, 9, 1996.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking Attention with Performers, March 2021. URL `http://arxiv.org/abs/2009.14794`. Number: arXiv:2009.14794 arXiv:2009.14794 [cs, stat].

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Guillem Collell and Jordi Fauquet. Brain activity and cognition: a connection from thermodynamics and information theory. *Frontiers in psychology*, 6:818, 2015.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Ursula Dicke and Gerhard Roth. Neuronal factors determining high intelligence. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 371(1685):20150180, 2016.

Subhabrata Dutta, Tanya Gautam, Soumen Chakrabarti, and Tanmoy Chakraborty. Redesigning the transformer architecture with insights from multi-particle dynamical systems. *Advances in Neural Information Processing Systems*, 34:5531–5544, 2021.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention, August 2020. URL `http://arxiv.org/abs/2006.16236`. Number: arXiv:2006.16236 arXiv:2006.16236 [cs, stat].

Soheil Keshmiri. Entropy and the brain: An overview. *Entropy*, 22(9):917, 2020.

Hyunjik Kim, George Papamakarios, and Andriy Mnih. The lipschitz constant of self-attention. In *International Conference on Machine Learning*, pp. 5562–5571. PMLR, 2021.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The Efficient Transformer, February 2020. URL `http://arxiv.org/abs/2001.04451`. Number: arXiv:2001.04451 arXiv:2001.04451 [cs, stat].

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pp. 142–150, 2011.

Nikita Nangia and Samuel R Bowman. Listops: A diagnostic dataset for latent tree learning. *arXiv preprint arXiv:1804.06028*, 2018.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2020.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29, 2016.

Ofir Press, Noah A. Smith, and Mike Lewis. Shortformer: Better Language Modeling using Shorter Inputs, June 2021. URL `http://arxiv.org/abs/2012.15832`. Number: arXiv:2012.15832 arXiv:2012.15832 [cs].

Glenn N Saxe, Daniel Calderone, and Leah J Morales. Brain entropy and human intelligence: A resting-state fmri study. *PloS one*, 13(2):e0191582, 2018.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse sinkhorn attention. In *International Conference on Machine Learning*, pp. 9438–9447. PMLR, 2020.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. Synthesizer: Rethinking self-attention for transformer models. In *International conference on machine learning*, pp. 10183–10192. PMLR, 2021.

Joshua B Tenenbaum, Vin de Silva, and John C Langford. A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323, 2000.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, December 2017. URL `http://arxiv.org/abs/1706.03762`. Number: arXiv:1706.03762 arXiv:1706.03762 [cs].

Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020.

Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel S Schoenholz, and Jeffrey Pennington. Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks. pp. 10.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

A Theoretical Results

We furnish the proofs of all the theoretical results presented in the main text.

A.1 Proof of Lemma 2

$$K_M = \sup_{x \neq 0} \frac{||\boldsymbol{M}\boldsymbol{x}||_2}{||\boldsymbol{x}||_2} = \sup_{||\boldsymbol{x}||_2 = 1} ||\boldsymbol{M}\boldsymbol{x}||_2.$$

Squaring both the side, we decompose $\boldsymbol{M}$ as $\boldsymbol{U\Sigma V}^T$, where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal matrices, and $\boldsymbol{\Sigma}$ is the diagonal matrix containing the singular values (square root of eigenvalues of $\boldsymbol{M}^T\boldsymbol{M}$), $\boldsymbol{\Sigma}_1 \geq \boldsymbol{\Sigma}_2 \geq \boldsymbol{\Sigma}_3 \cdots \geq 0$. For an orthogonal matrix $\boldsymbol{U}$, $||\boldsymbol{U}\boldsymbol{x}||_2{}^2 = ||\boldsymbol{x}^T\boldsymbol{U}^T U\boldsymbol{x}||_2 = ||\boldsymbol{x}^T\boldsymbol{x}||_2 = ||\boldsymbol{x}||_2^2$. This leads to

$$||\boldsymbol{M}\boldsymbol{x}||_2{}^2 = ||\boldsymbol{U\Sigma V}^T\boldsymbol{x}||_2{}^2 = ||\boldsymbol{\Sigma}\boldsymbol{x}||_2{}^2 = \sum_{i=1}^{n} \boldsymbol{\Sigma}_i^2 x_i^2.$$

