Attention Approximates Sparse Distributed Memory

Anonymous Author(s)
Affiliation
Address
email

Abstract

While Attention has come to be an important mechanism in deep learning, it emerged out of a heuristic process of trial and error, providing limited intuition for why it works so well. Here, we show that Transformer Attention closely approximates Sparse Distributed Memory (SDM), a biologically plausible associative memory model, under certain conditions on the data normalization. We confirm that these conditions are satisfied in pre-trained GPT2 Transformer models. We discuss the implications of the Attention-SDM map and provide new computational and biological interpretations of Attention.

Introduction

Used most notably in the Transformer, Attention helped deep learning to arguably approach human level performance in various tasks with larger models continuing to boost performance ([1], [2] [3], [4], [5], [6], [7], [8], [9]). However, the heuristic motivations that produced Attention leave open the question of why it performs so well on many tasks humans do ([1], [10]). Insights into why Attention is so effective would not only make it more interpretable but also guide future improvements.

Much work has been done to try and explain Attention's success, including work showing that Transformer representations map more closely to human brain recordings and inductive biases than other models ([11], [12]). Our work goes further in showing the potential relationship between Attention and the brain at the level of its neuronal wiring, giving a new and exciting perspective behind Attention's success. This potential relationship to the brain is created by showing mathematically that Attention closely approximates Sparse Distributed Memory (SDM).

SDM is an associative memory model developed in 1988 to solve the "Best Match Problem", where we have a set of memories and want to quickly find the "best" match to any given query ([13], [14]). In the development of its solution, SDM respected fundamental biological constraints, such as Dale's law that synapses are fixed to be either excitatory or inhibitory and cannot dynamically switch (see Section 1 for an overview of SDM and Appendix D for a review). Despite being developed independently of neuroanatomy, SDM's biologically plausible solution maps strikingly well onto the cerebellum[1] ([13], [20]).

At a high level, the relationship between SDM and Attention exists because SDM's read operation is approximately an exponential over a sum of exponentials that closely approximates the Attention softmax rule (Section 2). Establishing that Attention approximates SDM mathematically, we then test it in pre-trained GPT2 Transformer models (Section 3). We use the Query-Key Normalized Transformer variant to directly show that the relationship to SDM holds well [21]. We then use original GPT2 models to help confirm this result and make it more general. Given these results, we

[1] This relationship is made even more compelling by the fact that cerebellum-like neuroanatomy exists in many other organisms including numerous insects (eg. the Drosophila Mushroom Body) and potentially cephalopods ([15], [16], [17], [18], [19]).

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

put forward the hypothesis that for Attention to work well it may need to exist in the same regime as that used by SDM.

Using the SDM framework, we are able to go beyond Attention and interpret the Transformer architecture as a whole, providing its heuristic structure with deeper intuition (Section 4). Motivated by this mapping between SDM and the Transformer, we then review SDM's relationship to the cerebellum and other potential brain regions (Section 5), introducing the hypothesis that Attention is performing the same associative memory operations as the brain. In related work (Section 6), we link SDM to other memory models ([22], [23]), including how SDM is a generalization of Hopfield Networks and, in turn, how our results explain and extend work relating Hopfield Networks to Attention [24]. Finally, we discuss limitations, Vector Symbolic Architectures that are compatible with enhancing SDM, and look beyond our work (Section 7) [25].

To what extent will our most successful deep learning systems converge with the brain? Some convergence seems inevitable, given that we want our models to excel at the same tasks that brains do. However, the brain and machine learning models have different optimizers, objective functions, and constraints. In this light, it is surprising that the connectome of the cerebellum, if it implements Sparse Distributed Memory, closely approximates Attention and many aspects of the whole Transformer architecture. We believe this work provides a small piece of evidence that the brain and the best deep learning models may be more convergent than we might otherwise think.

1 Review of Kanerva's SDM

Here, we present a short overview of SDM. We provide an in depth review on the motivations behind SDM and the features that make it biologically plausible in Appendices D and E.

SDM provides an algorithm for how memories are stored in and retrieved from neurons in the brain. There are three primitives that exist in an n dimensional binary vector space:

- Patterns ($\mathbf{p}$) - have two components: the pattern address $\mathbf{p}_a^\mu \in \{0,1\}^n$, is the vector representation of a memory and the pattern "pointer" $\mathbf{p}_p^\mu \in \{0,1\}^n$ is bound to the address and points to itself when autoassociative or to a different pattern address when heteroassociative. A heteroassociative example is memorizing where the pattern address for the letter a points to b, b points to c etc. For ease of analyzing SDM, we assume our pattern addresses and pointers are random. There are m patterns and they are indexed by the superscript $\mu \in \{1, \ldots, m\}$.

- Neurons ($\mathbf{x}$) - it is sufficient in proving SDM's relationship to Attention to know there are r neurons with fixed addresses $\mathbf{x}_a^\tau \in \{0,1\}^n$ that store a set of all patterns written to them. Their biologically plausible features are outlined in Appendix D. When we assume our patterns are random, we also assume our neurons are randomly distributed so that they can best cover the space of patterns. Of the 2^n possible vectors in our binary vector space, SDM is "sparse" because it assumes that $r \ll 2^n$ neurons exist in the space.

- Query ($\boldsymbol{\xi}$) - is the input to SDM, denoted $\boldsymbol{\xi} \in \{0,1\}^n$. The goal is to return the pattern pointer stored at the closest pattern address to the query. For any memory system we care about the maximum noise corruption that can be applied to our query while still having it read out the correct target pattern. An autoassociative example is wanting to recognize faces in poor lighting. The patterns stored in memory are images of faces we know and our query is a noisy representation of one of the faces. Using this query we want SDM to return the noise-free version of the face stored in memory.

SDM uses Hamming distance as the metric to determine the distance between any two vectors $d(\mathbf{a}, \mathbf{b}) := 1_n^T |\mathbf{a} - \mathbf{b}|$. The all ones vector 1_n is of n dimensions and $|\mathbf{a} - \mathbf{b}|$ takes the absolute value of the element-wise difference between the binary vectors. When the Hamming distance is between two arbitrary vectors we will sometimes use the shorthand $d_v := d(\mathbf{a}, \mathbf{b})$.

