PNAS RESEARCH ARTICLE | NEUROSCIENCE APPLIED MATHEMATICS

A mathematical theory of relational generalization in transitive inference

Samuel Lippl[a,b,c,1], Kenneth Kay[a,b,d], Greg Jensen[a,c,e], Vincent P. Ferrera[a,c,f], and L. F. Abbott[a,b,c]

Edited by James McClelland, Stanford University, Stanford, CA; received August 22, 2023; accepted May 30, 2024

Humans and animals routinely infer relations between different items or events and generalize these relations to novel combinations of items. This allows them to respond appropriately to radically novel circumstances and is fundamental to advanced cognition. However, how learning systems (including the brain) can implement the necessary inductive biases has been unclear. We investigated transitive inference (TI), a classic relational task paradigm in which subjects must learn a relation ($A > B$ and $B > C$) and generalize it to new combinations of items ($A > C$). Through mathematical analysis, we found that a broad range of biologically relevant learning models (e.g. gradient flow or ridge regression) perform TI successfully and recapitulate signature behavioral patterns long observed in living subjects. First, we found that models with item-wise additive representations automatically encode transitive relations. Second, for more general representations, a single scalar "conjunctivity factor" determines model behavior on TI and, further, the principle of norm minimization (a standard statistical inductive bias) enables models with fixed, partly conjunctive representations to generalize transitively. Finally, neural networks in the "rich regime," which enables representation learning and improves generalization on many tasks, unexpectedly show poor generalization and anomalous behavior on TI. We find that such networks implement a form of norm minimization (over hidden weights) that yields a local encoding mechanism lacking transitivity. Our findings show how minimal statistical learning principles give rise to a classical relational inductive bias (transitivity), explain empirically observed behaviors, and establish a formal approach to understanding the neural basis of relational abstraction.

relational learning | compositional generalization | transitive inference | relational representations

Significance

The ability to infer how elements are related is fundamental to our cognition: when we encounter new circumstances composed of familiar elements, grasping relationships helps us generalize. An important instance is transitive inference (TI): if we know that $A > B$ and $B > C$, we can infer that $A > C$. However, it has been unclear how the brain (and other learning systems) implement such relational generalizations. Here, we investigated artificial learning systems (such as neural networks) that do not have transitivity built in. Remarkably, we found that they perform TI and show behaviors seen in humans and animals. Our findings explain how simple learning models can implement the kind of relational generalization that is essential for successful behavior.

Humans and animals have a remarkable ability to generalize to circumstances that are radically different from their prior experience. They are able to do so, in part, by learning relationships between different events or items and extending these relationships to novel combinations of components (1). Such relational generalization is important across a broad range of domains: for example, subjects can reason about social relationships between individuals they have never seen interact (2), take a novel route between familiar locations (3), or apply familiar tools to novel problems (4). Accordingly, relational cognition has been implicated in a variety of cognitive abilities, including social cognition (5), spatial navigation (6), and logical and causal cognition (7).

It is unclear how living subjects, and learning systems more generally, can learn the kinds of abstractions needed for relational generalization. To generalize from limited experience, subjects (whether they are humans, animals, or learning models) need an "inductive bias": a disposition toward certain behaviors among the many that are consistent with past experience (8). Much attention has been devoted to clarifying suitable inductive biases on standard statistical tasks that require generalization to nearby data points ("near transfer" or "in-distribution generalization") (9, 10). An important instance of such a "statistical inductive bias" is norm minimization, which selects the model parameters with the smallest weight norm and is at the core of many successful learning models (11–13). Both theoretical and practical insights suggest that such a solution is likely to generalize well in distribution (14, 15). In contrast, our understanding of how learning models perform relational tasks, which require generalization to radically different circumstances ("far transfer" or "out-of-distribution generalization"), has been much more limited (16, 17). Addressing this gap is essential to understanding how living subjects perform these tasks.

To investigate this question, we studied transitive inference (TI; Fig. 1*A*; 18–20), a classical cognitive task that tests whether humans or animals can generalize transitively. In this task, subjects are presented with pairs of items (Fig. 1*B*) and must pick the "larger" item according to an implicit hierarchy ($A > B > \cdots > G$, Fig. 1*A*).

Author affiliations: [a]Mortimer B. Zuckerman Mind Brain Behavior Institute, Department of Neuroscience, Columbia University, New York, NY 10027; [b]Center for Theoretical Neuroscience, Department of Neuroscience, Columbia University, New York, NY 10027; [c]Department of Neuroscience, Columbia University Medical Center, New York, NY 10032; [d]Grossman Center for the Statistics of Mind, Columbia University, New York, NY 10027; [e]Department of Psychology, Reed College, Portland, OR 97202; and [f]Department of Psychiatry, Columbia University Medical Center, New York, NY 10032

Author contributions: S.L., K.K., and L.F.A. designed research; S.L. performed research; S.L. analyzed data; and S.L., K.K., G.J., V.P.F., and L.F.A. wrote the paper.

The authors declare no competing interest.

This article is a PNAS Direct Submission.

Copyright © 2024 the Author(s). Published by PNAS. This article is distributed under Creative Commons Attribution-NonCommercial-NoDerivatives License 4.0 (CC BY-NC-ND).

[1]To whom correspondence may be addressed. Email: samuel.lippl@columbia.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2314511121/-/DCSupplemental.

Published July 5, 2024.

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

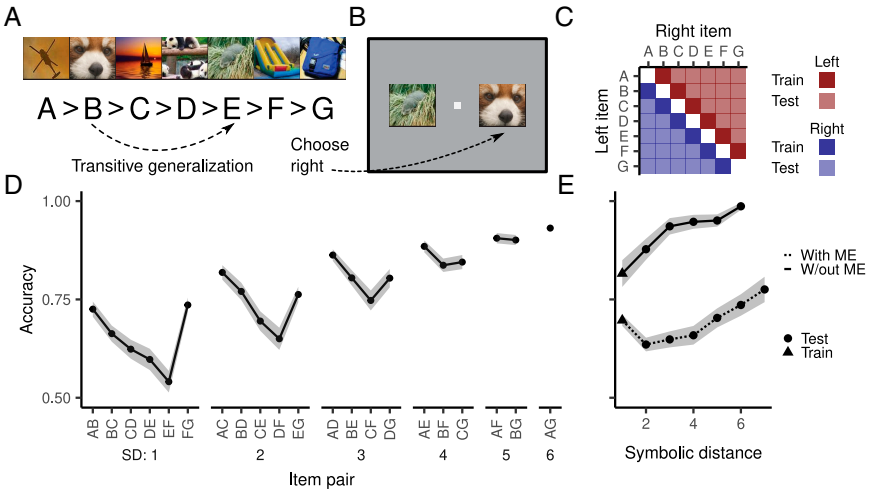


Fig. 1. Transitive inference and behavioral patterns observed in subjects performing this task. (*A*) Example stimuli taken from ref. 23. An example generalization ($B > E$) is highlighted. (*B*) In a given trial, the subject is asked to choose between the two presented items. They are rewarded if the chosen item is larger according to the underlying hierarchy. This panel depicts a test trial where successful performance would consist in picking the item on the right. (*C*) Schematic of the training and test cases. (*D*) Example accuracy on all training and test cases (in this case by rhesus macaques). Terminal item, symbolic distance, and asymmetry effect are apparent in the plot. In the subsequent figures, we leave off the item pair labels but use the same ordering. Symbolic distance (SD) is the separation in the rank hierarchy. The data are reproduced from ref. 23. (*E*) Symbolic distance effect with and without a memorization effect (ME). Data without ME reproduced from ref. 23 and data with ME reproduced from ref. 21. Shaded regions in both panels indicate mean ± one SE.

Importantly, subjects are not informed of this task structure (cf. refs. 21 and 22) and, further, only receive feedback about the correct response on adjacent pairs (*AB, BC, . . . , FG*). Hence, they must infer the underlying relation and use transitivity to determine the correct response to nonadjacent pairs ($B > E$, Fig. 1*C*).

Notably, humans (24) and a variety of animals [ranging from monkeys (25) and rodents (26) to wasps (27) and fish (28)] perform TI successfully. Moreover, they show consistent behavioral patterns (in their reaction time and accuracy, Fig. 1*D*), which may be informative of the underlying neural implementation (19, 20). First, subjects' performance improves with increasing separation in the hierarchy ("symbolic distance effect," e.g., 24, 29, 30). Second, subjects' performance tends to be better for trials involving more terminal (e.g., A and G) rather than more intermediate (e.g., D and E) items ("terminal item effect," e.g., refs. 31 and 32). Third, some studies indicate better performance on training trials (which have a symbolic distance of 1) than on test trials with a symbolic distance of 2 (21, 33, 34, Fig. 1*E*), a limited violation of the symbolic distance effect which we here refer to as the memorization effect. Finally, subjects' performance is often better for item pairs toward the beginning of the hierarchy than item pairs toward the end (the "asymmetry effect," which we do not address but return to in the discussion) (e.g., ref. 21).

Many simple learning models have been shown to perform TI. However, the inductive biases that give rise to this ability are not well understood. Various learning models associate a numerical "rank" with each presented item and choose whichever item has a higher rank (21, 23, 35). Such models are preconfigured to generalize transitively and leave unclear how transitive generalization could arise from more basic learning principles. They also leave unclear how the brain, which is not constrained in this way (36), could implement TI. Intriguingly, several studies have found that generic neural networks can generalize transitively, suggesting that statistical learning principles can sometimes give rise to a suitable relational inductive bias (22, 37–39). However, TI in neural networks has largely been studied through simulations, rather than analytically (cf., ref. 40), raising the question of when and how statistical learning models can implement TI.

Here we show, via both analytical approaches and simulations, that a broad range of biologically relevant learning models generalize transitively and recapitulate the symbolic distance, terminal item, and memorization effect. We first consider models with "additive" representations that represent the two presented items independently and show that they are constrained to implementing a transitive relation. If models additionally represent nonlinear conjunctions between items (as is important for many other tasks) but use norm minimization to determine their readout weights, they also generalize transitively. Remarkably, the same learning principle that underpins many instances of successful near transfer also enables this instance of successful far transfer. We further show that, for TI, the effect of a particular choice of internal representation can be characterized by a single scalar "conjunctivity factor." Finally, we consider models which adapt their internal representation to a given task, an ability thought critical to human and animal cognition. Surprisingly, we find that this impairs performance on TI and leads to behavioral patterns that deviate from those in living subjects. Notably, this anomalous behavior is explained by a different form of norm minimization, namely one over all weights in the network rather than just the readout weights.

At first glance, TI appears to be a complex task involving relational, rule-based cognition. Nevertheless, we can characterize a broad range of learning models performing this task in exact analytical terms, explaining how they give rise to the rich behavioral patterns observed in living subjects. In doing so, our investigation clarifies systematically how a learning principle that has largely been considered in the context of near transfer, also implements an important instance of far transfer and relational abstraction.

Model Setup

We represent individual items as high-dimensional vectors. A trial input is a concatenation of the two vectors X, Y corresponding to the two presented items. We generally consider a learning model f that represents this input as a numerical vector $g(X, Y)$ (in the simplest case, this could be the input vector itself). The model then computes a linear readout from that representation: $f(X, Y) = w \circ g(\mathrm{X}, \mathrm{Y})$. A positive model output ($f(X, Y) > 0$) corresponds to $X > Y$, whereas a negative output ($f(X, Y) < 0$) corresponds to $X < Y$. We generally assume that w is learned from the training trials, whereas $g(X, Y)$ remains fixed (for example arising from the representation in a neural network with random or prelearned weights). In the final two sections, we investigate models that also learn $g(X, Y)$.

Inputs where $f(X, Y) = 0$ lie on the model's decision boundary. Accordingly, a higher magnitude of the margin ($f(X, Y)$ if $X > Y$ or $-f(X, Y)$ if $X < Y$) corresponds to

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

a larger distance from the decision boundary. We take this to indicate better performance, corresponding to higher choice accuracy and lower reaction time. This is a standard assumption in decision-making models (22, 41). For example, in a drift diffusion model (42), the output $f(X, Y)$ determines the model's drift rate. A higher drift rate makes the model less susceptible to noise (improving accuracy) and makes it cross its threshold faster (improving reaction time) (43).

Results

An Additive Representation Yields Transitive Generalization. To perform TI, or indeed any kind of relational inference, a learning model's representation of items X and Y, $g(X, Y)$, should reflect the fact that X and Y are separate items. In the most extreme case, this would amount to an additive representation, where $g(X, Y)$ is a sum of two separate representations $g_1(X)$ and $g_2(Y)$: $g(X, Y) = g_1(X) + g_2(Y)$. A simple instance of this is a model architecture where nodes respond exclusively either to X or Y (Fig. 2*A*).

A change in one of the two items will leave half of the additive representation unchanged (Fig. 2*B*), implementing a kind of compositionality (44). A linear readout from an additive representation, $f(X, Y) = w \circ g(X, Y)$, is a sum of responses to each individual item, $w \circ g_1(X) + w \circ g_2(Y)$, and therefore also additive. The consequences of this are especially clear in the case of a model without a choice bias, i.e., if $f(X, X) = w \circ g_1(X) + w \circ g_2(X) = 0$ (*SI Appendix*, section S1.A considers the biased case). In this case, we know that $-w \circ g_1(X) = w \circ g_2(X)$, and, as a result, the model's decision function can be expressed as

$$f(X, Y) = w \circ g_1(X) - w \circ g_1(Y).$$

This means that the model necessarily learns to assign a scalar rank $r(X) = w \circ g_1(X) = -w \circ g_2(X)$ to each item, and computes its decision by comparing the two ranks (Fig. 2*C*):

$$f(X, Y) = r(X) - r(Y).$$

Note that $f(X, Y) > 0$, i.e., the decision to choose the left item, is equivalent to $r(X) > r(Y)$.

As shown in previous work (e.g., refs. 21, 23, and 31), learning systems that are preconfigured to have such a ranking system yield both transitive generalization and the symbolic distance effect. This is because, to learn the training set, the model's ranks must be monotonically decreasing:

$$r(A) > r(B) > \cdots > r(G).$$

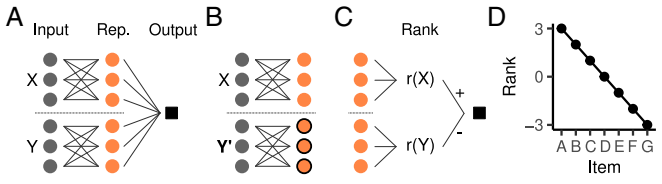


Fig. 2. An additive representation yields transitive generalization and the symbolic distance effect. (*A*) Schematic illustration of an additive representation (rep.). The first three orange nodes represent the first item (X), and the latter three represent the second item (Y). (*B*) Replacing the second item only results in changes to the highlighted half of the units. (*C*) The readout weights of the model can be grouped into those pertaining to item X and those pertaining to item Y. The model's output can be understood as assigning a rank $r(X)$ and $r(Y)$ to each item and then computing $r(X) - r(Y)$. (*D*) Example of a model's rank assignment.

As an example, if the model output has a margin of one for all inputs composed of adjacent items, the rank must decrease by one for each successive item (Fig. 2*D*). A monotonically decreasing rank directly implies that nonadjacent items are also ordered correctly and, consequently, that the model will generalize transitively. Further, item pairs with larger symbolic distance will have a larger difference in their ranks, giving rise to a symbolic distance effect.

Critically, and in distinction to previous work, the above model structure (additive representation) does not explicitly preconfigure or assume a ranking system. Rather, the above analysis shows that any learning model that implements an additive input representation necessarily implements a ranking system and thus encodes a transitive relation. Further, the above analysis makes no assumptions about how the model learns. As long as the model correctly classifies the training cases, it will correctly classify all test cases, i.e., transitively generalize, and show a symbolic distance effect.*

A Single Scalar Fully Characterizes a Broad Range of Relational Representations. The above analysis indicates that an additive representation would enable the brain to perform TI. However, neural representations are not thought to be fully additive (45). Indeed, nonadditive representations (Fig. 3*A*) are important for learning relevant tasks across a broad range of domains and, with some differences in implementation, are known under a correspondingly broad range of names, including conjunctive (46) or configural (47) representations, nonlinear features, and representations with mixed selectivity (45). We next asked whether and how nonadditive representations can support transitive generalization.

To begin, consider the most extreme conjunctive case: a one-hot representation in which each composition of items is represented by a different hidden unit (Fig. 3*B*). In this case, a change in one of the two items yields a completely different representation. Such a model is able to memorize the training cases, but cannot generalize transitively.

Many representations, whether in the brain or in other learning systems, are neither fully conjunctive nor fully additive, but rather lie in between these two extremes (45, 48). To characterize this spectrum formally, we considered the representational similarity between two trials (X, Y) and (X', Y'), as measured by their dot product, $\langle g(X, Y), g(X', Y') \rangle$ (leaving the representations $g(X, Y)$ fixed). We assumed that the representational similarity between two trials only depends on whether these trials are distinct ((X, Y) and (X', Y')), overlapping ((X, Y) and (X', Y) or (X, Y')), or identical ((X, Y) and (X, Y)) (Fig. 3*C*). We call this the "exchangeability assumption."