Hence,

$$K_M^2 = \sup_{\sum_{i=1}^n x_i^2 = 1} \sum_{i=1}^{n} \boldsymbol{\Sigma}_i^2 x_i^2.$$

Being a convex sum, $K_M^2 \leq \boldsymbol{\Sigma}_1^2$. Hence, $K_M \leq |\boldsymbol{\Sigma}_1|$, which completes the proof. ■

A.2 Proof of Corollary 2

For any square stochastic matrix $\boldsymbol{M} \in \mathbb{R}^{n\times n}$,

$$\boldsymbol{MI} = \begin{pmatrix} \sum_{j=1}^{n} \boldsymbol{M}_{j,1} \\ \sum_{j=1}^{n} \boldsymbol{M}_{j,2} \\ \vdots \\ \sum_{j=1}^{n} \boldsymbol{M}_{j,n} \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} = \boldsymbol{I}$$

Hence, 1 is an eigenvalue of M. Next we prove that 1 is the largest eigenvalue of M. For any eigenvalue λ and its corresponding eigenvector $\boldsymbol{v}$, $\lambda\boldsymbol{v} = \boldsymbol{Mv}$. Without loss of generality, we assume $\arg\max_i |v_i| = 1$. Hence, for the 1st entry in this column vector $\lambda v_1 = \sum_{j=1}^{n} \boldsymbol{M}_{1,j} v_j$. Using triangle inequality we get,

$$|\lambda||v_1| \leq |\lambda v_1| = |\sum_{j=1}^{n} \boldsymbol{M}_{1,j} v_j| \leq \sum_{j=1}^{n} |\boldsymbol{M}_{1,j}||v_1| \leq |v_1|.$$

Hence, for any eigenvalue $|\lambda| \leq 1$. Hence, it proves our corollary that the largest eigenvalue is 1. ■

A.3 Proof of Lemma 2

For any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, we define $g : [0, 1] \to \mathbb{R}^m$ as

$$g(\boldsymbol{t}) = f(\boldsymbol{x} + \boldsymbol{t}(\boldsymbol{y} - \boldsymbol{x})). \tag{9}$$

It is easy to verify that $g(0) = f(\boldsymbol{x})$ and $g(1) = f(\boldsymbol{y})$.

$$f(\boldsymbol{y}) - f(\boldsymbol{x}) = g(1) - g(0) = \int_0^1 \nabla_{\boldsymbol{t}} g(\boldsymbol{t}) d\boldsymbol{t} \tag{10}$$

Using chain rule of differentiation on Equation A.3 we get,

$$\nabla_{\boldsymbol{t}} g(\boldsymbol{t}) = \nabla_{\boldsymbol{t}} f\Big(x + \boldsymbol{t}(y - x)\Big)(y - x)$$

Using this in Equation A.3 we get

$$||f(\boldsymbol{y}) - f(\boldsymbol{x})|| = \Big|\Big| \int_0^1 \nabla_{\boldsymbol{t}} f\Big(\boldsymbol{x} + \boldsymbol{t}(\boldsymbol{y} - \boldsymbol{x})\Big)(\boldsymbol{y} - \boldsymbol{x}) d\boldsymbol{t}\Big|\Big| \leq \int_0^1 \Big|\Big|\nabla_{\boldsymbol{t}} f\Big(\boldsymbol{x} + \boldsymbol{t}(\boldsymbol{y} - \boldsymbol{x})\Big)(\boldsymbol{y} - \boldsymbol{x}) d\boldsymbol{t}\Big|\Big|$$

As f is $\mathcal{C}^1$, supremum of its derivative exists, which allow us to set $sup_{\boldsymbol{t}}\Big|\Big|\nabla_{\boldsymbol{t}} f\Big(\boldsymbol{x}+\boldsymbol{t}(\boldsymbol{y}-\boldsymbol{x})\Big)\Big|\Big| = K$ and deduce

$$||f(\boldsymbol{y}) - f(\boldsymbol{x})|| \leq K||\boldsymbol{y} - \boldsymbol{x}|| \int_0^1 d\boldsymbol{t} = K||\boldsymbol{y} - \boldsymbol{x}||. \quad ■$$