1.1 SDM Read Operation

In SDM, all patterns write their pointers $\mathbf{p}_p$ in a distributed fashion to all neuron addresses located within a Hamming distance d. This means that each neuron will store a set[2] of pattern pointers corresponding to pattern addresses within d: $\mathbf{x}_v^\tau = \{\mathbf{p}_p^\mu : d(\mathbf{p}_a^\mu, \mathbf{x}_a^\tau) < d, \forall \mu\}$. The optimal d depends upon the distribution of patterns in the vector space and what the memories are being used for (e.g. trading off the number of memories that can be stored versus the memory system's robustness to query noise). To serve as a useful reference d value, we use the equation derived in [26] for the optimal d^* that maximizes the probability a noise-free query will return its target pattern when assuming random patterns. This optimal d^* is a function of n, r and m and we will refer to it as the Signal to Noise Ratio (SNR) optimal d^*. We re-derive this equation in Appendix C.4 letting $\mathcal{B}_{\text{CDF}}^{-1}(\cdot)$ be the inverse CDF of the Binomial distribution:

$$d^* = \mathcal{B}_{\text{CDF}}^{-1}\left(\frac{1}{(2mr)^{\frac{1}{3}}}, n, \text{prob} = 1/2\right). \tag{1}$$

SDM's read operation uses the query to read stored pattern pointers from all neurons within distance d and averages them. This average is weighted because the same patterns are stored in multiple neurons being read from. The weighting will be higher for patterns with addresses nearer the query because they have written their pointers to more of the query read neurons. Geometrically, this weighting of each pattern can be interpreted as the intersection of the two Hamming circles[3] of radius d that are centered on the query and each pattern address $\mathbf{p}_a^\mu$ for all μ. An example of this circle intersection is shown in Fig. 1(a).

The number of 2^n possible neuronal locations that have both stored this pattern's pointer $\mathbf{p}_p$ *and* been read by $\boldsymbol{\xi}$ is: $|O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)|$, where $|\cdot|$ is the cardinality operator and $O_n(\boldsymbol{\xi}, d) = \{\mathbf{a} \in \{0,1\}^n : d(\boldsymbol{\xi}, \mathbf{a}) < d\}$ is the set of all possible neuronal addresses $\mathbf{a}$ within radius d of $\boldsymbol{\xi}$. Mathematically, SDM's read operation sums over each pattern's pointer, weighted by its query circle intersection:

$$\boldsymbol{\xi}^{new} = g\left(\frac{\sum_{\mathbf{p}\in P} |O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)| \mathbf{p}_p}{\sum_{\mathbf{p}\in P} |O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)|}\right), \qquad g(\mathbf{e}) = \begin{cases} 1, & \text{if } e_i > \frac{1}{2} \\ 0, & \text{else} \end{cases}. \tag{2}$$

The denominator normalizes all of the weights so they sum to 1 in the numerator and enable computing if the element-wise majority value is a 0 or 1, using the function $g(\cdot)$. Intuitively, the query to converge to the nearest "best" pattern because it will have the largest circle intersection weighting. The output of the SDM read operation is written as updating the query $\boldsymbol{\xi} \to \boldsymbol{\xi}^{new}$ so that it can (but is not required to) apply the read operation iteratively if full convergence to its "best match" pattern is desired and was not achieved with one update.

The circle intersection is calculated as a function of the Hamming threshold for the read and write operations d, the dimensionality n, and the vector distance between the query and pattern $d_v = d(\mathbf{p}_a, \boldsymbol{\xi})$. The size of the circle intersection (derived in Appendix C.1) is:

$$|O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)| = \sum_{a=n-d-\lfloor \frac{d_v}{2} \rfloor}^{n-d_v} \sum_{c=\max(0, n-d-a)}^{d_v-(n-d-a)} \left(\binom{n-d_v}{a} \cdot \binom{d_v}{c}\right) \tag{3}$$

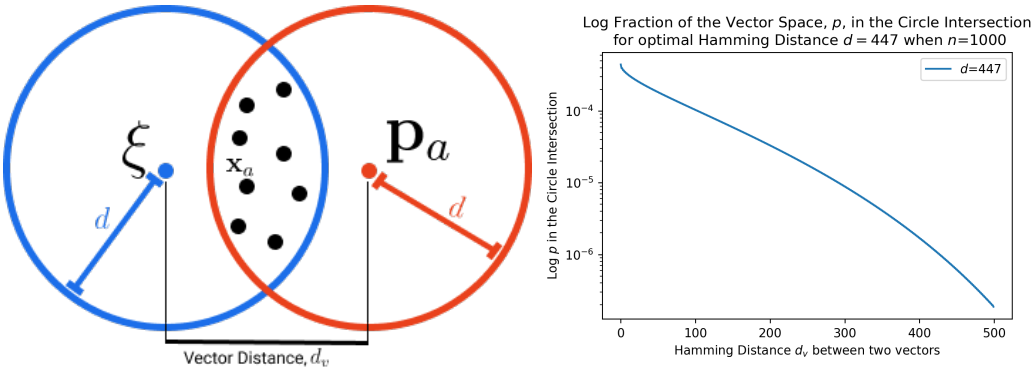
Eq. 3 is approximately exponential, which is crucial to how SDM approximates Attention. The details of this approximation are provided in Appendix C.2 but at a high level, the binomial coefficients can be represented as binomial distributions and then approximated by normal distributions that contain exponentials. With the correctly chosen constants, c_1 and c_2, that are independent of the vector distance $d(\mathbf{p}_a, \boldsymbol{\xi})$, we can make the following approximation:

$$|O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)| \approx c_1 \exp\big(-c_2 \cdot d(\mathbf{p}_a, \boldsymbol{\xi})\big) \tag{4}$$

[2] In reality, as is outlined in Appendix D, each neuron will sum over its set of patterns to create a superposition. This creates minimal noise interference between patterns because of the high dimensional nature of the vector space. However, this detail is not necessary for showing SDM's relation to Attention so we just say each neuron holds a set of patterns.

[3] In this binary space, the Hamming distance around a vector is in fact a hypercube but the vertices of an n dimensional unit cube lie on the surface of an n dimensional sphere with radius $\sqrt{n}/2$ and we refer to this as a circle because of our two dimensional diagrams. We adopt this useful analogy, taken from Kanerva's book on SDM [13], throughout the paper.

We also show the approximately exponential behavior of the exact circle intersection calculation (Eq. 3) for a representative instance of SDM in Fig. 1(b).



(a) (b)

Figure 1: **(a)** SDM's read operation using the Hamming distance d for reading and writing and the vector distance $d_v = d(\boldsymbol{\xi}, \mathbf{p}_a)$. Recall that during the write operation, pattern addresses $\mathbf{p}_a$ write their pattern pointer $\mathbf{p}_p$ to neurons located at the addresses $\mathbf{x}_a^\tau$ (denoted here as black dots) within radius d. During the read operation, the query $\boldsymbol{\xi}$ reads from each neuron within radius d, thus creating an intersection. **(b)** As d_v between the query and pattern increases (x-axis), the size of their circle intersection declines approximately exponentially (y-axis). We use Eq. 3 with $d = 447$ and $n = 1000$, while varying d_v up to the orthogonal distance of $d_v = 500$. We divide by 2^n to see what fraction of the vector space (denoted p) each circle intersection occupies and plot the y-axis on a log scale to show how, because the curve is approximately linear, the circle intersection is approximately exponential. See Appendix C.2 for a formal analysis of the exponential approximation the circle intersection creates that is robust across parameters n and d.

2 Attention Approximates SDM

To be able to handle a large number of patterns, we let the pattern address matrix be: $P_a = [\mathbf{p}_a^1, \mathbf{p}_a^2, ..., \mathbf{p}_a^m]$ with pointers $P_p = [\mathbf{p}_p^1, \mathbf{p}_p^2, ..., \mathbf{p}_p^m]$ so the patterns are columns.