The exchangeability assumption captures the fact that model behavior should not depend on the particular (i.e., arbitrary) hierarchy in which the set of items is arranged (49). To promote exchangeability, we assumed that all input items are equally correlated with each other. In this case, most commonly used nonlinear representations of that input satisfy exchangeability as well. As a paradigmatic learning model, we considered a neural network with a ReLU nonlinearity and random weights. By determining the expected value of the representational similarity analytically (*SI Appendix*, section S1.C), we found that the network's hidden layer, in expectation, satisfies the exchangeability assumption (Fig. 3*C*). This is because even

*A related paper (39) has come to our attention, in which the authors are concerned with a linear item representation. This is a special case of the additive representations here. Further, the authors derive an analytic solution for a specific learning algorithm, whereas we demonstrate a transitive constraint for a broader class of models.

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

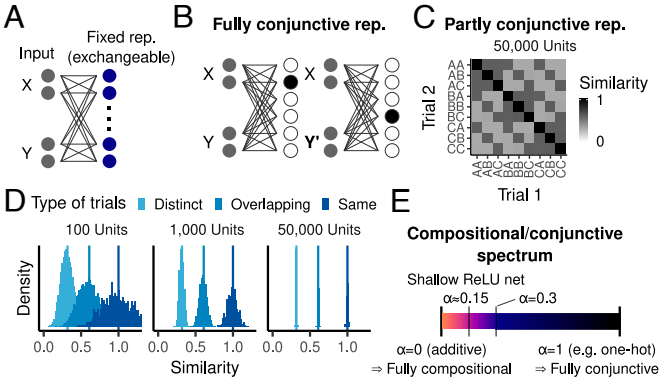


Fig. 3. Nonadditive representations of TI cases. (*A*) A nonadditive representation encodes nonlinear interactions between items. (*B*) A one-hot representation represents each combination of items by a distinct node. (*C*) Representational similarity (cross-correlation) between a subset of trials in a ReLU network with 50,000 units. (*D*) Representational similarity between all possible trials in networks with different numbers of hidden units, organized according to the type of trials. (*E*) The conjunctivity factor characterizes a given representation according to how similarly it represents overlapping trials. Additive representations lie at one end of the spectrum, whereas one-hot representations lie at the other end.

though the hidden layer computes a nonlinear transformation, it partially inherits the input's similarity structure (50). In network simulations, the empirical representational similarity exhibits some variance around the expected value due to the random weight initialization. However, this variance vanishes as the network's hidden layer becomes wider (Fig. 3*D*).

Under the exchangeability assumption, we found that for learning models that only modify their readout weights, a single scalar, which we call the conjunctivity factor $\alpha \in [0, 1]$, fully determines their TI task behavior, i.e., all models that have representations with the same α exhibit the same behavior on TI (*SI Appendix*, section S1.B). The conjunctivity factor is given by

$$\alpha := 1 - 2\frac{\kappa_o - \kappa_d}{\kappa_s - \kappa_d}, \quad \kappa_s = \langle g(X, Y), g(X, Y)\rangle,$$
$$\kappa_o = \langle g(X, Y), g(X, Y')\rangle, \quad \kappa_d = \langle g(X, Y), g(X', Y')\rangle. \qquad [1]$$

Here κ_s, κ_o, and κ_d are the similarity between identical, overlapping, and distinct pairs, respectively. Notably, we can extend the definition of the conjunctivity factor to nonexchangeable representations by taking the average over all identical, overlapping, or distinct pairs.

For an additive representation, half the units share their activation between different overlapping pairs. As a result, the similarity between overlapping pairs is halfway between that of distinct and that of identical pairs, corresponding to $\alpha = 0$. At the other extreme, $\alpha = 1$ indicates that overlapping pairs are encoded in the model with equal similarity to each other as completely distinct pairs (as is the case for a one-hot representation). Consequently, α systematically characterizes the spectrum from fully additive to fully conjunctive representations (Fig. 3*E*).

In most commonly used representations, overlapping trials are more similar to each other than distinct trials and therefore have an intermediate value for α ("partly conjunctive" representations). This is because the input space represents overlapping trials as more similar than distinct trials and, as noted above, the hidden layer partly inherits the input's similarity structure. Through analytical computation, we found that a random ReLU network with one hidden layer, for example, has $\alpha \approx 0.15$ (*SI Appendix*, Eq. **S28**; Fig. 3 *C* and *D*).

Importantly, the network giving rise to the representation could have an arbitrary number of layers; all we need to know is the conjunctivity factor of the network's final layer (see also *SI Appendix*, section S1.C).

For TI, the conjunctivity factor raises two questions. First, how does $\alpha > 0$ affect TI behavior? Second, if the learning model's internal representation is modifiable rather than fixed, how does this affect the conjunctivity factor and subsequent TI behavior?

Norm Minimization and Partly Conjunctive Representations Yield Transitive Generalization. Unlike an additive representation, a representation with $\alpha > 0$ is not constrained to implementing a transitive relation. To understand how models with partly conjunctive representations perform on TI, we need to consider additional constraints. In particular, we analyzed models in which the learning of readout weights implements norm minimization (Fig. 4*A*), a paradigmatic statistical inductive bias (Fig. 4*B*). Under the exchangeability assumption, we were able to characterize model behavior on TI through exact analytical solutions. As noted above, to perform this analysis, we do not need to know the particular representation implemented, only its conjunctivity factor α.

On the training cases, the minimal norm model necessarily assigns a margin of ± 1, as dictated by the desired output. On the test cases, however, our analysis revealed an intriguing emergent behavior: the model's response to item pair (i, j) invariably reflects a ranking system:

$$f(i, j) = r_i(\alpha) - r_j(\alpha),$$
$$\text{with } r_i(\alpha) = \frac{\sinh((\frac{n+1}{2} - i)\lambda(\alpha))}{\sinh(\frac{n+1}{2}\lambda(\alpha)) - \sinh(\frac{n-1}{2}\lambda(\alpha))} \qquad [2]$$
$$\text{and } \lambda(\alpha) := \operatorname{arccosh}\left(\frac{1}{1-\alpha}\right),$$

where n is the total number of items (see *SI Appendix*, section S1.D.3 for the derivation of this result). This is remarkable as the model architecture is not constrained or preconfigured to implement a ranking system. Rather, the behavior is a consequence of the principle of norm minimization, operating on a partly conjunctive representation. Importantly, as implied by its ranking system, the model generalizes transitively (as long as $\alpha < 1$), and exhibits a symbolic distance effect.

For an intuition as to why a ranking system emerges, consider a particular class of representations having one-hot representations of the first and second item individually as well as their conjunction (Fig. 4*C*). This representation will have a conjunctivity factor of α if the item-wise units are weighted by $\sqrt{\frac{1-\alpha}{2}}$ and the conjunctive units are weighted by $\sqrt{\alpha}$. Changes in the item-wise unit weights correspond to changes in the model's rank, as they generalize to overlapping trials. In contrast, changes in the conjunctive unit weights correspond to memorization, as they have no effect on overlapping trials. In principle, the model could learn the training set through changes in the conjunctive unit weights alone. However, because more distributed weights tend to have a smaller norm, norm minimization causes the model to learn the training trials by changing both the conjunctive unit weights (resulting in memorization) and the item-wise unit weights (resulting in transitive generalization). Thus, partial conjunctivity is necessary for the existence of an

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

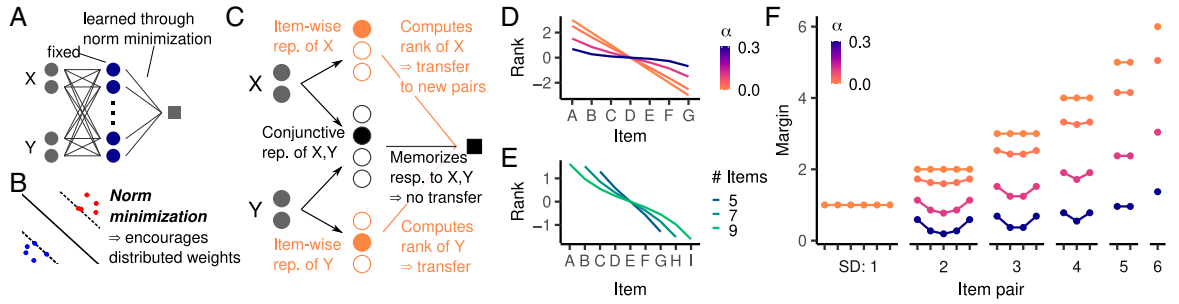


Fig. 4. We analyze the behavior of models having readout weights trained with norm minimization. (*A*) Schematic illustration of the setup. (*B*) Norm minimization implements a useful inductive bias for generalization to nearby data points. On categorization tasks, it determines the hyperplane separating the two categories with the maximal margin. (*C*) Intuitively, a partly conjunctive representation is given by an item-wise representation of *X* and *Y* concatenated with a fully conjunctive representation of *X* and *Y*. The readout from the item-wise representations computes a rank for each item that transfers to overlapping pairs. The readout from the fully conjunctive representation memorizes a response to a given pair and does not transfer to overlapping pairs. Because norm minimization encourages distributed weights, it finds a solution that partly uses the item-wise representation and hence computes a rank. This leads to transitive generalization. (*D* and *E*) The emergent rank representation at the end of training (Eq. **2**) for (*D*) seven items and different values for α; and (*E*) $\alpha = 0.1$ and different numbers of items. (*F*) For seven items and different values for α, the corresponding margin for all trials. Item pairs are arranged by their position in the hierarchy, as in Fig. 1*E*.

item-wise population and norm minimization ascertains that this population is implicated in the learning process.

Beyond transitive generalization, the principle of norm minimization gives rise to several empirically observed behavioral patterns. The hyperbolic sine making up the rank expression compresses more intermediate items more strongly than more terminal items (Fig. 4*D*), thus giving rise to a terminal item effect (Fig. 4*F*). This effect becomes stronger for higher values of α. A higher α also compresses the ranking more strongly overall, leading to lower margins on the test set. As α approaches one (the fully conjunctive case), the ranking becomes entirely flat and therefore no longer supports transitive generalization.

The form of the ranking also depends on the total number of items. Specifically, intermediate items are compressed more strongly when there are more items in total, an effect that is moreover dependent on α. For $\alpha = 0$, the ranking grows linearly with the number of items, whereas at higher values of α, the ranking's overall range (between the first and last item) is nearly invariant to the number of items (Fig. 4*E*).

Finally, when $\alpha > 0$, the model assigns a larger margin to the training cases than specified by the ranking. Intuitively, this is because the conjunctive unit weights contribute to model behavior on the training cases but do not transfer to the test cases. Since a higher α compresses the ranks further, it leads to a higher discrepancy between training and test behavior. At a sufficiently high α, the margin of the training cases is larger than that of the test cases with a small symbolic distance (Fig. 4*F*), giving rise to a memorization effect. While Ciranka et al. (21) explained this effect by fitting an explicit memorization parameter, our analysis reveals that it is an emergent consequence of having a nonlinear representation with sufficiently high conjunctivity factor. Notably, the fact that the memorization effect only arises in a subset of models (i.e., those with relatively conjunctive representations) may explain why it is only occasionally observed in living subjects. In contrast, the symbolic distance and terminal item effect arise across the full spectrum of relational representations (except the fully conjunctive case) and indeed are also observed more consistently in living subjects.

The above analytical solutions depend on the exchangeability assumption. In practice, the trials might be represented in a manner that violates this assumption. Indeed, we already saw that though randomly sampled features satisfy exchangeability in expectation, a model with insufficiently many of those features will have a nonexchangeable representation due to finite samples (Fig. 3*D*). In simulations (*SI Appendix*, section S2.B), we found that a slight violation of exchangeability (e.g., resulting from a representation with many random features) does not change model behavior substantially. For larger violations (e.g., resulting from a representation with fewer random features), model behavior deviates from our theoretical account, but is still well approximated by it. Further, our account captured the average behavior across many models with a small number of random features almost exactly. This suggests that models with nonexchangeable representations behave differently from those with exchangeable representations but our analytical solutions can still be useful for understanding their behavior.

Our analysis demonstrates that norm minimization can explain not only transitive generalization, but also the symbolic distance effect, the terminal item effect (though only on test cases), and the memorization effect. We next characterize two popular statistical learning models implementing norm minimization: ridge regression and gradient flow, as applied to the learning of network readout weights.

Learning through Gradient Flow or with Weight Regularization Smoothly Approaches the Minimal Norm Solution. To see how the principle of norm minimization governs TI behavior across learning, we analyzed models with readout weights learned through either ridge regression (Fig. 5*A*; ref. 11) or gradient flow, using mean squared error as the loss function $L(w)$ (with the target response on the training cases being ± 1). Ridge regression minimizes the sum of this loss and the squared L_2-norm of the model weights, i.e., $L(w) + \frac{1}{c}\|w\|_2^2$.

Here, the regularization coefficient c balances the weight penalty $\|w\|_2^2$ with the minimization of the loss function $L(w)$. Gradient flow, on the other hand, assumes that a model minimizes $L(w)$ by following the pointwise gradient exactly. This approximates gradient descent with a small learning rate and is more amenable to formal analysis (51). In this case, the learning duration t determines how well the model has learned to minimize the loss function in the allotted time. At initialization, the model should be agnostic to all choices (i.e., output zero) and we therefore assumed that the weights are initialized at zero.

The regularization coefficient c and the learning duration t play a similar role in the two learning models. With $c = 0$, the weight penalization is infinitely more important than the task-based

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

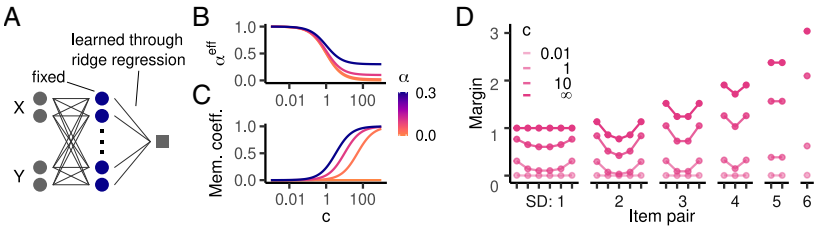


Fig. 5. TI behavior for models with a fixed representation and readout weights trained using regularized regression. (*A*) Illustration of the setup. (*B* and *C*) The (*B*) effective conjunctivity factor and (*C*) memorization coefficient as a function of the inverse regularization coefficient c. (*D*) Generalization behavior for $\alpha = 0.1$ and different values of c. The margins overall become larger as c increases.

component of the loss function and therefore the model weights are all zero. Similarly, at $t = 0$, the weights are initialized to zero. On the other hand, in the limit of infinite training ($t \to \infty$), the model converges to the minimal norm solution determined in the section above (52). This is also the case for the limit of models trained with increasingly small weight penalization ($c \to \infty$) (53). Ridge regression and gradient flow are therefore two instances of common learning models which can perform TI by implementing the principle of norm minimization.

Going beyond these limits, we obtained exact solutions to model behavior for arbitrary t or c. For ridge regression (*SI Appendix*, section S1.D.3), we found that the test behavior of a model with weight regularization and a given conjunctivity factor α is equivalent to that of a model without weight regularization but with a different, "effective" conjunctivity factor α^{eff}, which depends on both α and c (*SI Appendix*, Lemma S1.4): $\alpha^{\text{eff}} = (\alpha + \frac{1}{c})/(1 + \frac{1}{c})$. α^{eff} is generally higher (i.e., more conjunctive) than α and, as c becomes larger, gradually approaches α from above (Fig. 5*B*). Thus, a model with smaller c has a more compressed rank and a more pronounced terminal item effect.

On the training cases, the model's performance is boosted compared to the rank difference, just as in the case of norm minimization. Specifically, its behavior is given by

$$f(i, i \pm 1) = m \cdot \pm 1 + (1 - m) \cdot (r_i(\alpha^{\text{eff}}) - r_{i\pm1}(\alpha^{\text{eff}})),$$
$$m := \frac{\alpha}{\alpha + \frac{1}{c}}, \qquad [3]$$

i.e., a mixture of memorization (assigning a constant margin of ± 1) and reliance on the ranking system. The balance between the two behaviors is specified by the "memorization coefficient" $m \in [0, 1]$. m starts out at zero, indicating no memorization and full reliance on the ranking system. As c grows larger, m increases as well (Fig. 5*C*). The smaller m, the more strongly model behavior relies on the ranking. In particular, this partial reliance leads to a terminal item effect on the training cases in addition to the test cases (Fig. 5*D*). For $c \to \infty$, m converges to 1, indicating full memorization as observed for the minimal norm solution.

For gradient flow, we found qualitatively similar solutions for model behavior (*SI Appendix*, section S1.D.4). In particular, the model's behavior on test cases can be described by a ranking system throughout learning. Further, the model ranks gradually approach the minimal norm solution and, at earlier stages of training, have a more pronounced terminal item effect. Finally, the model has a transient terminal item effect on the training cases that vanishes as $t \to \infty$.