A.4 Proof of Theorem 3.1

We can compute non-normalized attention as $Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = (\boldsymbol{QK}^T)\boldsymbol{V} = (\boldsymbol{XW}^Q\boldsymbol{W}^{K^T}\boldsymbol{X}^T)\boldsymbol{XW}^V$. Being a real symmetric matrix, $\boldsymbol{X}^T\boldsymbol{X}$ can be decomposed into $\tilde{\boldsymbol{Q}}\boldsymbol{\Sigma}\tilde{\boldsymbol{Q}}^T$, which leads to

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{X}\underbrace{\boldsymbol{W}^Q\boldsymbol{W}^{K^T}\tilde{\boldsymbol{Q}}}_{\text{orthogonal}}\underbrace{\boldsymbol{\Sigma}}_{\text{diagonal}}\underbrace{\tilde{\boldsymbol{Q}}^T\boldsymbol{W}^V}_{\text{orthogonal}}. \tag{11}$$

As the product of two orthogonal matrices is orthogonal, Equation A.4 reduces to

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \boldsymbol{XU\Sigma V}. \tag{12}$$

with a suitable set of learnable orthogonal matrices $\boldsymbol{U}$ and $\boldsymbol{V}$. ■

A.5 Proof of Lemma 3.2

Let us assume $\exists \boldsymbol{x} \neq \boldsymbol{y}$ such that $f(\boldsymbol{x}) = f(\boldsymbol{y})$, which implies $||f(\boldsymbol{x}) - f(\boldsymbol{y})|| = 0$. Using triangle inequality and Equation 3 we get

$$\begin{aligned} ||\boldsymbol{x}-\boldsymbol{y}|| - |\frac{\alpha_l}{L}| \cdot ||F(\boldsymbol{x}) - F(\boldsymbol{y})|| \leq ||f(\boldsymbol{x}) - f(\boldsymbol{y})|| = 0 \\ \iff ||\boldsymbol{x}-\boldsymbol{y}|| \leq |\frac{\alpha_l}{L}| \cdot ||F(\boldsymbol{x}) - F(\boldsymbol{y})||. \end{aligned}$$

Using Lemma 2 we can derive the Lipschitz bound for F as 1 (derivative of ELU activation is bounded by 1). Using this we get

$$||\boldsymbol{x}-\boldsymbol{y}|| \leq |\frac{\alpha_l}{L}| \cdot ||\boldsymbol{x}-\boldsymbol{y}||.$$

Which contradicts that fact that $|\frac{\alpha_l}{L}| < 1$. Hence, we proof the lemma by contradiction. ■

A.6 Proof of Corollary 3.2 and Corollary 3.3

Being a convex sum of residual contractions, it follows directly from A.5. Similarly, Corollary 3.3 also follows directly. ■

A.7 Proof of Theorem 3.4

Being a composition of continuously differentiable injective functions, $f = ORF \circ MOE \circ IR$ is a continuously differentiable injective function.

Next we compute the Lipschitz bound for f. For the sake of simplicity, let us assume $\frac{\alpha_1}{L} = \frac{\alpha_2}{L} \cdots = \frac{\alpha_l}{L} = \alpha < \frac{1}{L}$. We first expand f as

$$f(\boldsymbol{x}) = \sum_{e=1}^{E} \lambda_e \boldsymbol{x} + \sum_{e=1}^{E} \lambda_e \alpha \cdot ELU(\boldsymbol{x}\boldsymbol{U}^e \Sigma \boldsymbol{V}^e) + \alpha \cdot ELU(\overline{\boldsymbol{x}})$$

$$\overline{\boldsymbol{x}} = ELU\Big(\sum_{e=1}^{E} \lambda_e \boldsymbol{x}\boldsymbol{W}_1 + \sum_{e=1}^{E} \lambda_e \alpha \cdot ELU(\boldsymbol{x}\boldsymbol{U}^e \Sigma \boldsymbol{V}^e)\boldsymbol{W}_1 + \boldsymbol{b}_1\Big)\boldsymbol{W}_2 + \boldsymbol{b}_2$$