The Attention update rule [1] using its original notation is:

$$\boldsymbol{\xi}^{new} = V\text{softmax}(\beta K^T Q) = (W_v Y)\text{softmax}\big(\beta (W_k Y)^T (W_q \mathbf{q})\big)$$

We can re-write this using our notation as:

$$\boldsymbol{\xi}^{new} = P_p\text{softmax}(\beta P_a^T \boldsymbol{\xi}) \tag{5}$$

where K, V, and Q symbolize the "key", "value", and "query" matrices, respectively. $\mathbf{q}$ is a single query vector and Y represents the patterns that are stored in memory. We write $K = W_k Y = P_a$ as the patterns Y are projected into the SDM vector space to become the addresses P_a by the learnt weight matrix W_k. Similarly, $V = W_v Y = P_p$ and $Q = W_q \mathbf{q} = \boldsymbol{\xi}$. The $\text{softmax}(\beta \mathbf{x}) = \exp(\beta \mathbf{x}) / \sum_{i=1}^n \exp(\beta x_i)$, where the division operation is element-wise and Attention sets $\beta = 1/\sqrt{n}$. The softmax normalizes a vector of values to sum to 1 and gives the largest values the most weight, to what extent depending on β.

To show that SDM Eq. 2 is equivalent to Attention Eq. 5 we must make SDM continuous and choose the β coefficient such that it closely approximates the almost exponential decay of SDM's circle intersection calculation. We let SDM be continuous by letting its addresses and pointers exist in the space $\mathbb{R}^n$ but be L^2 normalized. This makes the dot product of any two vectors fall in the interval $[-1, 1]$ with a value of 0 for orthogonal vectors and is also know as cosine similarity, denoted $\mathcal{C}(\mathbf{a}, \mathbf{b})$. We can relate Hamming distance to cosine similarity by the following equation:

$$d(\mathbf{a}, \mathbf{b}) = \frac{n\big(1 - \mathcal{C}(\mathbf{a}, \mathbf{b})\big)}{2}. \tag{6}$$

This relationship re-scales the cosine similarity interval, flips it from a similarity to distance metric, and linearly maps it from being continuous to the nearest integer: $[-1, 1] \rightarrow [n, 0]$. We can plug this relationship into the exponential approximation of Eq. 4 to get a continuous version that allows us to use the dot product between the pattern and query instead of Hamming distance:

$$|O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)| \approx c_1 \exp\big(-c_2 \cdot d(\mathbf{p}_a, \boldsymbol{\xi})\big) = c_1 \exp\Big(\frac{-c_2 n}{2}\big(1 - \mathcal{C}(\mathbf{p}_a, \boldsymbol{\xi})\big)\Big)$$
$$= c_3 \exp\big(\beta \mathbf{p}_a^T \boldsymbol{\xi}\big), \tag{7}$$

where c_3 encompasses the constants outside of the exponential. We replaced the remaining constants in the exponential with β that is a function of n and d. Finally, this result allows us to show the relationship between SDM and Attention:

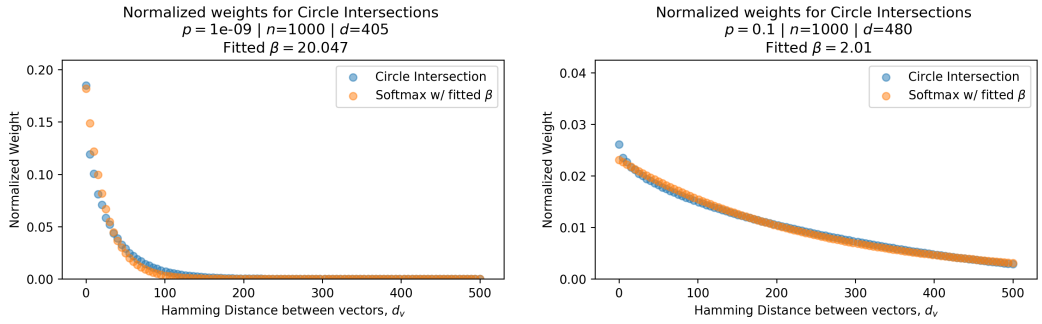
$$\boldsymbol{\xi}^{new} = g\left(\frac{\sum_{\mathbf{p} \in P} |O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)| \mathbf{p}_p}{\sum_{\mathbf{p} \in P} |O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)|}\right) \approx \frac{\sum_{\mathbf{p} \in P} \exp(\beta \mathbf{p}_a^T \boldsymbol{\xi}) \mathbf{p}_p}{\sum_{\mathbf{p} \in P} \exp(\beta \mathbf{p}_a^T \boldsymbol{\xi})} = P_p \text{softmax}(\beta P_a^T \boldsymbol{\xi}). \tag{8}$$

Recall that $g(\cdot)$ is the elementwise majority function that ensures our output is a valid binary vector. In the continuous setting, this is analogous to L^2 normalizing inputs to the Attention update. One way to interpret this mapping from discrete SDM to continuous Attention is as a continuous relaxation. A given instantiation of SDM, using a particular n and d, corresponds to Attention with a specific β coefficient.

In Fig. 2, we show that if the β coefficient is learnt to approximate the circle intersection the relationship can be tight. To emphasize the quality of the approximation across vector distances d_v, we generate a query vector and position 100 vectors at equally spaced distances between a distance of 0 (equal to the query) up to the orthogonal distance of 500, when $n = 1,000$. We perform this analysis using $d = 405$ and $d = 480$, which are significantly smaller and larger than the SNR optimal d^* value[4] to further emphasize the robustness of the approximation. To fit β we convert the Hamming distances into cosine similarity using Eq. 6 and use a univariate log linear regression that has a closed form solution:

$$\log\Big(|O_n(\mathbf{p}_a, d) \cap O_n(\boldsymbol{\xi}, d)|\Big) \approx \log(c_3) + \beta(\mathbf{p}_a^T \boldsymbol{\xi}). \tag{9}$$

In formally bounding the circle intersection by exponential equations in Appendix C.2, we also derive approximate values for β that match those fitted by the regression quite well.



(a) (b)

Figure 2: Representative plots comparing the softmax approximation to the actual circle intersection for $n = 1000$, $r = 1000000$, $m = 100$ and in **(a)** $d = 405$, which corresponds to the Hamming distance reading and writing to $p = 1e-09$ of the vector space and in **(b)** $d = 480$, $p = 0.1$. These results hold well for other values of n and d. The blue dots are the true circle intersection, the orange dots are the softmax approximation with regression fitted β.