Humans and animals generally exhibit a terminal item effect on the training set. Our analysis suggests that this could be caused by either weight regularization or incomplete training (or both), arising from a mechanism that is related to but distinct from the mechanism giving rise to the terminal item effect on the test cases.

The Conjunctivity Factor Exposes a Tradeoff between Learning Transitive and Nontransitive Relational Tasks. Our analysis thus far indicates that a higher conjunctivity factor generally yields worse performance on TI. Indeed, if maximal generalization performance on TI were the sole aim, models with fully additive representations would be ideal. However, as noted above, humans and animals learn a broad range of relational tasks, not all of them transitive. Because models with fully additive representations are constrained to implementing a transitive relation, this makes partially conjunctive representations necessary (*SI Appendix*, Fig. S1*A*).

To investigate how different representational geometries change model behavior beyond TI, we considered transverse patterning ("rock, paper, scissors" with more than three items, e.g.: $A > B, B > C, \ldots, F > G, G > A$). This task exemplifies a nontransitive relation and can be learned by both humans and various animals (36, 54, 55).

As with TI, we considered models with fixed, exchangeable representations with readout weights trained through gradient flow (*SI Appendix*, Fig. S1*A*). We found that as long as $\alpha > 0$, models were able to learn the task by relying on the conjunctive population. Further, by solving the learning dynamics of gradient descent analytically (*SI Appendix*, section S1.F), we found that a higher α leads to faster learning of the training trials (*SI Appendix*, Fig. S1 *B* and *C*). In contrast, on TI, a higher α causes such models to have a smaller test margin. These behaviors highlight a potential tradeoff that α imposes across different kinds of relations. In particular, our analysis predicts that subjects who are better at TI should be slower to learn transverse patterning and vice versa.

One strategy for avoiding the tradeoff described above is representation learning. If models were able to adapt their internal representations to a given task, they could in principle learn an additive representation for TI and a nonadditive representation for nontransitive relational tasks such as transverse patterning. We now investigate this hypothesis.

Neural Networks with Adaptive Representations Show Anomalous TI Behavior. Deep neural networks, which learn by updating their internal weights, have become increasingly relevant both as artificial learning systems (56) and models of cognitive processes (57, 58). Importantly, recent work has revealed that their generalization behavior fundamentally depends on the magnitude of their initial weights. For large initial weights, learning dynamics can be approximated by gradient flow on a particular fixed-feature model called the neural tangent kernel (NTK; 59). Thus, even though the network updates its internal weights, it effectively still relies on a fixed representation. Accordingly, this regime is often called "lazy" (60). In contrast, neural networks initialized from sufficiently small values learn truly task-specific representations ("rich regime"). Broadly considered, adapting a model's internal representation to a given task could address the competing demands imposed by the wide range of tasks subjects need to perform.

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

Indeed, the rich regime is seen as essential to the remarkable generalization capabilities of deep neural networks (60–62). There is also widespread evidence that subjects adapt their representation to a given task (63) and may do so similarly to deep neural networks trained in the rich regime (64). In relational tasks, in particular, neural representations change to reflect how different items are related to each other (65–67), and this may be essential for successful generalization on certain tasks (22, 68). In light of this lazy vs. rich distinction and its potential relevance to biological learning, we investigated through simulations whether deep neural networks are suitable as a model of relational representation learning (Fig. 6*A*). Specifically, we trained neural networks from different scales of initialization using gradient descent (for details on training, see *SI Appendix*, section S2.C). In light of the important role played by the initialization scale, we covered a broad range of potential values, focusing on three representative values: 1 (resulting in lazy behavior), 10^{-3} (resulting in rich behavior), 10^{-16} (to characterize model behavior in the limit of small initialization).

In the lazy regime, we computed the NTK's conjunctivity factor analytically (*SI Appendix*, section S1.E). For large initialization, the learning trajectory of neural networks trained with full weight updating is approximated by gradient flow on the fixed NTK representation. Hence we were able to use the gradient flow solutions determined in the previous sections to predict the network behavior over the course of training, finding a virtually perfect match with simulations (Fig. 6 *B* and *E*, green line). Our account may therefore be able to explain why previous studies (37, 38) found empirically that feedforward neural networks generalize transitively.

Surprisingly, we found that networks trained in the rich regime performed worse at TI. They had a smaller test margin (Fig. 6*C*) and also made systematic errors on the cases CE, BE, and CF for a sufficiently small initialization scale (Fig. 6 *D* and *E*, hollow points). Further, and in contrast to all learning models considered thus far, the behavior of neural networks in the rich regime was not consistent with a ranking system. This is apparent from the fact that the networks' margin at a symbolic distance of three was smaller than at a symbolic distance of two (Fig. 6*E*). Importantly, in contrast to the previous limited violation of the symbolic distance effect, this cannot be explained by memorization as none of these cases are in the training set. Finally, the networks in the rich regime exhibited a violation of the terminal item effect at a symbolic distance of 4, a somewhat surprising pattern that no models examined thus far have exhibited. The networks' unconventional behavior was not due to the specific setup considered here: we observed similar behavior for alternative activation functions (*SI Appendix*, Fig. S6), alternative loss functions (*SI Appendix*, Fig. S8*A*), and deeper networks (*SI Appendix*, Fig. S8*B*). Further, the networks exhibited even more overtly idiosyncratic behavior for larger numbers of items, one striking behavioral pattern being a periodic (rather than monotonically increasing) symbolic distance effect (*SI Appendix*, Fig. S7). These findings indicate that, for TI, representation learning in standard neural networks does not necessarily confer the benefits of representation learning suggested in prior work.

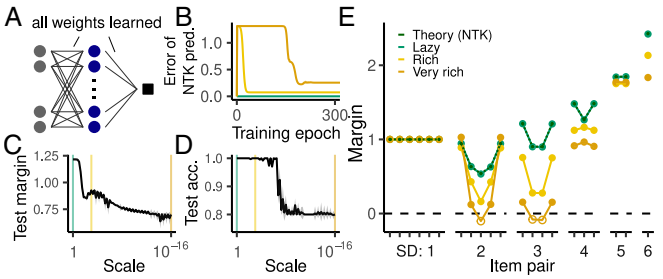


Fig. 6. TI behavior in ReLU networks with one hidden layer and 50,000 units that are trained through backpropagation. Shaded regions (sometimes too small to be visible) indicate mean ± one SD across twenty instances. (*A*) Illustration of network training. In contrast to the previous setups, the hidden layer weights were also trained using backpropagation. (*B*) The mean squared error of the prediction made by the NTK at three different initialization scales (lazy: 1; rich: 10^{-3}; very rich: 10^{-16}). (*C* and *D*) The (*C*) average test margin and (*D*) test accuracy as a function of initialization scale. The colored lines highlight the three representative values analyzed in more detail in panels *B* and *E*. (*E*) TI performance according to our NTK-based prediction as well as of networks trained with backpropagation at the three representative scales.

Rich-Regime Networks Implement a Cooperative Code that Lacks a Transitive Inductive Bias. Given that the rich regime has been found previously to improve generalization on other tasks, we sought to understand why it yields anomalous behavior on TI. To this end, we leveraged previous work indicating that lazy and rich regimes implement different forms of norm minimization: the lazy regime minimizes the ℓ_2-norm of the network's readout weights, whereas the rich regime approximately minimizes the ℓ_2-norm of all weights in the network (69, 70; but see refs. 71 and 72). For the networks studied here, we found that the norm of all weights in a fully trained network is dramatically smaller for smaller scales of initialization (Fig. 7*A*). To clarify why norm minimization over all network weights is associated with an anomalous inductive bias (unlike norm minimization over readout weights, analyzed in the previous sections), we directly analyzed the computations performed by the rich-regime neural network.

Prior studies (70, 73, 74) have found that minimization of the ℓ_2-norm over all network weights induces the networks' hidden units to "specialize" into a low number of functionally distinct clusters with low ℓ_2-norm. To assess this possible structure, we performed k-means clustering with respect to the normalized weight vectors across all units, finding that in the rich regime, all 50,000 units of the network fall into just six clusters (Fig. 7*B*; details described in *SI Appendix*, section S3.C). In contrast, such a compact description was not apparent in the lazy regime. Remarkably, the six cluster centroids were highly consistent across random initializations (Fig. 7*C*).

We found that three of the centroids ("units") had positive readout weights, whereas the other three units had negative readout weights. This is because the network has rectified activations in its hidden layer, which caused different units to specialize for trials with positive and negative labels. In examining hidden-layer weights, we focused on the positive units, which we denote by E_{1+}, E_{2+}, and E_{3+} (negative units are analogous, *SI Appendix*, Fig. S11). Because we presented the network with concatenated one-hot vectors, each weight entry corresponds to a different presented item and we identify each weight by its corresponding item. Note that items presented in the first and second position correspond to entirely distinct weights; we denote item X in the first or second position by $X^{(1)}$ and $X^{(2)}$, respectively.

We noticed two putative aspects of the underlying computation in the network: different units responded predominantly to a nonoverlapping set of trials ("staggered response"; e.g., E_{1+} (pink) responded to AB, BC, EF, and FG, while E_{2+} (blue) responded to CD and E_{3+} (green) responded to DE; Fig. 7 *C*, *Bottom Left*) and each unit encoded its corresponding trials

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

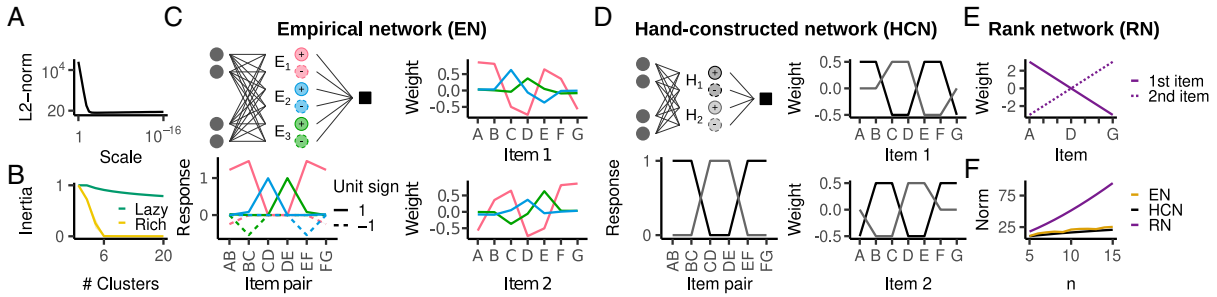


Fig. 7. A mechanistic analysis of the rich regime's inductive bias. All plots show the mean ± SD across twenty random initializations. SD is too small to be visible. (*A*) ℓ_2-norm of all weights in the fully trained networks as a function of initialization scale. The rich regime yields much smaller norm. (*B*) Inertia (i.e., proportion of explained variance) as a function of the number of clusters, for the lazy and rich network. Six clusters only leave 0.001% of the variance unexplained. (*C*) The empirical network is therefore described by a network with six units, three with positive and three with negative responses. The *Right* panels show the weights of the different units and the *Bottom* panel shows how each units responds to the different training trials. Only units with positive readout weights are shown; the units with negative readout weights have the same structure but with Items 1 and 2 reversed. (*D*) Analogous depiction of the hand-constructed network (HCN), which has four units. (*E*) The rank network only has two units, but they span a much wider range. (*F*) Weight norm of the empirical, hand-constructed, and rank network as a function of the number of items. The rank network has a much larger ℓ_2-norm, whereas the norm of the empirical network is similar to that of the HCN.

by distributing positive weights across both items ("cooperative code"; e.g., to encode AB, E_{1+} assigned a positive weight to $A^{(1)}$ and $B^{(2)}$; Fig. 7 *C*, *Right*).

To see whether these two aspects could provide a sufficient explanation for the behavior in the rich regime, we used them to hand-construct a simplified ReLU network (Fig. 7*D*). Specifically, the hand-constructed network (HCN) has four hidden units, two with positive readout weights and two with negative readout weights. Again, we focus on those with positive readout weights, which we denote by H_{1+} and H_{2+} (see *SI Appendix*, Fig. S10 for the analogous negative units). Note that we do not prove that the HCN actually learns the training trials with minimal ℓ_2-norm (though it has the lowest norm among all networks considered here). Rather, it serves as a useful construction to understand why the described computation yields both low norm and a nontransitive inductive bias.

Each unit in our construction implements a cooperative code, i.e., to encode its response to a trial, it distributes its weights equally between the two items. In particular, H_{1+} classifies trials AB and BC by assigning a weight of 0.5 to $A^{(1)}$ and $B^{(2)}$ as well as $B^{(1)}$ and $C^{(2)}$. Because of the positive weights associated with these items, H_{1+} would also respond positively to trials BA, CB, and DC. Preventing this positive response requires negative weights associated with items $A^{(2)}$, $C^{(1)}$, and $D^{(1)}$. However, because a negative weight is already associated with $C^{(1)}$ and $D^{(1)}$, H_{1+} cannot classify trials CD and DE using a cooperative code. Thus, H_{1+} stays silent on these trials, which are instead encoded by H_{2+} (also using a cooperative code). This pattern explains the staggered responses observed in the empirical network. The interference from the negative weights associated with $E^{(1)}$ and $F^{(1)}$ prevents H_{2+} from classifying EF and FG, but these trials are no longer affected by interference with AB and BC and can therefore be encoded by H_{1+}. For TI variants with more than seven items, the two units continue alternating, giving rise to periodic network weights (*SI Appendix*, Fig. S10).

Intuitively, the coding scheme amounts to a set of local rules that support learning of the training trials, but do not generalize to the test trials. For example, because the cooperative code implemented in unit H_{1+} encodes a positive response to trials AB and EF, it also results in an incorrectly positive response on EB (*SI Appendix*, Fig. S10).

Critically, the HCN has a lower ℓ_2-norm because the cooperative code keeps the network weights in a range from -0.5 to 0.5. In contrast, a network using a ranking system has weights between $-\frac{n-1}{2}$ and $\frac{n-1}{2}$ (Fig. 7*E*). As a result, its ℓ_2-norm is not only consistently larger, but also grows faster with an increasing number of items (Fig. 7*F*).

Importantly, the staggered cooperative coding scheme is only norm-efficient because the hidden weights are followed by a rectifying nonlinearity. For example, H_{1+} takes on a value of -1 in response to CD. Without the rectification, H_{2+} would then have to compensate for the negative response to produce a positive label, which would require larger weights and thus would not be norm-efficient. However, because of the rectification following the hidden weights, H_{1+} remains silent on CD and requires no such compensation from H_{2+}. This explains why a cooperative and staggered coding scheme has a low ℓ_2-norm when all weights of the network are trained, but not when only the readout weights are trained.

The HCN illustrates how a constraint that imposes a low ℓ_2-norm on all network weights can give rise to anomalous behavior on TI. As noted above, the empirical network implemented the same computational principles as the HCN, albeit in a slightly different way: E_{1+} approximately implemented H_{1+}, whereas E_{2+} and E_{3+} jointly approximated H_{2+}. These differences resulted in a somewhat higher, but qualitatively similar norm in the empirical network (Fig. 7*F*; see *SI Appendix*, section S3.C.3). In particular, the norm of both the empirical and the HCNs is systematically lower than that of the rank-network (Fig. 7*F*). Remarkably, the mechanism implemented by the empirical network is consistent across not only different random initializations but also different numbers of items n: E_{1+} always approximates H_{1+}, whereas E_{2+} and E_{3+} jointly implement H_{2+} (*SI Appendix*, Fig. S11*F*).

Our findings in this section may appear contrary to those of Nelli et al. (22), who also studied TI performance of neural networks, but found that the rich regime did not impair transitive generalization and further yielded an explicit rank representation in the hidden layer. We found that this was caused by a weight symmetry imposed on the hidden layer that constrains their networks to an additive internal representation (*SI Appendix*, section S2.F). Indeed, our clustering analysis revealed that these networks approximately implemented the rank network (Fig. 7*E*) and the ℓ_2-norm of their weights was much higher than that of

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

the unconstrained neural network (*SI Appendix*, Fig. S11*A*). This further illustrates the misalignment between the rich regime's norm minimization and a transitive inductive bias: architectural constraints that improve transitive generalization (here, weight symmetry) lead to a higher weight norm.

Discussion

We found that standard statistical learning models can perform TI and recapitulate three empirically observed behavioral patterns (symbolic distance effect, terminal item effect, and memorization effect). The behavior of a given model is sufficiently captured by a single scalar conjunctivity factor α, which characterizes the model's internal representation (of task items) on a spectrum from fully compositional ($\alpha = 0$) to fully conjunctive ($\alpha = 1$). For $\alpha = 0$, the model is constrained to encoding a transitive relation. For partly conjunctive representations ($0 < \alpha < 1$), the model is not constrained in this way, but the principle of norm minimization nevertheless yields transitive generalization. For fully conjunctive representations ($\alpha = 1$), the model cannot generalize transitively. Finally, we found that when representation learning is enabled in hidden layers, networks perform worse on TI and exhibit different behavioral patterns than living subjects. Through hand-constructed networks and a clustering-based analysis of empirical networks, we suggest that this anomalous behavior arises from a different form of norm minimization.