$$\implies \overline{\boldsymbol{x}} = ELU\Big(\boldsymbol{x}\boldsymbol{W}_1 + \sum_{e=1}^{E} \lambda_e \alpha \cdot ELU(\boldsymbol{x}\boldsymbol{U}^e \Sigma \boldsymbol{V}^e)\boldsymbol{W}_1 + \boldsymbol{b}_1\Big)\boldsymbol{W}_2 + \boldsymbol{b}_2 \tag{13}$$

We break down the expression and compute Lipschitz bound of $ELU(x\boldsymbol{U}\Sigma\boldsymbol{V})\boldsymbol{W}_1 + \boldsymbol{b}_1$ first. Using the Lipschitz continuity of ELU, we get

$$\begin{aligned} & ||ELU(\boldsymbol{x}\boldsymbol{U}\Sigma\boldsymbol{V})\boldsymbol{W}_1 + \boldsymbol{b}_1 - ELU(\boldsymbol{y}\boldsymbol{U}\Sigma\boldsymbol{V})\boldsymbol{W}_1 - \boldsymbol{b}_1|| \\ & = ||ELU(\boldsymbol{x}\boldsymbol{U}\Sigma\boldsymbol{V})\boldsymbol{W}_1 - ELU(\boldsymbol{y}\boldsymbol{U}\Sigma\boldsymbol{V})\boldsymbol{W}_1|| \\ & \leq ||ELU(\boldsymbol{x}\boldsymbol{U}\Sigma\boldsymbol{V}) - ELU(\boldsymbol{y}\boldsymbol{U}\Sigma\boldsymbol{V})|| \cdot ||\boldsymbol{W}_1|| \\ & \leq ||ELU(\boldsymbol{x}\boldsymbol{U}\Sigma\boldsymbol{V}) - ELU(\boldsymbol{y}\boldsymbol{U}\Sigma\boldsymbol{V})|| \\ & \leq ||\boldsymbol{x}\boldsymbol{U}\Sigma\boldsymbol{V} - \boldsymbol{y}\boldsymbol{U}\Sigma\boldsymbol{V}|| \\ & \leq ||\boldsymbol{x}-\boldsymbol{y}|| \cdot ||\boldsymbol{U}|| \cdot ||\boldsymbol{\Sigma}|| \cdot ||\boldsymbol{V}|| \\ & \leq ||\boldsymbol{x}-\boldsymbol{y}||. \end{aligned}$$

Feeding this into Equation A.7 and using the fact $\sum_{e=1}^{E} \lambda_e = 1$, we get

$$\begin{aligned} & ||\overline{\boldsymbol{x}} - \overline{\boldsymbol{y}}|| \\ & \leq (||\boldsymbol{x}-\boldsymbol{y}|| \cdot ||\boldsymbol{W}_1|| + |\alpha| \cdot ||\boldsymbol{x}-\boldsymbol{y}||) \cdot ||\boldsymbol{W}_2|| \\ & \leq (1 + |\alpha|)||\boldsymbol{x}-\boldsymbol{y}||. \end{aligned}$$

Finally,

$$\begin{aligned}
&||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \\
&\leq ||\boldsymbol{x}-\boldsymbol{y}|| + |\alpha| \cdot ||\boldsymbol{x}-\boldsymbol{y}|| + |\alpha|(1+|\alpha|) \cdot ||\boldsymbol{x}-\boldsymbol{y}|| \\
&\leq (1+|\alpha|)^2 ||\boldsymbol{x}-\boldsymbol{y}|| \\
&\leq (1+\frac{\alpha}{L})^2 ||\boldsymbol{x}-\boldsymbol{y}||.
\end{aligned}$$

Here, f is a layer-wise operator. Hence, for all the L encoder layers,

$$||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq (1+\frac{\alpha}{L})^{2L} ||\boldsymbol{x}-\boldsymbol{y}||.$$

The sequence $\prod_n (1+\frac{x}{n})^n$ converges only if $x < n$, which holds in our case through the design of residual weights. Using $\lim_{n\to\infty}(1+\frac{x}{n})^n = e^x$, we get $||f(\boldsymbol{x}) - f(\boldsymbol{y})|| \leq e^2 ||\boldsymbol{x}-\boldsymbol{y}|| \quad \blacksquare$.