Therefore, Attention closely approximates the continuous version of our SDM update rule when using a correctly chosen β and acknowledging the use of the weight matrices W_k, W_q and W_v to

[4]The SNR optimal d^* is given as a useful reference point. When $r = 1,000,000$ and $m = 100$ as we assume here, $d^* = 454$ using Eq. 1.

learn useful projections of the vectors. To ensure all of our approximations are correct and test how well they perform with both correlated and uncorrelated patterns, we test every SDM variant in comparison to Attention in an autoassociative retrieval task on random patterns and the MNIST dataset (Appendix D.8). These variants include SDM implemented using simulated neurons and the Attention approximation with a fitted β.[5]

3 Trained Attention Converges with SDM

For many instantiations of SDM, there exists a β that can be found via the log linear regression Eq. 9 that makes Attention approximate it well. However, with some dependence on the task at hand, there are instantiations of SDM that are better than others. If Attention in the Transformer model is implementing SDM, we should expect for trained Attention to use βs that correspond to at least reasonable instances of SDM. Here, we provide empirical evidence that this is indeed the case. We find the β coefficients used by the "Query-Key Normalization" Transformer Attention variant [21]. Query-Key Normalization, makes it straightforward to find β because it learns it via backpropagation and easy to interpret because it uses cosine similarity between the query and key vectors. To further evaluate the convergence between Attention and SDM β values and make it more general, we also investigate the regular GPT2 architecture. However, in this case we need to infer "effective" β values from the size of query key dot products in the softmax. This makes these results, outlined in Appendix A.3, more approximate but they remain largely in agreement with the learnt βs of Query-Key Norm.

The Query-Key Norm Attention heads learn $\beta \in [10, 25]$ as shown in Fig. 3.[6] To compare these results with what SDM predicts, we use as a reference the SNR optimal d^* that assumes random patterns. Attention learns useful pattern representations that are far from random so this SDM β that fits the SNR optimal d^* is only a weak reference for what β values might be reasonable. Larger than optimal d values with random patterns suffer from too much noise, while smaller d values miss signal from the target pattern. In the case of learned pattern representations, we might expect there to be more signal and less noise meaning that d can be larger than it otherwise would be, making the SNR optimal d potentially a lower bound. For Attention implemented by the GPT2 architecture, $n = 64$, $r = 2^n$ and $m \leq 1024$, plugging these into the memory SNR optimal Hamming distance Eq. 1 gives $d = 11$.

Using Eq. 9 to fit a β coefficient to the circle intersection created with $n = 64$, $r = 2^n$, $m \leq 1024$, and $d \geq 11$ gives $\beta \leq 19.3$, which Query-Key Norm closely obeys as is shown in Fig. 3.

Note that the whole range of $\beta \in [10, 25]$ can approximate SDM d values very closely. The relationship between the regression fitted β values and the d values of the SDM instances they approximate is shown in Fig. 3(b). Fig. 2 shows d values that are far from the SNR optimal to show exactly this claim that the approximation remains tight across d, and we also show the quality of these approximations specifically for the Transformer parameters ($n = 64$, $r = 2^n$, $m = 1024$) in Appendix A.1.

In requiring that Attention vectors be L^2 normalized and β fitted, SDM predicted Query-Key Norm. This is interesting because Query-Key Norm has initial evidence of improving Transformer performance and training speed ([21], [27]). We say more about its advantages and caveats in Appendix A.2.

4 Transformer Components Interpreted with SDM

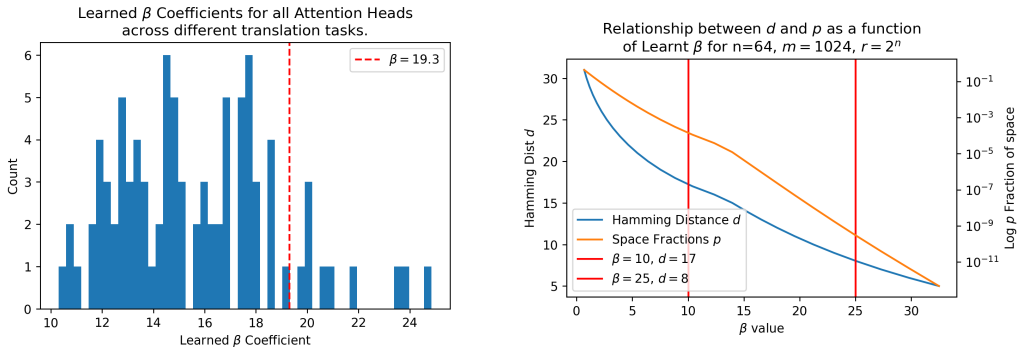
We can leverage how Attention approximates SDM to interpret many components of the Transformer architecture[7]. This exercise demonstrates the explanatory power of SDM and, in relating it to the more unique Transformer components, widens the bridge upon which ideas related to SDM can cross into deep learning and vice versa.

A crucial component of the Transformer is its Feed Forward (FF) layer that is interleaved with the Attention layers and uses approximately 2/3rds of the Transformer's parameter budget [1]. Attention's

[5] The code for running these experiments, other analyses, and reproducing all figures is available at https://github.com/attention-sdm/attention-approximates-sdm.

[6] These values were graciously provided by the authors of [21] in private correspondence for their trained Transformer models.

[7] For a review of the components that make up the full Transformer architecture see Fig. 9 of Appendix A.



(a) (b)

Figure 3: **(a)** Histogram showing the learned β coefficients for all Attention heads across the 5 translation tasks used in [21]. **(b)** The Hamming distance d values (and corresponding space fractions p) that Attention is approximating via its regression fitted βs. The red lines to denote the range of Query-Key Norm $\beta \in [10, 25]$. These values for d and p are all reasonable depending upon how patterns are distributed throughout the space. We present p in addition to d because it is independent of n. This instantiation of SDM uses the same parameters as the Attention head $n = 64$, $r = 2^n$, $m = 1024$.

Transformer implementation and SDM deviate importantly in Attention's use of ephemeral patterns from the current receptive field. In order to model temporal dependencies longer than the receptive field, we want Attention to be able to store persistent memories. Work has compellingly shown that the FF layers store these persistent memories ([28], [29], [30]). SDM can be interpreted as this FF layer because in [28] the FF layer was substituted with additional, persistent key and value vectors that Attention learnt independently of current inputs. This substitution performed on par with the FF layer which, combined with deeper analysis in [29], shows the FF layers are performing Attention with long term memories and thus can be directly interpreted as SDM. Beyond using this long term Attention substitution, SDM can also be interpreted as a FF layer itself with specific requirements that are outlined in B.1. The Transformer FF layer misses some but not all of these requirements, but the aforementioned substitution result shows it is similar enough to be implementing SDM.

Another crucial component of the Transformer is its use of LayerNorm ([31], [32], [1]). LayerNorm has a natural interpretation by SDM as implementing a similar constraint to L^2 normalization. However, while it ensures all of the vectors are on the same scale such that their dot products are comparable to each other, it does not constrain these dot products to the interval $[-1, 1]$ like SDM's cosine similarity.