Models of relational cognition often represent relations explicitly and are preconfigured to have a particular relational inductive bias (75). In particular, alternative accounts of TI are either preconfigured to associate a rank (or value) with each item (21, 23, 35) or suggest that humans rely on abstract knowledge of transitivity (e.g., refs. 23, 76, and 77) [as implemented, for instance, in a cognitive map (78–80)]. In contrast to these accounts, we took a "minimal principles" approach, representing nothing but the input itself (i.e., the two presented items). This perspective casts higher-level behavioral capacities (in this case transitive generalization) as emergent from minimally structured learning systems (81). Studies within this paradigm usually rely on simulations, which can leave the mechanisms for emergent behavior, such as generalization, unclear (cf. ref. 82). In contrast, our analytical account identifies the specific model components responsible for transitive generalization. This clarifies how the brain could implement relational generalizations without preconfigured representations or compositional constraints.

More generally, relational cognition likely relies on a wide range of learning mechanisms (83, 84). The models considered here require repeated interleaved presentations of the training trials (85), suggesting learning mechanisms associated with prefrontal or higher-level association cortices. In contrast, other brain regions, such as the hippocampus, support rapid learning without the need for repeated trial presentations, presumably through the operation of memory reactivation (e.g., inferring that $B > D$ by recalling that $B > C$ and $C > D$) (86–88). A recurrent neural network model with Hebbian plasticity ("REMERGE") has been proposed to explain how a reactivation-based mechanism could support transitive generalization (89). While this learning model does not recapitulate behavioral patterns such as the symbolic distance effect, it can explain how subjects may learn TI from a minimal number of trials. Finally, whereas both the reactivation-based and statistical learning models above rely on emergent inductive biases of the underlying learning mechanisms, learning systems can also develop relational inductive biases through structure learning or meta-learning (90, 91), which are associated with both hippocampus and prefrontal cortex (84, 92–94).

Which of these learning mechanisms is implicated in a particular TI task variant likely depends on factors such as the stimulus structure and how training and test trials are presented. Subjects may also rely on a mixture of learning mechanisms on a single task. For example, trials could initially be encoded in the hippocampus but eventually be consolidated in the prefrontal cortex (83, 95). Creating more unified models of relational learning (e.g., fusing the *REMERGE*-model and our similarity-based mechanism, or incorporating structure learning) could shed further light on the interplay between different learning mechanisms on TI and their dependency on different task parameters. Doing so may also allow closer investigation of proposed neural implementations of relational learning (86).

With respect to similarity-based relational learning models, our account could be seen as an endpoint to a series of investigations of TI behavior: expanding upon previous studies (22, 37, 38), we show comprehensively that the principle of norm minimization enables any model with partly conjunctive representations to generalize transitively and further gives rise to naturalistic behavior on TI. Importantly, norm minimization is implicated not only in gradient flow and ridge regression (the examples we consider), but also a much broader range of learning models (96), including reinforcement learning (97). Accordingly, the consistent behaviors many different animals exhibit on the task could be due to this shared, underlying learning principle. This is an alternative to the view that the ubiquity of TI stems from its ecological role in social cognition (98), and the view that TI entails explicit reasoning.

Our results, while providing an alternative explanatory account of TI, should nevertheless be interpreted cautiously with regard to the basis of TI in living subjects, as our model is limited in a number of ways. In terms of behavioral predictions, it cannot account for the asymmetry effect, the observation that performance in living subjects is often better for items toward the start of the hierarchy (e.g., AB) than items toward the end (e.g., FG) (21). Further, our model does not generate predictions for what behavior we might expect from subjects that, after being trained on item pairs, are presented with three items at a time and expected to pick the largest item (25, 99). Finally, our results do not speak to tasks testing for TI in the form of a single question (e.g., "Alice is taller than Bob and Bob is taller than Chris. Who is taller, Alice or Chris?") or after minimal training (18, 100). Such a format likely requires a different mechanism from the one considered here and may be better understood as an instance of explicit reasoning.

These limitations, and extensions of TI, can inspire refining or expanding the statistical modeling approach. For example, Nelli et al. (22) found that conventional statistical learning models could not account for human behavior on a particular variant of TI ("list linking," 101), but that a modification in which uncertainty is encoded could. More broadly, TI, with its rich and well-established set of empirical phenomena (e.g., refs. 20 and 102), may serve as a useful model task to compare statistical learning models (and their failure modes) to human and animal behavior. For example, the lack of an asymmetry effect and the inability to perform the three-item task indicate what is lost by treating TI as a binary categorization task. A model that understands its decision as tied to one of the presented items (rather than as a choice between two arbitrary categories) may be able to better account for these behaviors (21). Notably, better accounts of behavior within the TI framework may not only elucidate how subjects perform relational learning, but

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

even inform more general models of statistical learning in living subjects.

Despite these potential limitations, our account still makes a set of falsifiable predictions that can produce evidence for or against the biological relevance of our insights. In particular, we identified the conjunctivity factor α as a broadly important parameter for TI task behavior. Our account would predict that changes in α should lead to a set of coordinated changes in behavior. For example, a stronger memorization effect should be associated with a weaker terminal item effect on the training cases (as both are caused by an increase in α). Observing that these different behavioral patterns indeed change in a coordinated manner (for example, between different subjects performing the same task) may indicate that our theory can accurately describe behavior.

Changes in the conjunctivity factor may arise from interindividual differences (different subjects may have representations that are best described by different values of α) but could also be introduced through experimental interventions. In particular, certain task variations may promote a representation of the input as either two separate items or a single conjunctive stimulus; for example, presenting the two items in a common scene rather than as distinct stimuli may result in a higher α (103). More broadly, this suggests that the conjunctivity factor may be a useful parameter for comparing subjects and TI task variants.

Beyond behavioral predictions, the conjunctivity factor could also be used to clarify the neural basis of TI. On the one hand, perturbations in relevant neural areas, for example through lesions (55), may affect the conjunctivity factor. On the other hand, using neural recordings to estimate the empirical representational similarity (104) between distinct, overlapping, and identical trials could enable the estimation of an effective conjunctivity factor in a particular neural area. If differences in this estimated conjunctivity factor (either due to interindividual differences or as a result of lesions) are associated with the corresponding behavioral differences predicted by our model, this may suggest a role for statistical learning as described here, and, in addition, that the recorded neural area is indeed involved in TI.

Moreover, our findings on rich-regime neural networks performing worse on TI may have important implications for machine learning, as deep neural networks have been observed to struggle with tasks involving compositional generalization more broadly (105–108). A common strategy in attempting to address these shortcomings consists in training ever larger models on ever larger datasets (109). Such models have shown impressive results on tasks such as natural language production (110), but the overall scale and complexity of the training data, model, and learning algorithm make it difficult to attribute successful or unsuccessful generalization behaviors to specific components of the model (though see refs. 111 and 112). In contrast, TI is a particularly simple relational task on which deep networks exhibit nonnaturalistic behavior. Our findings explain why the rich regime, which has a useful inductive bias on many different tasks, can give rise to such anomalous behavior on TI. This can help in designing principled modifications to prevent such anomalous behavior—for example, by adding additional training trials, regularizing the hidden layer to prevent a local encoding mechanism, or changing the connectivity structure of the network—and in this way improve our understanding of how relational learning is implemented (113–115). Notably, the tools we employ to analyze the rich-regime behavior may also prove useful for analyzing neural networks trained on other tasks. At the same time, our analysis here was primarily empirical, but future work could provide an analytical characterization of the rich-regime solution (70, 116).

Finally, TI is a simple instance of compositional generalization, i.e., the ability to conceptualize prior experience in terms of components that can be reconfigured in a novel situation (117). More broadly, compositional generalization has long been understood to be a crucial component for human-like learning and generalization, in large part due to the diversity and breadth of its applications: compositional generalization often involves the composition of many rules, relations, or attributes (114). The comparative simplicity of TI enabled us to identify how minimally structured learning systems can implement the inductive biases needed for this task. Our analysis provides a case study for how standard statistical inductive biases determine behavior on compositional tasks, in particular clarifying how representational structure (conceptualized through the conjunctivity factor α) impacts compositional generalization (118). At the same time, TI certainly does not capture the complexity of most compositional task paradigms and future work should extend our formal analysis to a broader scope of compositional tasks. This would clarify the conditions under which standard statistical learning principles can explain compositional generalization and which compositional motifs require additional learning mechanisms.

In summary, we here investigated TI, a task that tests for a fundamental logical capacity and has fascinated researchers across neuroscience and psychology for many decades. We derived exact mathematical equations describing the TI behavior of a large class of statistical learning models. This allowed us to understand systematically how these models can generalize compositionally, using TI as an example. Our theory provides a basis for using TI to investigate the neural structures implementing relational cognition. At the same time, our findings also suggest that TI, in its standard form, may not be sufficient to clarify relational inference abilities that require more than statistical learning. For this purpose, other relational tasks should be considered.

Materials and Methods

Additional methods and results are provided in *SI Appendix*. *SI Appendix*, section S1.A analyzes models with additive representations. *SI Appendix*, sections S1.B and S1.C describe the conjunctivity factor and determine its value across network architectures and nonlinearities. *SI Appendix*, section S1.D describes the mathematical analysis of models whose readout weights are trained using norm minimization, ridge regression, and gradient flow. *SI Appendix*, Section S1.E derives the learning dynamics of neural networks trained in the lazy regime. *SI Appendix*, section S1.F analyzes models whose readout weights are trained on transverse patterning with gradient flow. *SI Appendix*, section S2.A describes the packages used in data analysis and visualization and *SI Appendix*, section S2.B describes experiments analyzing TI behavior in models with non-exchangeable representations. *SI Appendix*, section S2.C describes how neural networks were trained with backpropagation. *SI Appendix*, section S3 describes the analysis of networks related to rich-regime training (empirical and HCNs).

Data, Materials, and Software Availability. The code and data required to reproduce our findings has been stored in a Zenodo database (119).

ACKNOWLEDGMENTS. We are grateful to the members of the Center for Theoretical Neuroscience at Columbia University, the SueYeon Chung lab at Flatiron/NYU, and attendees at Cosyne 2023 and the Gatsby Tri-Centre Meeting 2023 for helpful comments. We thank Taiga Abe, Veronica Bossio, Tala Fakhoury, Ching Fang, Jeff Johnston, Erica Shook, Sharon Su, and Denis Turcu for detailed feedback. The work was supported by NSF 1707398 (Neuronex), Gatsby Charitable Foundation GAT3708, National Institute of Mental Health MH126158-01A1, and NIH-R01MH111703.