To compute the lower bound, we can use the triangle inequality to obtain

$$\begin{aligned}
&||\boldsymbol{x}-\boldsymbol{y}|| \cdot |1-\alpha-\alpha(1-\alpha)| \\
&\leq ||\boldsymbol{x}-\boldsymbol{y}|| \cdot |1-\frac{1}{L}-\frac{1}{L}(1-\frac{1}{L})| \\
&= ||\boldsymbol{x}-\boldsymbol{y}|| \cdot (1-\frac{1}{L})^2 \\
&\leq ||f(\boldsymbol{x}) - f(\boldsymbol{y})||.
\end{aligned}$$

For a multi-layered encoder with $L > 1$, this value is bounded by 0.25.

B Experimental Details

In this section, we describe the experimental setup we have followed to run experiments with TransJect and its variants on the four sequence classification tasks. All the experiments are run for 30 epochs. To terminate learning on plateaus, we use a early stopping based on the test loss with a patience of 4 epochs. For all the experiments, we use Adam optimizer with learning rate of 0.0005, $\beta_1 = 0.9, \beta_2 = 0.98$ and $\epsilon = 10^{-9}$. We use TransJect and all the variants with 6 encoder layers with hidden size $d = 512$ and 4 experts in each encoder layer, for all the models. We highlight the model parameters in Table 4. We use both training and test batches of size 32. One Tesla P100 and one Tesla V100 GPUs are used for conducting all the experiments.

Table 4: Comparison of number of learnable parameters. For a fair comparison we choose the number of experts $E = 8$ for TransJect. Additionally, Transformer uses $d_{ff} = 2048$ in the feedforward layer, whereas TransJect uses the original encoder hidden dimension to maintain invertibility. For a comparable configuration, TransJect contains 45% more parameters (synapses) than Transformer.

Model	**#Parameters**
Transformer (Vaswani et al., 2017)	34.5M
Transformer with $d_{ff} = 512$	25.1M
Transformer with $W^Q = W^K$	32.9M
TransEvolve (Dutta et al., 2021)	4.2M
TransJect	36.4M

C Analysis on Long sequence classification

In this section, we continue the analysis of TransJect on the long sequence classification task. Figure 7. presents the test accuracy obtained by our models in Similar to the word-level IMDb classification, TransJect shows more robust performance under different selection of depth and sequence length. On a similar note, we observe that the non-entropic variants tend to perform better with smaller training data. On the other hand, entropic models require more data for stability. At the same time, a higher hidden size boosts the performances for our models, which could be attributed to the number of synapses or inter-connections among neurons.