Multi-headed Attention is a Transformer component where multiple instantiations of Attention operate at the same hierarchical level and have their outputs combined. Multi-heading allows SDM to model probabilistic outputs, providing an interpretation for why it benefits Attention. For example, if we are learning a sequence that can go "$A \rightarrow C$" and "$A \rightarrow Z$" with equal probability, we can have one SDM module learn each transition. By combining their predictions we correctly assign equal probability to each. This probabilistic interpretation could explain evidence showing Attention heads pay attention to different inputs and some are redundant post training ([33], [34], [10]).

Finally, we note that the discrepancy between SDM and Attention around L^2 normalization of the key, query and value vectors has led to two insights. First, as previously mentioned, it predicted that Query-Key Normalization [21] could be a useful inductive bias (more details in Appendix A.2). Second, it has provided caution to interpretations of Attention weights. Fig. 8 of Appendix A.4, shows that there are many value vectors that receive very small amounts of attention but have large L^2 norms that dominate the weighted summation of value vectors. Beyond realizing that interpretations of Attention weights can be highly misleading without this L^2 normalization, we tested L^2 normalized value vectors as another potentially useful inductive bias by training a GPT2 model with it. Our results showed that this did not change performance but L^2 normalization should still be done in cases where the Attention weights are going to be interpreted. It is otherwise misleading to directly

interpret the weights as has been done in work including ([1], [35], [36], [37]). See Appendix A.4 for a full discussion.

5 SDM's Mapping to the Cerebellum

Here, we provide an overview of SDM's biological plausibility to introduce the hypothesis that Attention is performing the same associative memory operations as the cerebellum. In respecting key biological constraints such as Dale's law, SDM's read and write operations can be implemented by a population of neurons satisfying specific connectivity patterns. This connectivity pattern is described in ([13], [26]) and reproduced for convenience in Fig. 21 of Appendix D.6. The most difficult requirements of SDM, that any neuronal implementation of must meet, are ensuring every neuron has three sophisticated abilities: (i) checking if it is within d Hamming distance of the incoming queries being read and patterns being written; (ii) updating its synaptic weights using LTD and LTP (Long Term Depression/Potentiation) to write patterns within d; (iii) outputting its stored memories to the same location as every other neuron to update the query being read (again within d).

Unique architectural features of the cerebellar cortex can implement all of SDM's requirements, specifically via the Granule cells: (i) all granule cells receive inputs from the same mossy fibers to check if they are within d of the incoming query or pattern[8]; (ii) each granule cell has a very long parallel fiber that form synapses with on the order of thousands of Purkinje cells [39]. These synaptic connections store memories and are updated by LTP/LTD through strong action potentials on the Purkinje cell generated by climbing fibers that wrap tightly around individual Purkinje cells and carry the pattern to be written in; (iii) all granule cells output their stored memories via their synaptic weights to the Purkinje cells that perform the summation operation and use their firing threshold to determine if the majority bit was a 1 or 0, updating the query. We provide more details and discussion including some limitations and open questions regarding this model in Appendix E.2. These details include a diagram of the cerebellar connectome in Fig. 34 and a mapping of each of its neurons onto the architectural requirements of SDM by replacing the labels of Fig. 21 with their corresponding neurons in Fig. 35.

While SDM fits the unique features of the cerebellum well, this connection and its evidence relies on neuroscientific evidence available in 1988 [13]. Future work will be necessary to update the state of SDM's biological plausibility with modern neuroscientific findings, although there is reason to believe the core connection will remain because the Marr-Albus-Ito model of the cerebellum that is largely in agreement with SDM remains popular [20]. There are two reasons to be excited about updating SDM's neuroscientific grounding. First, there has been growing evidence that the cerebellum is important for many more cognitive functions than just precise motor control, long believed to be its only role ([40], [41], [42], [43]). Second, improvements to our understanding of the computational capabilities of Pyramidal neurons suggest they may have parallels to Purkinje cells and that the core algorithmic requirements of SDM could be implemented in additional brain regions such as the hippocampus and cortical columns ([44], [16]). These remain early hypotheses but the success of Transformers and their link to SDM should catalyze novel investigations.

6 Related Work

Previous work showed that the modern Hopfield Network, when made continuous and optimized differently, becomes Attention [24]. This result was one motivation for this work because Hopfield Networks are another associative memory model. It has been shown that SDM is a generalization of the original Hopfield Network (Appendix C.5) [45]. In (Appendix C.6), we go further in showing how the new energy functions of the modern discrete Hopfield network that increase memory capacity make its relationship to SDM closer, helping to explain the connection to Attention that was discovered ([46], [47]). While SDM is a generalization of Hopfield Networks, their specific discrepancies provide different perspectives on Attention. Most notably, Hopfield Networks assume symmetric weights that create an energy landscape, which can be powerfully used in convergence proofs [24]. However, these symmetric weights come at the cost of biological plausibility [45] that SDM compellingly provides.

[8] These inputs are sparse while SDM uses dense vectors, however an improved version of SDM resolves this discrepancy [38].

Since the publication of SDM there have been a number of advancements not only to SDM specifically, but also through the creation of related associative memory algorithms under the name of "Vector Symbolic Architectures" [48]. Advancements to SDM include using integer rather than binary vectors [49], handling correlated patterns [45], and hierarchical data storage [50]. Vector Symbolic Architectures, most notably Holographic Reduced Representations, have ideas that can be related back to SDM and the Transformer in ways that may be fruitful ([51], [25], [52], [53], [54], [55]).

The use of external memory modules in neural networks has been explored most notably with the Neural Turing Machine (NTM) and its followup, the Differential Neural Computer (DNC) ([22], [56]). In order to have differentiable read and write operations to the external memory, they use the softmax function. This, combined with their use of cosine similarity between the query and memory locations, makes both models closely related to SDM. A more recent improvement to the NTM and DNC directly inspired by SDM is the Kanerva Machine ([23], [57], [58], [59]). However, the Kanerva Machine remains distinct from SDM and Attention because it lacks the softmax function on the cosine similarity between its query and patterns. Independent of these discrepancies, we believe relating these alternative external memory modules to SDM presents a number of interesting ideas that will be explored further.

7 Discussion

The result that Attention approximates SDM should enable more cross pollination of ideas between neuroscience, theoretical models of associative learning, and deep learning. Considering avenues for future deep learning research, SDM's relationship to Vector Symbolic Architectures is particularly compelling because they can apply logical and symbolic operations on memories, making SDM more powerful ([25], [60], [61], [62]). SDM and its relation to the brain can inspire new research in not only deep learning but also neuroscience, because of the empirical success of the Transformer and its relation to the cerebellum via SDM.

Limitations One limitation of SDM is that Attention remains an approximation rather than the perfect mapping that Hopfield Networks were shown to be capable of. However, what SDM loses in its precise mapping to Attention it gains in stronger biological plausibility. The best efforts to date to make the modern Hopfield network biologically plausible still fail to address the weight transport problem that is used as a core model component [63].