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

1. G. S. Halford, W. H. Wilson, S. Phillips, Relational knowledge: The foundation of higher cognition. *Trend. Cognit. Sci.* **14**, 497–505 (2010).
2. D. L. Cheney, R. M. Seyfarth, J. B. Silk, The responses of female baboons (*Papio cynocephalus ursinus*) to anomalous social interactions: Evidence for causal reasoning? *J. Comp. Psychol.* **109**, 134–141 (1995).
3. A. S. Etienne, K. J. Jeffery, Path integration in mammals. *Hippocampus* **14**, 180–192 (2004).
4. A. Seed, R. Byrne, Animal tool-use. *Curr. Biol.* **20**, R1032–R1039 (2010).
5. M. W. Baldwin, Relational schemas and the processing of social information. *Psychol. Bull.* **112**, 461 (1992).
6. E. C. Tolman, Cognitive maps in rats and men. *Psychol. Rev.* **55**, 189–208 (1948).
7. D. C. Penn, D. J. Povinelli, Causal cognition in human and nonhuman animals: A comparative, critical review. *Annu. Rev. Psychol.* **58**, 97–118 (2007).
8. T. M. Mitchell, "The need for biases in learning generalizations" (Tech. Rept. CBM-TR-117, Rutgers University, NJ, 1980).
9. A. Krogh, J. A. Hertz, Generalization in a linear perceptron in the presence of noise. *J. Phys. A: Math. Gen.* **25**, 1135 (1992).
10. A. Canatar, B. Bordelon, C. Pehlevan, Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nat. Commun.* **12**, 2914 (2021).
11. A. E. Hoerl, R. W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* **12**, 55–67 (1970).
12. C. Cortes, V. Vapnik, Support-vector networks. *Mach. Learn.* **20**, 273–297 (1995).
13. D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, N. Srebro, The implicit bias of gradient descent on separable data. *J. Mach. Learn. Res.* **19**, 1–57 (2018).
14. A. Krogh, J. Hertz, "A simple weight decay can improve generalization" in *Advances in Neural Information Processing Systems*, J. Moody, S. Hanson, R. P. Lippmann, Eds. (Morgan-Kaufmann, San Francisco, CA, 1991), vol. 4, pp. 950–957.
15. J. Moody, "The effective number of parameters: An analysis of generalization and regularization" in *Nonlinear Learning Systems in Advances in Neural Information Processing Systems*, J. Moody, S. Hanson, R. P. Lippmann, Eds. (Morgan-Kaufmann, San Francisco, CA, 1991), vol. 4, pp. 847–854.
16. S. M. Barnett, S. J. Ceci, When and where do we apply what we learn? A taxonomy for far transfer. *Psychol. Bull.* **128**, 612–637 (2002).
17. P. W. Battaglia *et al*., Relational inductive biases, deep learning, and graph networks. arXiv [Preprint] (2018). https://arxiv.org/abs/1806.01261 (Accessed 14 June 2024).
18. J. Piaget, *Judgment and reasoning in the child* (Harcourt, Brace, Oxford, England, 1928).
19. M. Vasconcelos, Transitive inference in non-human animals: An empirical and theoretical analysis. *Behav. Proc.* **78**, 313–334 (2008).
20. G. Jensen, "Serial learning" in *APA Handbook of Comparative Psychology: Perception, Learning, and Cognition. APA Handbooks in Psychology*, J. Call, G. M. Burghardt, I. M. Pepperberg, C. T. Snowdon, T. Zentall, Eds. (American Psychological Association, Washington, DC, 2017), vol. 2, pp. 385–409.
21. S. Ciranka *et al*., Asymmetric reinforcement learning facilitates human inference of transitive relations. *Nat. Hum. Behav.* **6**, 555–564 (2022).
22. S. Nelli, L. Braun, T. Dumbalska, A. Saxe, C. Summerfield, Neural knowledge assembly in humans and neural networks. *Neuron* **111**, 1504–1516.e9 (2023).
23. G. Jensen, F. Muñoz, Y. Alkan, V. P. Ferrera, H. S. Terrace, Implicit value updating explains transitive inference performance: The betasort model. *PLoS Comput. Biol.* **11**, e1004523 (2015).
24. P. E. Bryant, T. Trabasso, Transitive inferences and memory in young children. *Nature* **232**, 456–458 (1971).
25. B. O. McGonigle, M. Chalmers, Are monkeys logical? *Nature* **267**, 694–696 (1977).
26. H. Davis, Transitive inference in rats (*Rattus norvegicus*). *J. Comp. Psychol.* **106**, 342–349 (1992).
27. E. A. Tibbetts, J. Agudelo, S. Pandit, J. Riojas, Transitive inference in *Polistes* paper wasps. *Biol. Lett.* **15**, 20190015 (2019).
28. L. Grosenick, T. S. Clement, R. D. Fernald, Fish can infer social rank by observation alone. *Nature* **445**, 429–432 (2007).
29. G. R. Potts, Storing and retrieving information about ordered relationships. *J. Exp. Psychol.* **103**, 431–439 (1974).
30. M. R. D'Amato, M. Colombo, The symbolic distance effect in monkeys (*Cebus apella*). *Anim. Learn. Behav.* **18**, 133–140 (1990).
31. L. Von Fersen, C. D. Wynne, J. D. Delius, J. E. Staddon, Transitive inference formation in pigeons. *J. Exp. Psychol. Anim. Behav. Process.* **17**, 334 (1991).
32. C. D. L. Wynne, Pigeon transitive inference: Tests of simple accounts of a complex performance. *Behav. Process.* **39**, 95–112 (1997).
33. B. D. Acuna, J. N. Sanes, J. P. Donoghue, Cognitive mechanisms of transitive inference. *Exp. Brain Res.* **146**, 1–10 (2002).
34. M. Van Elzakker, R. C. O'Reilly, J. W. Rudy, Transitivity, flexibility, conjunctive representations, and the hippocampus. I. An empirical analysis. *Hippocampus* **13**, 334–340 (2003).
35. C. D. L. Wynne, Reinforcement accounts for transitive inference performance. *Anim. Learn. Behav.* **23**, 207–217 (1995).
36. C. D. L. Wynne, Transverse patterning in pigeons. *Behav. Process.* **38**, 119–130 (1996).
37. C. De Lillo, D. Floreano, F. Antinucci, Transitive choices by a simple, fully connected, backpropagation neural network: Implications for the comparative study of transitive inference. *Anim. Cognit.* **4**, 61–68 (2001).
38. K. Kay *et al*., Emergent neural dynamics and geometry for generalization in a transitive inference task. *PLOS Comput. Biol.* **20**, e1011954 (2024).
39. G. Di Antonio, S. Raglio, M. Mattia, Ranking and serial thinking: A geometric solution. bioRxiv [Preprint] (2023). https://www.biorxiv.org/content/10.1101/2023.08.03.551859v1 (Accessed 14 June 2024).
40. H. O. Carmesin, H. Schwegler, Parallel versus sequential processing of relational stimulus structures. *Biol. Cybernet.* **71**, 523–529 (1994).
41. R. Bogacz, E. Brown, J. Moehlis, P. Holmes, J. D. Cohen, The physics of optimal decision making: A formal analysis of models of performance in two-alternative forced-choice tasks. *Psychol. Rev.* **113**, 700–765 (2006).
42. R. Ratcliff, A theory of memory retrieval. *Psychol. Rev.* **85**, 59–108 (1978).
43. R. Ratcliff, G. McKoon, The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Comput.* **20**, 873–922 (2008).
44. S. M. Frankland, J. D. Greene, Two ways to build a thought: Distinct forms of compositional semantic representation across brain regions. *Cereb. Cortex* **30**, 3838–3855 (2020).
45. M. Rigotti *et al*., The importance of mixed selectivity in complex cognitive tasks. *Nature* **497**, 585–590 (2013).
46. J. W. Rudy, R. C. O'Reilly, Conjunctive representations, the hippocampus, and contextual fear conditioning. *Cognit. Affect. Behav. Neurosci.* **1**, 66–82 (2001).
47. R. A. Rescorla, "Configural" conditioning in discrete-trial bar pressing. *J. Comp. Physiol. Psychol.* **79**, 307–317 (1972).
48. V. Mante, D. Sussillo, K. V. Shenoy, W. T. Newsome, Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature* **503**, 78–84 (2013).
49. H. Davis, "Logical transitivity in animals" in *Cognitive Aspects of Stimulus Control*, W. K. Honig, J. G. Fetterman, Eds. (Psychology Press, 1992).
50. Y. Cho, L. Saul, "Kernel methods for deep learning" in *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, A. Culotta, Eds. (Curran Associates, Inc., Red Hook, NY, 2009), vol. 22, pp. 342–350.
51. A. Saxe, J. McClelland, S. Ganguli, "Exact solutions to the nonlinear dynamics of learning in deep linear neural networks" in *International Conference on Learning Representations* (2014).
52. S. Gunasekar, J. Lee, D. Soudry, N. Srebro, "Characterizing implicit bias in terms of optimization geometry" in *International Conference on Machine Learning* (PMLR, 2018), pp. 1832–1841.
53. Z. Ji, M. Dudík, R. E. Schapire, M. Telgarsky, "Gradient descent follows the regularization path for general losses" in *Conference on Learning Theory* (PMLR, 2020), pp. 2109–2136.
54. K. W. Spence, The nature of the response in discrimination learning. *Psychol. Rev.* **59**, 89–93 (1952).
55. J. A. Dusek, H. Eichenbaum, The hippocampus and transverse patterning guided by olfactory cues. *Behav. Neurosci.* **112**, 762–771 (1998).
56. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**, 436–444 (2015).
57. B. A. Richards *et al*., A deep learning framework for neuroscience. *Nat. Neurosci.* **22**, 1761–1770 (2019).
58. A. Saxe, S. Nelli, C. Summerfield, If deep learning is the answer, what is the question? *Nat. Rev. Neurosci.* **22**, 55–67 (2021).
59. A. Jacot, F. Gabriel, C. Hongler, "Neural tangent kernel: Convergence and generalization in neural networks" in *Advances in Neural Information Processing Systems*, S. Bengio *et al*., Eds. (Curran Associates, Inc., Red Hook, NY, 2018), vol. 31, pp. 8580–8589.
60. L. Chizat, E. Oyallon, F. Bach, "On lazy training in differentiable programming" in *Advances in Neural Information Processing Systems*, H. Wallach *et al*., Eds. (Curran Associates, Inc., Red Hook, NY, 2019), vol. 32, pp. 2937–2947.
61. S. Fort *et al*., "Deep learning versus kernel learning: An empirical study of loss landscape geometry and the time evolution of the Neural Tangent Kernel" in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, H. Lin, Eds. (Curran Associates, Inc., Red Hook, NY, 2020), vol. 33, pp. 5850–5861.
62. N. Vyas, Y. Bansal, P. Nakkiran, Limitations of the NTK for understanding generalization in deep learning. arXiv [Preprint] (2022). https://arxiv.org/abs/2206.10012 (Accessed 14 June 2024).
63. J. Poort *et al*., Learning enhances sensory and multiple non-sensory representations in primary visual cortex. *Neuron* **86**, 1478–1490 (2015).
64. T. Flesch, K. Juechems, T. Dumbalska, A. Saxe, C. Summerfield, Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron* **110**, 1258–1270.e11 (2022).
65. F. Van Opstal, W. Fias, P. Peigneux, T. Verguts, The neural representation of extensively trained ordered sequences. *NeuroImage* **47**, 367–375 (2009).
66. E. C. Hinton, S. Dymond, U. von Hecker, C. J. Evans, Neural correlates of relational reasoning and the symbolic distance effect: Involvement of parietal cortex. *Neuroscience* **168**, 138–148 (2010).
67. A. O. Constantinescu, J. X. O'Reilly, T. E. J. Behrens, Organizing conceptual knowledge in humans with a gridlike code. *Science* **352**, 1464–1468 (2016).
68. J. C. Whittington *et al*., The Tolman–Eichenbaum machine: Unifying space and relational memory through generalization in the hippocampal formation. *Cell* **183**, 1249–1263 (2020).
69. K. Lyu, J. Li, "Gradient descent maximizes the margin of homogeneous neural networks" in *International Conference on Learning Representations* (2020).
70. L. Chizat, F. Bach, "Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss" in *Proceedings of Thirty Third Conference on Learning Theory* (PMLR, 2020), pp. 1305–1338.
71. N. Razin, N. Cohen, "Implicit regularization in deep learning may not be explainable by norms" in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, H. Lin, Eds. (Curran Associates, Inc., Red Hook, NY, 2020), vol. 33, pp. 21174–21187.
72. B. Woodworth *et al*., "Kernel and rich regimes in overparametrized models" in *Conference on Learning Theory* (2020), pp. 3635–3673.
73. P. Savarese, I. Evron, D. Soudry, N. Srebro, "How do infinite width bounded norm networks look in function space?" in *Proceedings of the Thirty-Second Conference on Learning Theory* (2019), vol. 99, pp. 2667–2690.
74. G. Ongie, R. Willett, D. Soudry, N. Srebro, "A function space view of bounded norm infinite width ReLU nets: The multivariate case" in *International Conference on Learning Representations* (2020). https://openreview.net/pdf?id=H1lNPxHKDH. Accessed 21 June 2024.
75. C. Kemp, J. B. Tenenbaum, The discovery of structural form. *Proc. Natl. Acad. Sci. U.S.A.* **105**, 10687–10692 (2008).
76. D. J. Gillan, Reasoning in the chimpanzee: II. Transitive inference. *J. Exp. Psychol.: Anim. Behav. Process.* **7**, 150–164 (1981).
77. L. F. Jacobs, From movement to transitivity: The role of hippocampal parallel maps in configural learning. *Rev. Neurosci.* **17**, 99–109 (2006).
78. C. B. De Soto, M. London, S. Handel, Social reasoning and spatial paralogic. *J. Person. Soc. Psychol.* **2**, 513–521 (1965).
79. R. C. Wilson, Y. K. Takahashi, G. Schoenbaum, Y. Niv, Orbitofrontal cortex as a cognitive map of task space. *Neuron* **81**, 267–279 (2014).

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.

80. T. E. J. Behrens *et al*., What is a cognitive map? Organizing knowledge for flexible behavior. *Neuron* **100**, 490–509 (2018).
81. J. L. McClelland *et al*., Letting structure emerge: Connectionist and dynamical systems approaches to cognition. *Trend. Cognit. Sci.* **14**, 348–356 (2010).
82. A. M. Saxe, J. L. McClelland, S. Ganguli, A mathematical theory of semantic development in deep neural networks. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 11537–11546 (2019).
83. J. L. McClelland, B. L. McNaughton, R. C. O'Reilly, Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. *Psychol. Rev.* **102**, 419–457 (1995).
84. D. Kumaran, D. Hassabis, J. L. McClelland, What learning systems do intelligent agents need? Complementary learning systems theory updated. *Trend. Cognit. Sci.* **20**, 512–534 (2016).
85. G. Jensen, F. Munoz, A. Meaney, H. S. Terrace, V. P. Ferrera, Transitive inference after minimal training in rhesus macaques (*Macaca mulatta*). *J. Exp. Psychol.: Anim. Learn. Cognit.* **47**, 464 (2021).
86. D. Zeithamova, M. Schlichting, A. Preston, The hippocampus and inferential reasoning: Building memories to navigate future decisions. *Front. Hum. Neurosci.* **6**, 70 (2012).
87. D. Shohamy, N. D. Daw, Integrating memories to guide decisions. *Curr. Opin. Behav. Sci.* **5**, 85–90 (2015).
88. Z. Kurth-Nelson *et al*., Replay and compositional computation. *Neuron* **111**, 454–469 (2023).
89. D. Kumaran, J. L. McClelland, Generalization through the recurrent interaction of episodic memories: A model of the hippocampal system. *Psychol. Rev.* **119**, 573–616 (2012).
90. J. Russin, M. Zolfaghar, S. A. Park, E. Boorman, R. C. O'Reilly, Complementary structure–learning neural networks for relational reasoning. *Annu. Conf. Cognit. Sci. Soc.* **2021**, 1560–1566 (2021).
91. T. Miconi, K. Kay, An active neural mechanism for relational learning and fast knowledge reassembly. bioRxiv [Preprint] (2023). https://doi.org/10.1101/2023.07.27.550739 (Accessed 14 June 2024).
92. J. X. Wang *et al*., Prefrontal cortex as a meta-reinforcement learning system. *Nat. Neurosci.* **21**, 860–868 (2018).
93. J. C. R. Whittington, D. McCaffary, J. J. W. Bakermans, T. E. J. Behrens, How to build a cognitive map. *Nat. Neurosci.* **25**, 1257–1272 (2022).
94. L. Ambrogioni, H. F. Ólafsdóttir, Rethinking the hippocampal cognitive map as a meta-learning computational module. *Trend. Cognit. Sci.* **27**, 702–712 (2023).
95. G. Winocur, M. Moscovitch, B. Bontempi, Memory formation and long-term retention in humans and animals: Convergence towards a transformation account of hippocampal–neocortical interactions. *Neuropsychologia* **48**, 2339–2356 (2010).
96. M. Telgarsky, Y. Singer, A primal-dual convergence analysis of boosting. *J. Mach. Learn. Res.* **13**, 561–606 (2012).
97. A. Kumar *et al*., "DR3: Value-based deep reinforcement learning requires explicit regularization" in *International Conference on Learning Representations* (2022).
98. A. B. Bond, A. C. Kamil, R. P. Balda, Social complexity and transitive inference in corvids. *Anim. Behav.* **65**, 479–487 (2003).
99. B. McGonigle, M. Chalmers, Monkeys are rational! *Q. J. Exp. Psychol., Sec. B* **45**, 189–228 (1992).
100. G. S. Halford, Can young children integrate premises in transitivity and serial order tasks? *Cognit. Psychol.* **16**, 65–93 (1984).
101. F. R. Treichler, D. Van Tilburg, Concurrent conditional discrimination tests of transitive inference by macaque monkeys: List linking. *J. Exp. Psychol.: Anim. Behav. Process.* **22**, 105 (1996).
102. O. F. Lazareva, E. A. Wasserman, Transitive inference in pigeons: Measuring the associative values of Stimuli B and D. *Behav. Process.* **89**, 244–255 (2012).
103. H. Eichenbaum, *The Cognitive Neuroscience of Memory: An Introduction* (Oxford University Press, 2011).
104. N. Kriegeskorte, M. Mur, P. A. Bandettini, Representational similarity analysis–Connecting the branches of systems neuroscience. *Front. Syst. Neurosci.* **2**, 4 (2008).
105. J. A. Fodor, Z. W. Pylyshyn, Connectionism and cognitive architecture: A critical analysis. *Cognition* **28**, 3–71 (1988).
106. B. M. Lake, R. Salakhutdinov, J. B. Tenenbaum, Human-level concept learning through probabilistic program induction. *Science* **350**, 1332–1338 (2015).
107. E. M. Bender, A. Koller, "Climbing towards NLU: On meaning, form, and understanding in the age of data" in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 5185–5198.
108. C. Summerfield, F. Luyckx, H. Sheahan, Structure learning and the posterior parietal cortex. *Progr. Neurobiol.* **184**, 101717 (2020).
109. J. Kaplan *et al*., Scaling laws for neural language models. arXiv [Preprint] (2020). https://arxiv.org/abs/2001.08361 (Accessed 14 June 2024).
110. T. Brown *et al*., "Language models are few-shot learners" in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, H. Lin, Eds. (Curran Associates, Inc., 2020), vol. 33, pp. 1877–1901.
111. N. Kandpal, H. Deng, A. Roberts, E. Wallace, C. Raffel, "Large language models struggle to learn long-tail knowledge" in *International Conference on Machine Learning* (PMLR, 2023), pp. 15696–15707.
112. R. Grosse *et al*., Studying large language model generalization with influence functions. arXiv [Preprint] (2023). https://arxiv.org/abs/2308.03296 (Accessed 14 June 2024).
113. P. Smolensky, On the proper treatment of connectionism. *Behav. Brain Sci.* **11**, 1–23 (1988).
114. B. M. Lake, T. D. Ullman, J. B. Tenenbaum, S. J. Gershman, Building machines that learn and think like people. *Behav. Brain Sci.* **40**, e253 (2017).
115. F. Hill, A. Santoro, D. Barrett, A. Morcos, T. Lillicrap, "Learning to make analogies by contrasting abstract relational structure" in *International Conference on Learning Representations* (2019).
116. A. Saxe, S. Sodhani, S. J. Lewallen, "The neural race reduction: Dynamics of abstraction in gated networks" in *Proceedings of the 39th International Conference on Machine Learning* (PMLR, 2022), pp. 19287–19309.
117. S. M. Frankland, J. D. Greene, Concepts and compositionality. In search of the brain's language of thought. *Annu. Rev. Psychol.* **71**, 273–303 (2020).
118. M. Lepori, T. Serre, E. Pavlick, "Break it down: Evidence for structural compositionality in neural networks" in *Advances in Neural Information Processing Systems*, A. Oh *et al*., Eds. (Curran Associates, Inc., Red Hook, NY, 2024), vol. 36, pp. 42623–42660.
119. S. Lippl, K. Kay, G. Jensen, V. P. Ferrera, L. F. Abbott, Code for "A mathematical theory of relational generalization in transitive inference." Zenodo. https://doi.org/10.5281/zenodo.12172070. Deposited on 19 June 2024.

Downloaded from https://www.pnas.org by UC San Francisco on July 14, 2024 from IP address 128.218.42.41.



Supporting Information for

A mathematical theory of relational generalization in transitive inference

Samuel Lippl, Kenneth Kay, Greg Jensen, Vincent P. Ferrera, and L.F. Abbott

Samuel Lippl.
E-mail: samuel.lippl@columbia.edu

This PDF file includes:

Supporting text
Figs. S1 to S11
Tables S1 to S2
SI References

Supporting Information Text

S1. Mathematical companion

In this section, we prove the mathematical statements made in the main text and provide additional details on the analysis of model behavior.

- Section S1.A extends the analysis of additive models presented in Section "An additive representation yields transitive generalization" (p. 3).
- Section S1.B defines the conjunctivity factor and determines the possible range of values it can take on.
- Section S1.C determines the conjunctivity factor of a broad range of possible networks, clarifying the connection between network archiectures and TI behavior.
- Section S1.D characterizes the TI behavior of models implementing norm minimization, gradient descent, and ridge regression.
- Section S1.E uses this theory to predict the behavior of neural networks in the lazy regime.
- Section S1.F analyzes the behavior of models trained through gradient flow on transverse patterning.

To leave the section relatively compact, we outsource the details for some of the more involved proofs to Section S4.