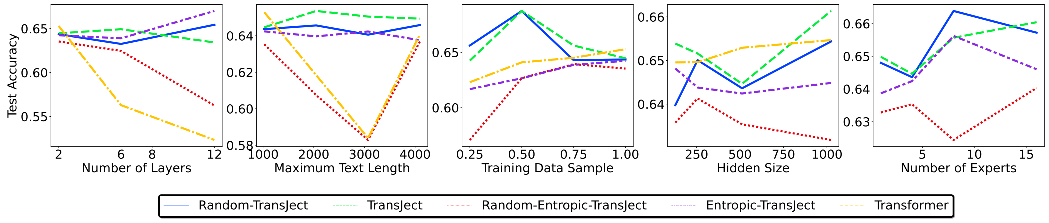
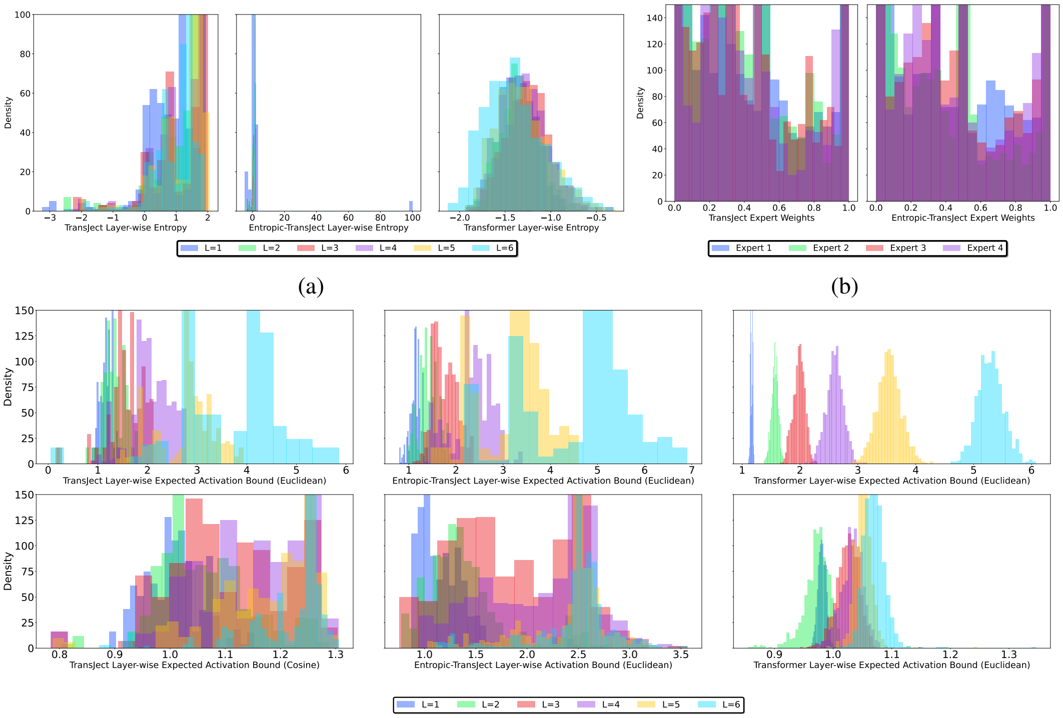


Figure 7: Performance of `TransJect` and Transformer under different configurations on the CharIMDb classification task.



(a)

(b)

(c)

Figure 9: Distribution of (a) layer-wise entropy of token representations, (b) expert weights, (c) layer-wise activation bounds for the CharIMDb classification task.

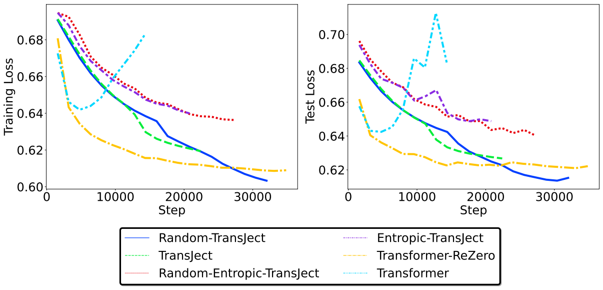


Figure 8: Training and test loss obtained by `TransJect`, Transformer and Transformer+ReZero on the CharIMDb classification task.

The convergence analysis in Figure 8 shows the robustness of our models in terms of generalization capabilities. Similar to the IMDb classification task, in the CharIMDb classification task, `TransJect` and its variants display a very stable convergence. `TransJect` with random eigenvalues generalizes better than with ReZero and even achieves lower training loss after 30000 steps. On the other hand, Transformer diverges quickly after 10000 steps.

We conduct a similar statistical analysis on CharIMDb classification. For longer sequences, `TransJect` and

Entropic-TransJect display even higher entropy, as compare to Transformer. Figure 9a suggests that Transformer follows a normal distribution over state entropies, whereas TransJect shows a left-skewed distribution, suggesting more entropic states. However, in CharIMDb task, Transformer exhibits a similar behavior in terms of activation bound. In the CharIMDb task, the activation bound for TransJect increases (though stays within the theoretical bound) due to high-entropic projections. For the CharIMDb task, the initial embeddings sit in a lower-rank space due to a smaller vocabulary. This allows Transformer to explore in a low-dimensional dense space, leading to a tighter activation bound.