Beyond motivating specific research agendas in deep learning and neuroscience, we hope that our results serve as an existence proof for how powerful deep learning equations can be approximated by, and mapped onto, the functional attributes and connectivity patterns of neuronal populations. At a time when many new neuroscientific tools are mapping out uncharted neural territories, we hope that more discoveries along the lines of this work connecting deep learning to the brain will be made ([64], [65], [66]).

8 Conclusion

We have shown that the Attention update rule closely approximates SDM when made continuous, L^2 norms its vectors, and has an appropriate β coefficient. This result has been shown to hold true in both theory and empirical evaluation of GPT2 Transformer models. SDM predicts that Transformers should normalize their key, query and value vectors, preempting the development of Query-Key Normalization and adding nuance to the interpretation of Attention weights. In mapping SDM onto the Transformer architecture as a whole, relating it to other external memory implementations, and highlighting extensions to SDM, we lay the groundwork for future deep learning research directions. In SDM being a generalization of Hopfield Networks, we outline how it encompasses previous results while contributing new connectome level biological plausibility. More broadly, our work highlights a convergence between deep learning and neuroscience, providing support for how the brain may implement, and deep learning may approximate, associative memory.

References

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

[3] Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications. *OpenAI Blog https://openai.com/blog/better-language-models*, 2019.

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[5] Mark Chen, A. Radford, Jeff Wu, Heewoo Jun, Prafulla Dhariwal, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML 2020*, 2020.

[6] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020.

[7] Anonymous. Videogen: Generative modeling of videos using {vq}-{vae} and transformers. In *Submitted to International Conference on Learning Representations*, 2021. under review.

[8] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, and Scott Gray. Dall·e: Creating images from text, Jan 2021.

[9] Gwern. May 2020 news 'on gpt-3', Dec 2019.

[10] Anna Rogers, O. Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

[11] Martin Schrimpf, I. Blank, Greta Tuckute, Carina Kauf, Eghbal A. Hosseini, N. Kanwisher, J. Tenenbaum, and Evelina Fedorenko. The neural architecture of language: Integrative reverse-engineering converges on a model for predictive processing. *bioRxiv*, 2020.

[12] Shikhar Tuli, Ishita Dasgupta, Erin Grant, and T. Griffiths. Are convolutional neural networks or transformers more like human vision? *ArXiv*, abs/2105.07197, 2021.

[13] Pentti Kanerva. *Sparse distributed memory*. MIT Pr., 1988.

[14] Marvin Minsky and Papert Seymour. *Time vs. memory for best matching - an open problem*, page 222 – 225. 1969.

[15] M. Modi, Yichun Shuai, and G. Turner. The drosophila mushroom body: From architecture to algorithm in a learning circuit. *Annual review of neuroscience*, 2020.

[16] G. Wolff and N. Strausfeld. Genealogical correspondence of a forebrain centre implies an executive brain in the protostome–deuterostome bilaterian ancestor. *Philosophical Transactions of the Royal Society: Biological Sciences*, 371, 2016.

[17] Ashok Litwin-Kumar, K. D. Harris, R. Axel, H. Sompolinsky, and L. Abbott. Optimal degrees of synaptic connectivity. *Neuron*, 93:1153–1164.e7, 2017.

[18] T. Shomrat, Nicolas Graindorge, C. Bellanger, G. Fiorito, Y. Loewenstein, and B. Hochner. Alternative sites of synaptic plasticity in two homologous "fan-out fan-in" learning and memory networks. *Current Biology*, 21:1773–1782, 2011.

[19] S. Shigeno and C. W. Ragsdale. The gyri of the octopus vertical lobe have distinct neurochemical identities. *Journal of Comparative Neurology*, 523, 2015.

[20] M. Kawato, Shogo Ohmae, and Terry Sanger. 50 years since the marr, ito, and albus models of the cerebellum. *Neuroscience*, 462:151–174, 2021.

[21] A. Henry, Prudhvi Raj Dachapally, S. Pawar, and Yuxuan Chen. Query-key normalization for transformers. In *EMNLP*, 2020.

[22] A. Graves, G. Wayne, and Ivo Danihelka. Neural turing machines. *ArXiv*, abs/1410.5401, 2014.

[23] Yan Wu, Greg Wayne, Alex Graves, and Timothy Lillicrap. The kanerva machine: A generative distributed memory, 2018.

[24] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, Victor Greiff, David Kreil, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield networks is all you need, 2020.

[25] T. Plate. Holographic reduced representations: Convolution algebra for compositional distributed representations. In *IJCAI*, 1991.

[26] P. Kanerva. Sparse distributed memory and related models. 1993.

[27] Guang Yang, Xiang Chen, K. Liu, and C. Yu. Deeppseudo: Deep pseudo-code generation via transformer and code feature extraction. *ArXiv*, abs/2102.06360, 2021.

[28] Sainbayar Sukhbaatar, E. Grave, Guillaume Lample, H. Jégou, and Armand Joulin. Augmenting self-attention with persistent memory. *ArXiv*, abs/1907.01470, 2019.

[29] Mor Geva, R. Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. *ArXiv*, abs/2012.14913, 2020.

[30] N. Carlini, Florian Tramèr, Eric Wallace, M. Jagielski, Ariel Herbert-Voss, K. Lee, A. Roberts, Tom Brown, D. Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. *ArXiv*, abs/2012.07805, 2020.

[31] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[32] Kevin Lu, Aditya Grover, P. Abbeel, and Igor Mordatch. Pretrained transformers as universal computation engines. *ArXiv*, abs/2103.05247, 2021.

[33] Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *ArXiv*, abs/1905.10650, 2019.

[34] J Alammar. Interfaces for explaining transformer language models, 2020.

[35] Jesse Vig. Visualizing attention in transformer-based language representation models. *ArXiv*, abs/1904.02679, 2019.

[36] Ian Tenney, James Wexler, Jasmijn Bastings, Tolga Bolukbasi, Andy Coenen, Sebastian Gehrmann, Ellen Jiang, Mahima Pushkarna, Carey Radebaugh, Emily Reif, and Ann Yuan. The language interpretability tool: Extensible, interactive visualizations and analysis for nlp models. In *EMNLP*, 2020.

[37] Joris Baan, Maartje ter Hoeve, M. V. D. Wees, Anne Schuth, and M. Rijke. Do transformer attention heads provide transparency in abstractive summarization? *ArXiv*, abs/1907.00570, 2019.

[38] Louis A. Jaeckel. A class of designs for a sparse distributed memory. 1989.

[39] Eriola Hoxha, F. Tempia, P. Lippiello, and M. C. Miniaci. Modulation, plasticity and pathophysiology of the parallel fiber-purkinje cell synapse. *Frontiers in Synaptic Neuroscience*, 8, 2016.

[40] Christopher J O'Halloran, G. Kinsella, and E. Storey. The cerebellum and neuropsychological functioning: A critical review. *Journal of Clinical and Experimental Neuropsychology*, 34:35 – 56, 2012.

[41] P. Strick, R. Dum, and J. Fiez. Cerebellum and nonmotor function. *Annual review of neuroscience*, 32:413–34, 2009.