S1.A. Models with additive representations and choice bias can make mistakes in TI but generalize correctly on average. Section "An additive representation yields transitive generalization" (p. 3) considered a model with no choice bias (i.e. $f(X, X) = 0$). Here we discuss the possible ramifications of a choice bias in additive representations. As a reminder, we consider a model

$$f(X, Y) = w \circ g_1(X) + w \circ g_2(Y).$$

We can understand the model's decision making in terms of two "implicit" ranks

$$r_1(X) := w \circ g_1(X), \quad r_2(Y) := -w \circ g_2(Y),$$

so that

$$f(X, Y) = r_1(X) - r_2(Y).$$

f will classify adjacent items i and $i + 1$, $i \in \{1, \ldots, n - 1\}$, correctly if and only if those ranks satisfy

$$r_1(i) > r_2(i + 1), \quad r_2(i) > r_1(i + 1).$$

We can characterize this condition more easily by defining two sequences: one starting at $i = 1$ with r_1 (we call this one s_1) and one starting at $i = 1$ with r_2 (we call this one s_2). At each item those sequences flip whether they specify the rank of r_1 or r_2. That is: s_1 specifies r_1 at odd items and r_2 at even items and s_2 behaves in the exact opposite way:

$$r_1(i) = \begin{cases} s_1(i) & \text{if } i \text{ odd}, \\ s_2(i) & \text{if } i \text{ even}, \end{cases} \quad r_2(i) = \begin{cases} s_2(i) & \text{if } i \text{ odd}, \\ s_1(i) & \text{if } i \text{ even}. \end{cases} \tag{S1}$$

By construction, f will classify adjacent items if and only if both s_1 and s_2 are monotonically decreasing. For items i, j with an odd symbolic distance, this immediately implies that f must also correctly order those, as $r_1(i)$ and $r_2(j)$ as well as $r_2(i)$ and $r_1(j)$ will be assigned to the same sequence and therefore be monotonic. For items with an even symbolic distance, however, the two ranks will be mapped to different sequences. The models do not have to generalize transitively and can generalize pairs of items incorrectly for arbitrarily large symbolic distances.

We leave with two notes. First, the average margin

$$\overline{f}(i, j) := \frac{1}{2} \left(f(i, j) - f(j, i) \right), \tag{S2}$$

has no choice bias even when f itself does (i.e. $\overline{f}(i, i) = 0$). As a result, this average margin still generalizes transitively even when f itself does not in generalize. Second, if the model classifies pairs (i, j) correctly for some even symbolic distance, it must also do so for any higher symbolic distance.

S1.B. The conjunctivity factor. We consider a general representation

$$\phi(i,j) \in \mathbb{R}^h, \quad h \in \mathbb{N},$$

where i and j denotes the numerical rank of the two presented items X, Y. We then consider the representational similarity (measured by the dot product) between two trials (i,j) and (i',j'):

$$K(i,j;i',j') := \langle \phi(i,j), \phi(i',j') \rangle. \tag{S3}$$

We assume that there are three values K can take, κ_s if $i = i'$ and $j = j'$, κ_o if either $i = i'$ or $j = j'$, and κ_d if $i \neq i'$ and $j \neq j'$. α is then defined as

$$\alpha := 1 - 2\frac{\kappa_o - \kappa_d}{\kappa_s - \kappa_d}. \tag{S4}$$

We assume $\kappa_s \neq \kappa_d$. If $\kappa_s = \kappa_d$, then $\kappa_s = \kappa_o$, meaning that $\phi(i,j)$ is constant and unable to learn anything.

Lemma S1.1. *α is restricted to the range $[0,3]$. For arbitrary n, α can take on values in the range $[0,1]$.*

Proof. For compactness, we denote κ_o by o, κ_d by d, and κ_s by s. We then consider the Gram Matrix of all possible combinations of the first three items, i.e.

$$z := ((1,2),(2,1),(1,3),(3,1),(2,3),(3,2)), \tag{S5}$$

$$G := (K(z_i; z_j))_{ij} = \begin{pmatrix} s & d & o & d & d & o \\ d & s & d & o & o & d \\ o & d & s & d & o & d \\ d & o & d & s & d & o \\ d & o & o & d & s & d \\ o & d & d & o & d & s \end{pmatrix}. \tag{S6}$$

G is positive definite (as it is a Gram matrix), implying that all its eigenvalues are positive.

Using Mathematica, we determine two of these eigenvalues as

$$\lambda_1 = s + d - 2o, \quad \lambda_2 = s + o - 2d. \tag{S7}$$

We can transform

$$\alpha = \frac{s + d - 2o}{s - d}.$$

As $s - d \geq 0$ (by basic properties of a kernel), $\lambda_1 \geq 0$ implies that $\alpha \geq 0$. Further, $\lambda_2 \geq 0$ implies that

$$0 \leq s + o - 2d = (s - d) + (o - d),$$

and therefore

$$0 \leq 1 + \frac{o - d}{s - d},$$

implying that $\alpha \leq 3$. Finally, α can take on arbitrary values in $[0,1]$ as $\alpha = 0$ corresponds to an itemic encoding and $\alpha = 1$ corresponds to a onehot encoding. By linear combinations of those encodings, we can achieve arbitrary values in-between. □

Remark S1.2. *The lemma leaves open which values in the range $(1,3]$ are attainable. In Section S4.B, we provide a construction that can attain $\alpha > 1$ for all n. Notably, this indicates a feature basis that encodes overlapping item pairs as less similar than distinct item pairs. For $n = 3$ attains $\alpha = 3$. As n increases, however, α decreases, gradually approaching 1. It remains an open question whether there are features that, at larger n attain higher values for α than our construction.*

In subsequent sections, we will constrain ourselves to $\alpha \in [0,1]$.

S1.C. Impact of nonlinearity, depth, and connectivity structure on a deep neural network's conjunctivity factor. We here compute the conjunctivity factor for a broad range of hidden representations, to better understand how it is impacted by different choices in architecture.

S1.C.1. Densely connected neural networks. This corresponds to a standard densely connected neural network with layerwise weights $W^{(l)}$:

$$h^{(0)} = z, \quad h^{(l+1)} = \frac{1}{\sqrt{H_{l+1}}}\phi(W^{(l)}h^{(l)}), \quad W^{(l)} \in \mathbb{R}^{H_{l+1}\times H_l}. \tag{S8}$$

where ϕ is some nonlinearity. We assume that these weights are sampled from a rotationally invariant distribution with some variance σ_l^2 (e.g. a normal distribution). Further, we assume that ϕ is structured as

$$\phi(x) := A\min(0,x) + B\max(0,x), \tag{S9}$$

which has as special cases ReLU ($A = 0$, $B = 1$, 1), leaky ReLU ($A < 1$, $B = 1$, 2), and the absolute value ($A = -1$, $B = 1$). As shown below, the conjunctivity factor of $h^{(l)}$ is invariant to ϕ's "magnitude"

$$m := A^2 + B^2, \tag{S10}$$

but depends on its "relative slope"

$$\rho := \frac{(B-A)^2}{A^2+B^2}. \tag{S11}$$

In particular, $\rho = 1$ for ReLU. (Note that similar analytical expressions could be derived for a broad range of other nonlinearities.)

We are now interested in the trial-by-trial similarity in each network layer. The expected similarity (which the empirical similarity will converge to in the large-width limit) is

$$K^{(l+1)}(z,z') := \mathbb{E}\left[\langle h^{(l+1)}(z), h^{(l+1)}(z')\rangle\right] = \mathbb{E}_w\left[\phi(w^T h^{(l)}(z))\phi(w^T h^{(l)}(z'))\right]. \tag{S12}$$

Our goal is to express this expression in terms of $K^{(l)}$, the kernel of the previous layer in order to derive a recursive formula.

Notably, ϕ is homogeneous and therefore

$$\phi(w^T h^{(l)}(z)) = \|w\|_2\|h^{(l)}(z)\|_2\phi(\hat{w}^T\hat{h}^{(l)}(z)), \quad \hat{w} := w/\|w\|_2, \quad \hat{h}^{(l)}(z) := h^{(l)}(z)/\|h^{(l)}(z)\|_2 \tag{S13}$$

As a result,

$$\mathbb{E}_w\left[\phi(w^T h^{(l)}(z))\phi(w^T h^{(l)}(z'))\right] = \mathbb{E}_w\left[\|w\|_2^2\right]\|h^{(l)}(z)\|_2\|h^{(l)}(z')\|_2\mathbb{E}_w\left[\phi(\hat{w}^T\hat{h}^{(l)}(z))\phi(\hat{w}^T\hat{h}^{(l)}(z'))\right] \tag{S14}$$

Notably, prior work (3–6, e.g. see 6, Table 2) has found that

$$\mathbb{E}_w\left[\phi(\hat{w}^T\hat{h}^{(l)}(z))\phi(\hat{w}^T\hat{h}^{(l)}(z'))\right] = k\left(\hat{h}^{(l)}(z)^T\hat{h}^{(l)}(z')\right), \tag{S15}$$

where

$$k(u) = \frac{(B-A)^2}{2\pi}\left(\sqrt{1-u^2} + (\pi - \cos^{-1}(u))u\right) + ABu. \tag{S16}$$

This means that

$$\mathbb{E}_w\left[\phi(w^T h^{(l)}(z))\phi(w^T h^{(l)}(z'))\right] = \sigma_l^2\|h^{(l)}(z)\|_2\|h^{(l)}(z')\|_2 k\left(\hat{K}^{(l)}(z,z')\right), \tag{S17}$$

where

$$\|h^{(l)}(z)\|_2^2 = K^{(l)}(z,z), \quad \hat{K}^{(l)}(z,z') = \frac{K^{(l)}(z,z')}{\|h^{(l)}(z)\|_2\|h^{(l)}(z')\|_2}. \tag{S18}$$

Defining

$$\begin{aligned}\kappa_s^{(l)} &:= K^{(l)}((x,y),(x,y)),\\ \kappa_o^{(l)} &:= K^{(l)}((x,y),(x',y)) = K^{(l)}((x,y),(x,y')),\\ \kappa_d^{(l)} &:= K^{(l)}((x,y),(x',y')),\end{aligned} \tag{S19}$$

we infer

$$\kappa_s^{(l+1)} = \sigma_{l+1}^2\kappa_s^{(l)}k(1) = \sigma_{l+1}^2\kappa_s^{(l)}\left(\frac{(B-A)^2}{2} + AB\right) = \sigma_{l+1}^2\kappa_s^{(l)}\frac{m}{2}. \tag{S20}$$

Further,

$$\kappa_o^{(l+1)} = \sigma_{l+1}^2\kappa_s^{(l)}k\left(\frac{\kappa_o^{(l)}}{\kappa_s^{(l)}}\right), \quad \kappa_d^{(l+1)} = \sigma_{l+1}^2\kappa_s^{(l)}k\left(\frac{\kappa_d^{(l)}}{\kappa_s^{(l)}}\right). \tag{S21}$$

Defining

$$\begin{aligned}\tilde{k}(u) := \frac{2}{m}k(u) &= \rho\left(\frac{\sqrt{1-u^2}}{\pi} + \left(1 - \frac{\cos^{-1}(u)}{\pi}\right)u\right) + (1-\rho)u =\\ &= u + \rho\frac{\sqrt{1-u^2} - \cos^{-1}(u)u}{\pi},\end{aligned} \tag{S22}$$

and

$$\hat{\kappa}_{\Delta}^{(l)} := \frac{\kappa_{\Delta}^{(l)}}{\kappa_s^{(l)}} \tag{S23}$$

we infer

$$\hat{\kappa}_o^{(l+1)} = \tilde{k}\left(\hat{\kappa}_o^{(l)}\right) = \tilde{k}^{(l+1)}\left(\hat{\kappa}_o^{(0)}\right), \quad \hat{\kappa}_d^{(l+1)} = \tilde{k}\left(\hat{\kappa}_d^{(l)}\right) = \tilde{k}^{(l+1)}\left(\hat{\kappa}_d^{(0)}\right), \tag{S24}$$

where $\tilde{k}^{(l+1)}$ denotes $l+1$ applications of $\tilde{k}$. This allows us to recursively compute

$$\alpha = 1 - 2\frac{\hat{\kappa}_o^{(L)} - \hat{\kappa}_d^{(L)}}{1 - \hat{\kappa}_d^{(L)}}. \tag{S25}$$

As a special case, we consider input items x, which are all orthogonal to each other, meaning that

$$\kappa_s^{(0)} = 1, \quad \kappa_o^{(0)} = \tfrac{1}{2}, \quad \kappa_d^{(0)} = 0. \tag{S26}$$

and compute the similarities of the first hidden layer:

$$\hat{\kappa}_o^{(1)} = \tilde{k}(1/2) = \frac{1}{2} + \rho\left(\frac{\sqrt{3}}{2\pi} - \frac{1}{6}\right), \quad \hat{\kappa}_d^{(1)} = \tilde{k}(0) = \rho\frac{1}{\pi}. \tag{S27}$$

The conjunctivity factor α in this case is given by

$$\alpha = 1 - 2\frac{\hat{\kappa}_o^{(1)} - \hat{\kappa}_d^{(1)}}{1 - \hat{\kappa}_d^{(1)}} = \frac{1 + \hat{\kappa}_d^{(1)} - 2\hat{\kappa}_o^{(1)}}{1 - \hat{\kappa}_d^{(1)}} = \rho\frac{\pi/3 + 1 - \sqrt{3}}{\pi - \rho}. \tag{S28}$$

For ReLU (i.e. $\rho = 1$), $\alpha \approx 0.15$.

S1.C.2. Neural networks with additive nodes. We can now extend these formulas to networks with partly additive nodes. In particular, we assume that the fraction $(1-\lambda)$ is fully connected and the fraction λ is only connected to one of the two items, though followed by the same nonlinearity. This part of the representation has the similarities

$$\hat{\mu}_d^{(l)} = \hat{\kappa}_d^{(l)}, \quad \hat{\mu}_o^{(l)} = \frac{1 + \hat{\kappa}_d^{(l)}}{2}, \quad \hat{\mu}_s^{(l)} = 1, \tag{S29}$$

and so the new normalized dissimilarities are given by

$$\tilde{\kappa}_o^{(l)} = (1-\lambda)\hat{\kappa}_o^{(l)} + \frac{\lambda}{2}(\hat{\kappa}_d^{(l)} + 1), \quad \tilde{\kappa}_d^{(l)} = \hat{\kappa}_d^{(l)}. \tag{S30}$$

This, in turn, means that

$$\tilde{\alpha}^{(l)} := 1 - 2\frac{\tilde{\kappa}_o^{(l)} - \tilde{\kappa}_d^{(l)}}{1 - \tilde{\kappa}_d^{(l)}} = \frac{1 + \tilde{\kappa}_d^{(1)} - 2\tilde{\kappa}_o^{(1)}}{1 - \tilde{\kappa}_d^{(1)}} = (1-\lambda)\frac{(1 + \hat{\kappa}_d^{(l)} - 2\hat{\kappa}_o^{(l)})}{1 - \hat{\kappa_d}^{(l)}}. \tag{S31}$$

Predictably, for a single layer, the additive units therefore simply reduce the conjunctivity factor by a proportional factor. For multiple layers, this reduction has ramifications on the densely connected units as well, as these units draw upon both the additive and densely connected units. In fact, in the limit of infinite depth, any neural network with some proportion of additive units will approach a fully compositional representation:

Lemma S1.3. *For a nonlinearity ϕ with relative slope ρ and a proportion of additive units λ, the infinite-depth conjunctivity factor*

$$\alpha^{(\infty)} := \lim_{l\to\infty} \alpha^{(l)}. \tag{S32}$$

is given by

$$\alpha^{(\infty)} = \begin{cases} 0 & \text{if } \lambda > 0 \vee \rho = 0, \\ 1 & \text{else.} \end{cases} \tag{S33}$$

Proof. We prove the lemma in Section S4.E. □

S1.D. Standard statistical learning models have a transitive inductive bias. In this section, we derive the TI behavior of different standard statistical learning models. Section S1.D.1 describes the general setup of transitive inference and our learning models. Section S1.D.2 then provides a general intuition for why these models implement an emergent ranking system and how we can compute these ranks. The subsequent sections go on to characterize model behavior for ridge regression (Section S1.D.3) and gradient flow (Section S1.D.4).