[42] D. Timmann and I. Daum. Cerebellar contributions to cognitive functions: A progress report after two decades of research. *The Cerebellum*, 6:159–162, 2008.

[43] S. Petersen, P. Fox, M. Posner, M. Mintun, and M. Raichle. Positron emission tomographic studies of the processing of singe words. *Journal of Cognitive Neuroscience*, 1:153–170, 1989.

[44] M. Larkum. A cellular mechanism for cortical associations: an organizing principle for the cerebral cortex. *Trends in Neurosciences*, 36:141–151, 2013.

[45] James D. Keeler. Comparison between kanerva's sdm and hopfield-type neural networks. *Cognitive Science*, 12(3):299 – 329, 1988.

[46] D. Krotov and J. Hopfield. Dense associative memory for pattern recognition. In *NIPS*, 2016.

[47] M. Demircigil, Judith Heusel, M. Löwe, Sven Upgang, and F. Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168:288–299, 2017.

[48] P. Kanerva. Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors. *Cognitive Computation*, 1:139–159, 2009.

[49] R. Prager and F. Fallside. The modified kanerva model for automatic speech recognition. *Computer Speech Language*, 3:61–81, 1989.

[50] L. Manevitz and Yigal Zemach. Assigning meaning to data: Using sparse distributed memory for multilevel cognitive tasks. *Neurocomputing*, 14:15–39, 1997.

[51] Ivo Danihelka, Greg Wayne, B. Uria, Nal Kalchbrenner, and A. Graves. Associative long short-term memory. In *ICML*, 2016.

[52] P. Kanerva. Binary spatter-coding of ordered k-tuples. In *ICANN*, 1996.

[53] P. Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artif. Intell.*, 46:159–216, 1990.

[54] R. Gayler. Multiplicative binding, representation operators analogy. 1998.

[55] Jeff Hawkins and Subutai Ahmad. Why neurons have thousands of synapses, a theory of sequence memory in neocortex. *Frontiers in Neural Circuits*, 10:23, 2016.

[56] A. Graves, Greg Wayne, M. Reynolds, T. Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, J. Agapiou, Adrià Puigdomènech Badia, K. Hermann, Yori Zwols, Georg Ostrovski, A. Cain, H. King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476, 2016.

[57] Y. Wu, Gregory Wayne, K. Gregor, and T. Lillicrap. Learning attractor dynamics for generative memory. In *NeurIPS*, 2018.

[58] K. Gregor, Danilo Jimenez Rezende, Frédéric Besse, Y. Wu, Hamza Merzic, and Aaron van den Oord. Shaping belief states with generative environment models for rl. *ArXiv*, abs/1906.09237, 2019.

[59] Adam Marblestone, Yan Wu, and Greg Wayne. Product kanerva machines: Factorized bayesian memory, 2020.

[60] C. Eliasmith. How to build a brain: A neural architecture for biological cognition. 2013.

[61] B. Lake, Tomer D. Ullman, J. Tenenbaum, and S. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2016.

[62] S. Piantadosi. The computational origin of representation. *Minds and Machines*, 31:1–58, 2021.

[63] D. Krotov and J. Hopfield. Large associative memory problem in neurobiology and machine learning. *ArXiv*, abs/2008.06996, 2020.

[64] S. Alon, Daniel R. Goodwin, Anubhav Sinha, A. Wassie, Fei Chen, Evan R. Daugharthy, Yosuke Bando, Atsushi Kajita, Andrew G. Xue, Karl Marrett, Robert Prior, Yi Cui, A. Payne, Chun-Chen Yao, Ho-Jun Suk, Ru Wang, Chih chieh Yu, Paul W. Tillberg, P. Reginato, N. Pak, S. Liu, Sukanya Punthambaker, Eswar P. R. Iyer, Richie E. Kohman, J. Miller, E. Lein, Ana Lako, N. Cullen, S. Rodig, K. Helvie, Daniel L. Abravanel, N. Wagle, B. Johnson, J. Klughammer, Michal Slyper, Julia Waldman, J. Jané-Valbuena, O. Rozenblatt-Rosen, A. Regev, G. Church, Adam H. Marblestone, and E. Boyden. Expansion sequencing: Spatially precise in situ transcriptomics in intact biological systems. *bioRxiv*, 2020.

[65] C. Xu, Michał Januszewski, Z. Lu, S. Takemura, K. Hayworth, G. Huang, K. Shinomiya, Jeremy B. Maitin-Shepard, David Ackerman, Stuart E. Berg, T. Blakely, J. Bogovic, Jody Clements, Tom Dolafi, Philip M. Hubbard, Dagmar Kainmueller, W. Katz, Takashi Kawase, K. Khairy, Laramie Leavitt, Peter H. Li, Larry Lindsey, Nicole L. Neubarth, D. J. Olbris, H. Otsuna, Eric T. Troutman, L. Umayam, Ting Zhao, M. Ito, Jens Goldammer, T. Wolff, R. Svirskas, P. Schlegel, E. Neace, Christopher J Knecht, Chelsea X. Alvarado, Dennis A Bailey, Samantha Ballinger, J. Borycz, Brandon S Canino, Natasha Cheatham, Michael Cook, M. Dreher, Octave Duclos, Bryon Eubanks, K. Fairbanks, S. Finley, N. Forknall, Audrey Francis, Gary Hopkins, Emily M Joyce, Sungjin Kim, Nicole A. Kirk, Julie Kovalyak, Shirley Lauchie, Alanna Lohff, Charli Maldonado, Emily A. Manley, Sari McLin, C. Mooney, Miatta Ndama, Omotara Ogundeyi, Nneoma Okeoma, Christopher Ordish, Nicholas L. Padilla,

Christopher Patrick, Tyler Paterson, Elliott E Phillips, E. M. Phillips, Neha Rampally, Caitlin Ribeiro, Madelaine K Robertson, J. Rymer, S. Ryan, Megan Sammons, Anne K. Scott, Ashley L. Scott, A. Shinomiya, C. Smith, Kelsey Smith, N. L. Smith, Margaret A. Sobeski, Alia Suleiman, J. Swift, Satoko Takemura, Iris Talebi, Dorota Tarnogorska, Emily Tenshaw, Temour Tokhi, J. Walsh, Tansy Yang, J. Horne, Feng Li, Ruchi Parekh, P. Rivlin, V. Jayaraman, K. Ito, S. Saalfeld, R. George, I. Meinertzhagen, G. Rubin, H. Hess, Louis K. Scheffer, Viren Jain, and Stephen M. Plaza. A connectome of the adult drosophila central brain. *bioRxiv*, 2020.

[66] Kristin M Scaplen, M. Talay, John D Fisher, Raphael Cohn, Altar Sorkaç, Y. Aso, G. Barnea, and K. Kaun. Transsynaptic mapping of drosophila mushroom body output neurons. *bioRxiv*, 2020.

[67] Toan Q. Nguyen and Julian Salazar. Transformers without tears: Improving the normalization of self-attention. *ArXiv*, abs/1910.05895, 2019.

[68] W. Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *ArXiv*, abs/2101.03961, 2021.

[69] Aäron van den Oord, Oriol Vinyals, and K. Kavukcuoglu. Neural discrete representation learning. In *NIPS*, 2017.

[70] Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. On the relationship between self-attention and convolutional layers. *ArXiv*, abs/1911.03584, 2020.

[71] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and Franccois Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. *ArXiv*, abs/2006.16236, 2020.

[72] S. Ahmad and Luiz Scheinkman. How can we be so dense? the benefits of using highly sparse representations. *ArXiv*, abs/1903.11257, 2019.

[73] Chunjie Luo, Jianfeng Zhan, Lei Wang, and Q. Yang. Cosine normalization: Using cosine similarity instead of dot product in neural networks. *ArXiv*, abs/1702.05870, 2018.

[74] Diederik P. Kingma and M. Welling. An introduction to variational autoencoders. *Found. Trends Mach. Learn.*, 12:307–392, 2019.

[75] C. Doersch. Tutorial on variational autoencoders. *ArXiv*, abs/1606.05908, 2016.

[76] Adityanarayanan Radhakrishnan, Mikhail Belkin, and Caroline Uhler. Overparameterized neural networks implement associative memory, 2020.

[77] I. Kobyzev, S. Prince, and M. Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[78] Louis A. Jaeckel. An alternative design for a sparse distributed memory. 1989.

[79] P. Chou. The capacity of the kanerva associative memory. *IEEE Trans. Inf. Theory*, 35:281–298, 1989.

[80] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[81] J. Hopfield and D. Tank. "neural" computation of decisions in optimization problems. *Biological Cybernetics*, 52:141–152, 2004.

[82] A. Yuille and Anand Rangarajan. The concave-convex procedure (cccp). In *NIPS*, 2001.

[83] Watts DJ;Strogatz SH;. Collective dynamics of 'small-world' networks. *Nature*, Jun 1998.

[84] Frederico A.c. Azevedo, Ludmila R.b. Carvalho, Lea T. Grinberg, José Marcelo Farfel, Renata E.l. Ferretti, Renata E.p. Leite, Wilson Jacob Filho, Roberto Lent, and Suzana Herculano-Houzel. Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *The Journal of Comparative Neurology*, 513(5):532–541, 2009.

[85] Steve Cavill. Number of atoms in the universe, Mar 2016.

[86] Rosenblatt Frank. The perceptron—a perceiving and recognizing automaton. *Cornell Aeronautical Laboratory.*, Report 85-460-1, 1957.

[87] D Purves, GJ Augustine, and D Fitzpatrick. *Neurons Often Release More Than One Transmitter*. Sinauer Associates, 2nd edition, 2001.

[88] Joseph K. Blitzstein and Jessica Hwang. *Introduction to probability*. CRC Press, Taylor Francis Group, 2019.

[89] Ami Citri and Robert C. Malenka. Synaptic Plasticity: Multiple Forms, Functions, and Mechanisms. *Neuropsychopharmacology*, 33(1):18–41, January 2008. Number: 1 Publisher: Nature Publishing Group.

[90] Nikita Kitaev, L. Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. *ArXiv*, abs/2001.04451, 2020.

[91] S. Navlakha, A. Barth, and Z. Bar-Joseph. Decreasing-rate pruning optimizes the construction of efficient and robust distributed networks. *PLoS Computational Biology*, 11, 2015.

[92] Maja Abitz, R. D. Nielsen, E. Jones, H. Laursen, N. Graem, and B. Pakkenberg. Excess of neurons in the human newborn mediodorsal thalamus compared with that of the adult. *Cerebral cortex*, 17 11:2573–8, 2007.

[93] K. Lashley. Brain mechanisms and intelligence: A quantitative study of injuries to the brain. *American Journal of Psychology*, 43:527, 1929.

[94] D. Marr. A theory of cerebellar cortex. *The Journal of Physiology*, 202, 1969.

[95] A. M. Wilson, R. Schalek, Adi Suissa-Peleg, T. R. Jones, Seymour Knowles-Barley, H. Pfister, and J. Lichtman. Developmental rewiring between cerebellar climbing fibers and purkinje cells begins with positive feedback synapse addition. *Cell reports*, 29:2849 – 2861.e6, 2019.

[96] H. Markram, J. Lubke, M. Frotscher, and B. Sakmann. Regulation of synaptic efficacy by coincidence of postsynaptic aps and epsps. *Science*, 275:213 – 215, 1997.

[97] Y. Liang, Chaitanya K. Ryali, B. Hoover, L. Grinberg, S. Navlakha, Mohammed J. Zaki, and D. Krotov. Can a fruit fly learn word embeddings? *ArXiv*, abs/2101.06887, 2021.

[98] Y. Aso, D. Hattori, Y. Yu, R. Johnston, Nirmala Iyer, Teri-Tb Ngo, H. Dionne, L. Abbott, R. Axel, H. Tanimoto, and G. Rubin. The neuronal architecture of the mushroom body provides a logic for associative learning. *eLife*, 3, 2014.

[99] T. Zars. Behavioral functions of the insect mushroom bodies. *Current Opinion in Neurobiology*, 10:790–795, 2000.

[100] P. Mobbs. The brain of the honeybee apis mellifera. i. the connections and spatial organization of the mushroom bodies. *Philosophical Transactions of the Royal Society B*, 298:309–354, 1982.

[101] M. Suenaga, Y. Kawai, H. Watanabe, N. Atsuta, M. Ito, F. Tanaka, M. Katsuno, H. Fukatsu, S. Naganawa, and G. Sobue. Cognitive impairment in spinocerebellar ataxia type 6. *Journal of Neurology, Neurosurgery, and Psychiatry*, 79:496 – 499, 2007.

[102] Jinzhi Li, Brennan Dale Mahoney, Miles Solomon Jacob, and Sophie J. C. Caron. Visual input into the drosophila melanogaster mushroom body. *Cell reports*, 32 11:108138, 2020.

[103] Kristin M Scaplen, M. Talay, J. Fisher, Raphael Cohn, Altar Sorkaç, Yoshi Aso, Gilad Barnea, and K. Kaun. Transsynaptic mapping of drosophila mushroom body output neurons. *eLife*, 10, 2021.

[104] M. Sereno, J. Diedrichsen, M. Tachrount, Guilherme Testa-Silva, H. D'Arceuil, and C. D. De Zeeuw. The human cerebellum has almost 80neocortex. *Proceedings of the National Academy of Sciences of the United States of America*, 117:19538 – 19543, 2020.

[105] D. V. Van Essen, C. Donahue, and M. Glasser. Development and evolution of cerebral and cerebellar cortex. *Brain, Behavior and Evolution*, 91:158 – 169, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.

- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No] We do not see any negative societal impacts emerging from this work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] at https://github.com/attention-sdm/attention-approximates-sdm
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [No] Everything can be run quickly on a laptop making discussion of these details unimportant.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We make new figures that are in the main text and supplement.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]