S1.D.1. Setup. We consider $n \in \mathbb{N}$ items $(x^{(i)})_{i=1,\dots,n}$, $x^{(i)} \in \mathbb{R}^d$. The task presents two of those items together and we denote this by

$$x^{(i,j)} = \begin{pmatrix} x^{(i)} \\ x^{(j)} \end{pmatrix} \in \mathbb{R}^{2d}. \tag{S34}$$

The corresponding label is given by

$$y^{(i,j)} = \begin{cases} 1 & \text{if } i < j, \\ -1 & \text{else,} \end{cases} \tag{S35}$$

and the training dataset consists of all adjacent items, i.e.

$$\mathcal{D} = \left\{ (x^{(i,i+1)}, 1) | i = 1, \dots, n-1 \right\} \cup \left\{ (x^{(i+1,i)}, -1) | i = 1, \dots, n-1 \right\}. \tag{S36}$$

In matrix form, we denote this dataset as

$$X = \begin{pmatrix} x^{(1,2)} \\ \vdots \\ x^{(n-1,n)} \\ x^{(2,1)} \\ \vdots \\ x^{(n,n-1)} \end{pmatrix} \in \mathbb{R}^{2(n-1)\times h}, \quad y = \begin{pmatrix} y^{(1,2)} \\ \vdots \\ y^{(n-1,n)} \\ y^{(2,1)} \\ \vdots \\ y^{(n,n-1)} \end{pmatrix} \in \mathbb{R}^{2(n-1)} \tag{S37}$$

We consider a set of features

$$\phi : \mathbb{R}^{2d} \to \mathbb{R}^h, \quad h \in \mathbb{N}, \tag{S38}$$

and a linear model of those latent features that predicts the output, i.e.

$$\hat{y}^{(i,j)} = \langle w, \phi(x^{(i,j)}) \rangle. \tag{S39}$$

S1.D.2. General intuition. Many standard statistical learning models (in particular the ones we consider below) only depend on their input representation through its trial-by-trial similarity (or induced kernel, see also Section S1.B, 7):

$$K(x, x') = \langle \phi(x), \phi(x') \rangle. \tag{S40}$$

In particular, the minimal norm solution (described in Section "Norm minimization and partly conjunctive representations yield transitive generalization"), can be described in terms of the "dual coefficients"

$$a = K^{-1} y, \tag{S41}$$

where K is the trial-by-trial similarity of the training dataset:

$$K = \phi(X)^T \phi(X) \in \mathbb{R}^{2(n-1)\times 2(n-1)}. \tag{S42}$$

Each dual coefficient corresponds to a particular training data point (i, j) and we denote it by $a_{(i,j)}$. The model behavior on a new data point $x^{(i,j)}$ is then given by

$$f(x^{(i,j)}) = \langle k^{(\text{test})}_{(i,j)}, a \rangle, \quad k^{(\text{test})}_{(i,j)} := \phi(X)^T \phi(x^{(i,j)}). \tag{S43}$$

Importantly, the similarity between test data points and training data points can only take on two possible values: κ_o and κ_d. Specifically, the dual coefficients $a_{(i,i+1)}$, $a_{(i,i-1)}$, $a_{(j-1,j)}$, and $a_{(j+1,j)}$ are overlapping with the test input (i, j) and therefore have similarity κ_o. (Note that if $i, j \in \{1, n\}$, some of these dual coefficients will not exist. To handle this special cases, we simply set the corresponding dual coefficients to zero. For example, if $i = 0$, $a_{(i,i-1)} = a_{(1,0)}$ corresponds to a non-existent data point and is set to zero.) All other data points have similarity κ_d. Taken together, we can express model behavior as

$$f(x^{(i,j)}) = \kappa_d \sum a + (\kappa_o - \kappa_d)(a_{(i,i+1)} + a_{(i,i-1)} + a_{(j-1,j)} + a_{(j+1,j)}). \tag{S44}$$

Crucially, this behavior is a sum of values dependent on i and values dependent on j — just like we observed for a purely additive representation. This immediately implies that the model implements a ranking system. Specifically we define

$$r_1(i) = (\kappa_o - \kappa_d) \left(a_{(i,i+1)} + a_{(i,i-1)} \right), \quad r_2(j) = -(\kappa_o - \kappa_d) \left(a_{(j-1,j)} + a_{(j+1,j)} \right), \tag{S45}$$

yielding

$$f(x^{(i,j)}) = r_1(i) - r_2(j). \tag{S46}$$

To see whether the model generalizes, we need to determine whether the ranks are ordered correctly. To do so, we analytically solve the inverse problem Eq. (S41). This is described in detail in the next section, which considers a generalization of the problem considered here. As a brief summary, we rely on previous results on inverses for banded diagonal matrices (8) to compute the inverse matrix. This inverse is expressed in terms of hyperbolic functions. We then use hyperbolic identities in a few different ways to compute a and, finally r_1 and r_2 from this. This reveals that $r_1(i) = r_2(i)$ and further gives rise to the analytical expression provided in the main text.

S1.D.3. Ridge regression. Here we assume that the weights are learned through ridge regression. Specifically, our loss function is given by the mean squared error over the training set:

$$L(w) = \sum_{(x,y)\in\mathcal{D}} \ell(y, \langle w, \phi(x)\rangle) = \sum_{i=1}^{n-1} \ell(1, \langle w, \phi(x^{(i,i+1)})\rangle) + \ell(-1, \langle w, \phi(x^{(i+1,i)})\rangle), \tag{S47}$$

where $\ell(y, \hat{y}) = (y - \hat{y})^2$. The weights w are then determined by minimizing the optimization problem

$$L(w) + \tfrac{1}{c}\|w\|_2^2, \tag{S48}$$

where c is the inverse strength of the regularization. The corresponding dual problem is given by

$$a := (K + \tfrac{1}{c}I)^{-1}y, \tag{S49}$$

where K is the cross-sample similarity,

$$K \in \mathbb{R}^{(2(n-1))^2}, \quad K_{ij} = \phi(X)^T\phi(X) = \begin{cases} \kappa_s & \text{if } i = j, \\ \kappa_o & \text{if } |i-j| \in \{n-2, n\}, \\ \kappa_d & \text{else.} \end{cases} \tag{S50}$$

Note that for $\frac{1}{c} = 0$, we obtain the special case from the previous section (Eq. (S41)). Writing a as

$$a = \begin{pmatrix} a^{(1)} \\ a^{(2)} \end{pmatrix}, \quad a^{(1)} \in \mathbb{R}^{n-1}, \quad a^{(2)} \in \mathbb{R}^{n-1}, \tag{S51}$$

we can write $(K + \frac{1}{c}I)^{-1}a = y$ as

$$\begin{aligned} (\delta_s + \tfrac{1}{c})a_i^{(1)} + \delta_o(a_{i-1}^{(2)} + a_{i+1}^{(2)}) + \kappa_d\langle a\rangle = 1, \\ (\delta_s + \tfrac{1}{c})a_i^{(2)} + \delta_o(a_{i-1}^{(1)} + a_{i+1}^{(1)}) + \kappa_d\langle a\rangle = -1, \end{aligned} \tag{S52}$$

where we define $\delta_s = \kappa_s - \kappa_d$, $\delta_o = \kappa_o - \kappa_d$, $\langle a\rangle = \sum_{i=1}^{2(n-1)} a_i$ and, for notational simplicity, $a_0^{(j)} = a_n^{(j)} = 0$. This means that if $a^{(1)}$, $a^{(2)}$ are a solution to the equation, then so are

$$\tilde{a}^{(1)} := -a^{(2)}, \quad \tilde{a}^{(2)} := -a^{(1)}. \tag{S53}$$

Since (S49) is well-defined, the solution a must be unique and we can infer

$$a^{(1)} = -a^{(2)}. \tag{S54}$$

We thus define

$$\overline{a} := a^{(1)} = -a^{(2)}. \tag{S55}$$

Setting

$$\tilde{c} := c\delta_s, \tag{S56}$$

(S52) yields

$$\delta_s(1 + \tilde{c}^{-1})\overline{a} + \delta_o(\overline{a}_{i-1} + \overline{a}_{i+1}) = 1, \tag{S57}$$

as $\langle a\rangle = \langle a^{(1)}\rangle + \langle a^{(2)}\rangle = 0$ and therefore the last summand vanishes. We then divide by δ_s on both sides; as $\frac{\delta_o}{\delta_s} = \frac{1-\alpha}{2}$, this results in

$$(1 + \tilde{c}^{-1})\overline{a}_i - \tfrac{1-\alpha}{2}(\overline{a}_{i-1} + \overline{a}_{i+1}) = \frac{1}{\delta_s}, \tag{S58}$$

In matrix form, this results in

$$\frac{1}{\delta_s}(\tilde{K} + \tilde{c}^{-1}I)^{-1}1 = \overline{a}, \tag{S59}$$

where

$$\tilde{K} \in \mathbb{R}^{(n-1)\times(n-1)}, \quad \tilde{K}_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{1-\alpha}{2} & \text{if } |i-j| = 1, \\ 0 & \text{else.} \end{cases} \tag{S60}$$

The predictions of the fitted model can be expressed in terms of the dual coefficients by

$$f(x^{(j,k)}) = \sum_{(x,y)\in\mathcal{D}} a_x K(x, x^{(j,k)}) = \sum_{i=1}^{n-1} K(i, i+1; j, k)a_i^{(1)} + K(i+1, i; j, k)a_i^{(2)}. \tag{S61}$$

We can express this in terms of $\overline{a}$ as

$$f(x^{(j,k)}) = \sum_{i=1}^{n-1} K(i, i+1; j, k)\overline{a}_i - \sum_{i=1}^{n-1} K(i+1, i; j, k))\overline{a}_i. \tag{S62}$$

There are now two different cases: either $x^{(j,k)}$ is in the training data or not.

Case 1: Test data. If $x^{(j,k)}$ is not in the training data, all data points are either overlapping or distinct. In the first data point, a data point is overlapping if $i = j$ or $i = k - 1$ and therefore

$$\sum_{i=1}^{n-1} K(i, i+1; j, k)\overline{a}_i = \kappa_d \langle \overline{a} \rangle + \delta_o(\overline{a}_j + \overline{a}_{k-1}). \tag{S63}$$

(Note that one of these dual coefficients, as in the previous section, may not exist, if $i \in \{1, n\}$. To leave the expression simple, we define $\overline{a}_0 = \overline{a}_n = 0$, covering these cases as well. Analogously, the second sum amounts to

$$\sum_{i=1}^{n-1} K(i+1, i; j, k)\overline{a}_i = \kappa_d \langle \overline{a} \rangle + \delta_o(\overline{a}_{j-1} + \overline{a}_k), \tag{S64}$$

and overall, the model behavior is expressed as

$$f(x^{(j,k)}) = \underbrace{\delta_s \tfrac{1-\alpha}{2}}_{=\delta_o}(\overline{a}_j + \overline{a}_{k-1} - \overline{a}_k - \overline{a}_{j-1}). \tag{S65}$$

Case 2: Training data. If $x^{(j,k)}$ is in the training data, then either $k = j + 1$ or $k = j - 1$. In the first case, the first sum, has as one of its data points $(j, k) = (j, j + 1)$ itself. As it has no overlapping data points, this sum is expressed as

$$\sum_{i=1}^{n-1} K(i, i+1; j, k)\overline{a}_i = \kappa_d \langle \overline{a} \rangle + \delta_s \overline{a}_j. \tag{S66}$$

The second sum still has two overlapping data points as in the previous case and as a result, model behavior is given by

$$f(x^{(j,j+1)}) = \delta_s(\overline{a}_j - \tfrac{1-\alpha}{2}(\overline{a}_{j-1} + \overline{a}_{j+1})). \tag{S67}$$

The second case ($k = j - 1$) works analogously.

Taken together, the prediction made by $\overline{a}$ are expressed as

$$f(x_j, x_k) = \begin{cases} \delta_s(\overline{a}_j - \tfrac{1-\alpha}{2}(\overline{a}_{j-1} + \overline{a}_{j+1})) & \text{if } k = j+1, \\ -\delta_s(\overline{a}_k - \tfrac{1-\alpha}{2}(\overline{a}_{k-1} + \overline{a}_{k+1})) & \text{if } k = j-1, \\ \delta_s \tfrac{1-\alpha}{2}(\overline{a}_j + \overline{a}_{k-1} - \overline{a}_k - \overline{a}_{j-1}) & \text{else.} \end{cases} \tag{S68}$$

In particular, this makes apparent the emergent rank representation

$$r_j = \delta_s \tfrac{1-\alpha}{2}(\overline{a}_j - \overline{a}_{j-1}). \tag{S69}$$

To compute the specific rank representation, we need to invert $B := \tilde{K} + \tilde{c}I$. B is a tridiagonal matrix with diagonal $1 + \tilde{c}$ and off-diagonal $-\frac{1-\alpha}{2}$. The ratio between diagonal and off-diagonal is therefore given by

$$-2\tfrac{1+\tilde{c}^{-1}}{1-\alpha} \leq -2. \tag{S70}$$

As this ratio is smaller than -2, we can apply Theorem S4.1 to infer

$$B^{-1}_{ij} = \frac{\cosh\left((n - |j - i|)\lambda\right) - \cosh\left((n - i - j)\lambda\right)}{(1 - \alpha)\sinh(\lambda)\sinh(n\lambda)}, \quad \lambda := \operatorname{arccosh}\left(\frac{1 + \tilde{c}^{-1}}{1 - \alpha}\right). \tag{S71}$$

We can then use this to compute $\delta_s \overline{a} = B^{-1}1$. Using hyperbolic trigonometric identities, we can simplify these expression further, resulting in the following lemma:

Lemma S1.4. *For non-adjacent items,*

$$f(x_j, x_k) = r_j - r_k, \quad r_j = \frac{\sinh((\frac{n+1}{2} - j)\lambda)}{\sinh(\frac{n+1}{2}\lambda) - \sinh(\frac{n-1}{2}\lambda)}, \quad \lambda := \operatorname{arccosh}\left(\frac{1 + \tilde{c}^{-1}}{1 - \alpha}\right). \tag{S72}$$

For adjacent items,

$$f(x_j, x_{j+1}) = m \cdot 1 + (1 - m)(r_j - r_{j+1}), \quad m := \frac{\alpha}{\alpha + \tilde{c}^{-1}}. \tag{S73}$$

Proof. See Section S4.C. □

S1.D.4. Gradient flow dynamics. We now consider the case where the weights w are trained using gradient flow rather than ridge regression. Specifically, the loss function consists in

$$L(w) = \sum_{(x,y)\in\mathcal{D}} \ell(y\langle w, \phi(x)\rangle) = \sum_{i=1}^{n-1} \ell(\langle w, \phi(x^{(i,i+1)})\rangle) + \ell(-\langle w, \phi(x^{(i+1,i)})\rangle) \tag{S74}$$

and weights are updated using gradient flow:

$$\dot{w} = -\nabla L(w). \tag{S75}$$

Note that in contrast to the previous expression (Eq. (S113)), here we express the loss function as a function of one variable $\ell(y\hat{y})$. This is possible because y is a binary label, so the mean squared error, for example, turns into

$$\ell(u) = \tfrac{1}{2}(1-u)^2. \tag{S76}$$

More generally we consider any convex, montonically decreasing loss function $\ell(y\hat{y})$. Other examples include the crossentropy

$$\ell(u) = \log(1+\exp(-u)). \tag{S77}$$

To simplify our notation, we will denote

$$\phi^{(i,j)} := \phi(x^{(i,j)}). \tag{S78}$$

Our loss function is then given by

$$L(w) = \sum_{(x,y)\in\mathcal{D}} \ell(y\langle w, \phi(x)) = \sum_{i=1}^{n-1} \ell(\langle w, \phi^{(i,i+1)}) + \ell(-\langle w, \phi^{(i+1,i)}). \tag{S79}$$

We now first consider the general case explaining how gradient descent gives rise to a terminal item effect and, in turn, how a terminal item effect on the training trials is connected to successful generalization on test trials. We then make more precise statements regarding the case of mean squared error.

General loss functions. Setting

$$g(u) := -\ell'(u), \tag{S80}$$

we write

$$\dot{w} = -\nabla L(w) = \sum_{i=1}^{n-1} g(\langle w, \phi^{(i,i+1)}\rangle)\phi^{(i,i+1)} - g(-\langle w, \phi^{(i+1,i)}\rangle)\phi^{(i+1,i)}, \tag{S81}$$

and so writing

$$g^{(j,k)}(t) := g(y^{(j,k)}\langle w(t), \phi^{(j,k)}\rangle), \tag{S82}$$

we can express

$$\dot{g}^{(j,k)} = y^{(j,k)} g'(y^{(j,k)}\langle w(t), \phi^{(j,k)}\rangle)\langle \dot{w}, \phi^{(j,k)}\rangle. \tag{S83}$$

We now aim to express this differential equation purely in terms of g (rather than ϕ). To do so, we consider the intercept

$$b = \kappa_d \sum_{i=1}^{n-1} \left(g^{(i,i+1)} - g^{(i+1,i)}\right) \tag{S84}$$

Moreover, we define

$$h = g' \circ g^{-1}. \tag{S85}$$

Using Eq. (S81), we express

$$\begin{aligned}\langle \dot{w}, \phi^{(j,k)}\rangle &= \sum_{i=1}^{n-1} g(\langle w, \phi^{(i,i+1)}\rangle) \underbrace{\langle \phi^{(i,i+1)}, \phi^{(j,k)}\rangle}_{K(i,i+1;j,k)} - g(-\langle w, \phi^{(i+1,i)}\rangle) \underbrace{\langle \phi^{(i+1,i)}, \phi^{(j,k)}\rangle}_{K(i+1,i;j,k)} \\ &= \sum_{i=1}^{n-1} g^{(i,i+1)} K(i,i+1;j,k) - g^{(i+1,i)} K(i+1,i;j,k).\end{aligned} \tag{S86}$$

This allows us to express $\dot{g}$ as dependent on g and the kernel K:

$$\dot{g}^{(j,k)} = y^{(j,k)} h(g^{(j,k)}) \sum_{i=1}^{n-1} g^{(i,i+1)} K(i,i+1;j,k) - g^{(i+1,i)} K(i+1,i;j,k). \tag{S87}$$

First, consider $k = j + 1$. By the same logic as in the previous sections, the first sum has one identical data point and all other data points are distinct, whereas the second sum has two overlapping data points:

$$\dot{g}^{(j,j+1)} = h(g^{(j,j+1)}) \left(b + \delta_s g^{(j,j+1)} - \delta_o (g^{(j,j-1)} + g^{(j+2,j+1)}) \right). \tag{S88}$$

As $\frac{\delta_o}{\delta_s} = \frac{1-\alpha}{2}$, this implies

$$\dot{g}^{(j,j+1)} = \delta_s h(g^{(j,j+1)}) \left(g^{(j,j+1)} - \tfrac{1-\alpha}{2} (g^{(j,j-1)} + g^{(j+2,j+1)}) + \frac{1}{\delta_s} b \right), \tag{S89}$$

where we define as boundary conditions $g^{(0,1)} \equiv g^{(1,0)} \equiv 0$ and $g^{(n,n+1)} \equiv g^{(n+1,n)} \equiv 0$. Analogously, for $k = j - 1$,

$$\dot{g}^{(j,j-1)} = \delta_s h(g^{(j,j-1)}) \left(g^{(j,j-1)} - \tfrac{1-\alpha}{2} (g^{(j-2,j-1)} + g^{(j,j+1)}) + \frac{1}{\delta_s} b \right). \tag{S90}$$

At initialization, $w = 0$ and therefore,

$$g^{(i,i+1)}(0) = g^{(i+1,i)}(0) = -\ell'(0) > 0. \tag{S91}$$

The differential equations above imply that the equality $g^{(i,i+1)}(t) = g^{(i+1,i)}(t)$ keeps holding and we define

$$g^{(i)} \equiv g^{(i,i+1)} \equiv g^{(i+1,i)}. \tag{S92}$$

This implies $b = 0$ and we can express

$$\dot{g}^{(i)} = \delta_s h(g^{(i)}) \left(g^{(i)} - \tfrac{1-\alpha}{2} (g^{(i-1)} + g^{(i+1)}) \right), \tag{S93}$$

with the boundary conditions $g^{(0)} \equiv g^{(n)} \equiv 0$. At initialization, $g^{(i)}(0) = -\ell'(0) > 0$, as ℓ is monotonically decreasing. As ℓ is convex, h is strictly negative. If $\alpha = 0$, at initialization, for $i = 2, \dots, n-2$,

$$g^{(i)} - \tfrac{1-\alpha}{2} (g^{(i-1)} + g^{(i+1)}) = 0,$$

As a result, the gradient of all intermediate items initially remains unchanged. The boundary conditions break this symmetry for the terminal items $g^{(1)}$ and $g^{(n-1)}$, which immediately decrease. Once they decrease, the gradient of their neighboring items, $g^{(2)}$ and $g^{(n-2)}$ will also become negative and in this way, the changes in g percolate from outside in. This causes a terminal item effect.

For $\alpha > 0$,

$$g^{(i)} - \tfrac{1-\alpha}{2} (g^{(i-1)} + g^{(i+1)}) > 0,$$

all $g^{(i)}$ decrease from the start. However, the terminal items $g^{(1)}$ and $g^{(n-1)}$ decrease faster, and the faster decreases gradually percolate in as well. As a result, for higher α, the terminal item effect is more attenuated, but nevertheless still obtains.

On the other hand, for $|j - k| > 1$, all sums in Eq. (S86) consist only of overlapping and distinct data points. The distinct data points (consolidated in $b = 0$) cancel out, leaving us with

$$\dot{\hat{y}}^{(j,k)} = \langle \dot{w}, \phi^{(j,k)} \rangle = \delta_o \left(g^{(j,j+1)} + g^{(k-1,k)} - g^{(j,j-1)} - g^{(k+1,k)} \right) = \delta_o \left(g^{(j)} + g^{(k-1)} - g^{(j-1)} - g^{(k)} \right). \tag{S94}$$

As $\delta_o = \delta_s \frac{1-\alpha}{2}$, this implies

$$\dot{\hat{y}}^{(i,j)} = \delta_s \tfrac{1-\alpha}{2} (g^{(i)} + g^{(j-1)} - g^{(i-1)} - g^{(j)})) = \dot{r}^{(i)} - \dot{r}^{(j)}, \tag{S95}$$

where we define the rank representation as

$$r^{(i)}(0) := 0, \quad \dot{r}^{(i)} = \delta_s \tfrac{1-\alpha}{2} (g^{(i)} - g^{(i-1)}). \tag{S96}$$

The terminal item effect implies that this rank representation will be increasing for the $i < \frac{n}{2} + 1$ and decreasing for $i > \frac{n}{2} + 1$. Further, the rank representation will be monotonic in i if $g^{(i-1)} - g^{(i)}$ has higher magnitude for more terminal i. Remarkably, this means that a sufficient condition for the emergence of a rank representation from gradient descent is given by the observation of a convex terminal item effect throughout training.

Mean squared error. For the mean squared error,

$$\ell(u) = \tfrac{1}{2}(1 - u)^2,$$

the dynamics are simplified as

$$h \equiv -1.$$

As a result,

$$\dot{g}^{(i)} = \delta_s (g^{(i)} - \tfrac{1-\alpha}{2} (g^{(i-1)} + g^{(i+1)})). \tag{S97}$$

Defining the vector

$$g \in \mathbb{R}^{n-1}, \quad g_i := g^{(i)}, \tag{S98}$$

these dynamics can be written as

$$\dot{g} = -\tilde{K}g, \quad \tilde{K}_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{1-\alpha}{2} & \text{if } |i-j| = 1, \\ 0 & \text{else.} \end{cases} \tag{S99}$$

Further, as $g(u) = 1 - u$,

$$\hat{y}^{(i,i+1)}(t) = 1 - g^{(i)}(t). \tag{S100}$$

We determine $g(t)$ by using the eigenvalue decomposition of K, as provided by Theorem S4.2. After some algebraic manipulation, this yields the following formulas for g_j and r_j:

Lemma S1.5. *Given the mean squared error as a loss function, g_j and r_j amount to*

$$g_j = \sum_{\substack{k=1 \\ k \text{ odd}}} \frac{2\sin\left(\frac{jk\pi}{n}\right)\exp\left(-\delta_s\left(1-2\alpha\cos\left(\frac{k\pi}{n}\right)\right)t\right)}{n\tan\left(\frac{k\pi}{2n}\right)}. \tag{S101}$$

and

$$r_j(t) = (1-\alpha)\sum_{\substack{k=1 \\ k \text{ odd}}} \frac{\left(\sin\left(\frac{jk\pi}{n}\right) - \sin\left(\frac{(j-1)k\pi}{n}\right)\right)\left(1-\exp\left(-\delta_s\left(1-(1-\alpha)\cos\left(\frac{k\pi}{n}\right)\right)t\right)\right)}{n\tan\left(\frac{k\pi}{2n}\right)\left(1-(1-\alpha)\cos\left(\frac{k\pi}{n}\right)\right)}. \tag{S102}$$

Proof. See Section S4.D. □

We can further simplify this solution for $t \to \infty$:

Lemma S1.6. *Given the mean squared error as a loss function,*

$$\lim_{t\to\infty} \hat{y}^{(j,j+1)}(t) = 1, \tag{S103}$$

$$\lim_{t\to\infty} r_j(t) = \frac{\sinh((\frac{n+1}{2} - j)\lambda)}{\sinh(\frac{n+1}{2}\lambda) - \sinh(\frac{n-1}{2}\lambda)}, \tag{S104}$$

$$\lambda := \operatorname{arccosh}\left(\frac{1}{1-\alpha}\right). \tag{S105}$$

Proof. The solutions found by gradient descent and ridge regression (see below) converge to the same limit as $t, c \to \infty$. The lemma thus follows directly from Lemma S1.4 below. □

Remark S1.7. *Models are often trained on crossentropy for classification problems. While a closed-form solution to the learning dynamics cannot be determined, we can characterize its infinite-time solution. First, we note that crossentropy cannot be minimized by any finite value, but rather attains its minimum as $u \to \infty$. As a consequence, the weights w diverge: $\|w\|_2^2 \to \infty$. However, we can normalize by the minimal margin. This normalized model converges to the support vector machine solution (9). As all $2(n-1)$ training cases are support vectors for both the regression and classification problem, support vector regression and classification are equivalent and the normalized model is characterized by the lemma above.*

S1.E. Exact learning dynamics of neural networks trained with backpropagation in the lazy regime.

S1.E.1. Neural tangent kernel. We consider the deep network

$$f_w^{(0)}(x) := W^{(0)}x, \quad f_w^{(l+1)}(x) := W^{(l+1)}\phi(f_{w^{(:l)}}^{(l)}(x)), \quad W^{(l)} \sim \mathcal{N}(0, \sigma_l^2/H_l), \tag{S106}$$

where

$$w^{(:l)} = \left(W^{(0)}, \ldots, W^{(l)}\right), \tag{S107}$$

is the sequence of weights up to the l-th layer. We denote

$$\theta = \operatorname{vec}\left(w^{(:L)}\right) \tag{S108}$$

as the vectorized total parameters of the network, writing $f_\theta(x) = f_{w^{(:L)}}^{(L)}(x)$. The *neural tangent features* are given by the linearization of the network around its initial parameters θ_0,

$$\tilde{f}(x) = f_{\theta_0}(x) + \langle \nabla_\theta f_{\theta_0}(x), (\theta - \theta_0)\rangle, \tag{S109}$$

and the *neural tangent kernel* is given by the similarity according to those features:

$$\overline{K}(x, x') := \langle \nabla_\theta f_{\theta_0}(x), \nabla_\theta f_{\theta_0}(x')\rangle. \tag{S110}$$

If the distance $\|\theta - \theta_0\|_2^2$ remains infinitesimally small throughout training, the linear approximation $\tilde{f}$ remains valid, and backpropagation can be approximated by gradient descent on the linear readout model given above. By computing the conjunctivity factor associated with $\overline{K}$, we can therefore predict the behavior of neural networks in the regime where the approximation is valid (the "lazy" regime). We make this computation in the next section, but previously remark on our choice of initialization.

Remark S1.8. *We here scale the weight variance by the input dimensionality, H_l, rather than the output dimensionality, H_{l+1}. This initialization is standard in practice as it conserves the variance of the network output throughout the layers with adequately chosen σ_l^2 (10). In investigations of the neural tangent kernel, weights are frequently scaled by the output dimensionality instead. Part of the reason is that, as we will see below, standard scaling leads to less interesting behavior in wide neural networks. We here use the neural tangent kernel to show that our theory captures the behavior of neural networks in the lazy regime. For alternative initialization scales, this would still be the case, though the conjunctivity factor may be different.*

S1.E.2. The NTK's conjunctivity factor. We consider a neural network with one hidden layer. Note that

$$\nabla_{W_{i\cdot}^{(0)}} f_\theta(x) = W_i^{(1)} \phi'(W_{i\cdot}^{(0)} x), \quad \nabla_{W_i^{(1)}} f_\theta(x) = \phi(W_{i\cdot}^{(0)} x). \tag{S111}$$

As a result,

$$\begin{aligned}\mathbb{E}[\overline{K}(x,x')] &= \sum_{i=1}^{H_1} \left(\mathbb{E}\left[\left\langle \nabla_{W_{i\cdot}^{(0)}} f_\theta(x), \nabla_{W_{i\cdot}^{(0)}} f_\theta(x') \right\rangle \right] + \mathbb{E}\left[\left\langle \nabla_{W_i^{(1)}} f_\theta(x), \nabla_{W_i^{(1)}} f_\theta(x) \right\rangle \right] \right) \\ &= \sigma_1^2 \mathbb{E}_{w\sim\mathcal{N}(0,1)} \left[\phi'(w^T x)\phi'(w^T x') \right] + H_1 \frac{\sigma_0^2}{d} \mathbb{E}_{w\sim\mathcal{N}(0,1)} \left[\phi(w^T x)\phi(w^T x') \right].\end{aligned}$$

This means that as H_1 grows large, this similarity diverges. Normalizing the similarity by H_1, in turn, highlights that it is dominated by the similarity of the hidden layer. As a result, under standard initialization, the conjunctivity factor of the neural network with one hidden layer trained with backpropagation is equal to the conjunctivity factor of the hidden layer of the neural network.

In a sense, backpropagation is "trivial" in this regime as it is equivalent to training a linear readout. This is the origin of the different initialization paradigm highlighted in Remark S1.8. If the variance of $W^{(0)}$ is also scaled with H_1, the first term and the second term have comparable magnitude. While we refrain from computing the conjunctivity factor in that case, we note that similar analytical solutions can be derived and the resulting conjunctivity factor obeys the same qualitative trends observed in the previous section, though it is generally slightly higher.

S1.F. Exact learning dynamics of models with fixed representations on transverse patterning. To illustrate the impact of different possible representations on tasks other than transitive inference, we consider the transverse patterning task. In this case, the training dataset is given by

$$\mathcal{D} = \left\{ (x^{(i,i+1)}, 1) | i = 1, \dots, n-1 \right\} \cup \left\{ (x^{(i+1,i)}, -1) | i = 1, \dots, n-1 \right\} \cup \left\{ (x^{(1,n)}, 1), (x^{(1,n)}, -1) \right\}. \tag{S112}$$

Thus, the loss function is given by

$$L(w) = \sum_{(x,y)\in\mathcal{D}} \ell(y\langle w, \phi(x)\rangle) = \left(\sum_{i=1}^{n-1} \ell(\langle w, \phi^{(i,i+1)}\rangle) + \ell(-\langle w, \phi^{(i+1,i)}\rangle) \right) + \ell(\langle w, \phi^{(n,1)}\rangle) + \ell(-\langle w, \phi^{(1,n)}\rangle) \tag{S113}$$

To write this more compactly, we define $\phi^{(n,n+1)} := \phi^{(n,1)}$ and $\phi^{(n+1,n)} := \phi^{(1,n)}$ (i.e. consider modular addition). We consider the mean squared error and set

$$g^{(i)} := 1 - \langle w, \phi^{(i,i+1)}\rangle, \quad \overline{g}^{(i)} := 1 + \langle w, \phi^{(i+1,i)}\rangle. \tag{S114}$$

Gradient flow is then given by

$$\dot{w} = -\nabla L(w) = \sum_{i=1}^{n} g^{(i)} \phi^{(i,i+1)} - \overline{g}^{(i)} \phi^{(i+1,i)}. \tag{S115}$$

Thus,

$$\dot{g}^{(j)} = -\langle \dot{w}, \phi^{(j,j+1)}\rangle = -\sum_{i=1}^{n} g^{(i)} K(i, i+1; j, j+1) + \sum_{i=1}^{n} \overline{g}^{(i)} K(i+1, i; j, j+1). \tag{S116}$$

For the first sum, if $i = j$, the similarity will be κ_s and otherwise κ_d. For the second sum, if $i \in \{j+1, j-1\}$, the similarity is κ_o and otherwise κ_d. Thus, we can define

$$b := \sum_{i=1}^{n} g^{(i)} - \overline{g}^{(i)}, \tag{S117}$$

and express

$$\dot{g}^{(j)} = -b\kappa_d - g^{(j)}(\kappa_s - \kappa_d) + (\overline{g}^{(j+1)} + \overline{g}^{(j-1)})(\kappa_o - \kappa_d) \tag{S118}$$

Analogously,

$$\dot{\overline{g}}^{(j)} = \sum_{i=1}^{n} g^{(i)} K(i, i+1; j+1, j) - \sum_{i=1}^{n} \overline{g}^{(i)} K(i+1, i; j+1, j). \tag{S119}$$

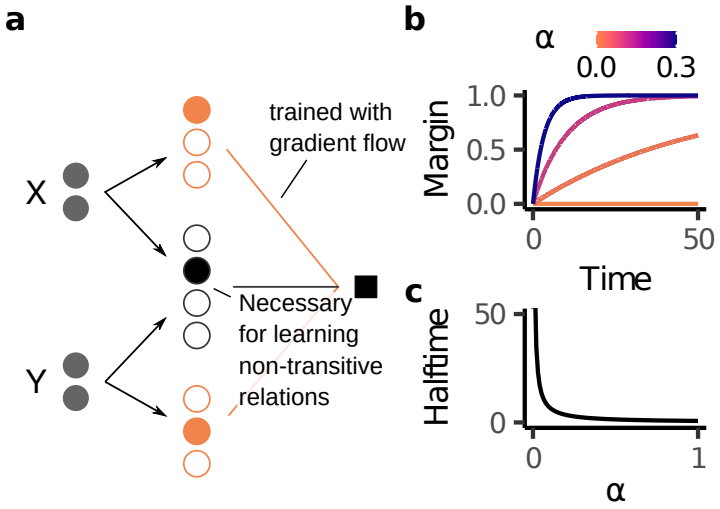


Fig. S1. Behavior of models trained with readout weights trained through gradient flow on transverse patterning. **a**, Learning a non-transitive relation requires a conjunctive component. **b**, Margin over time for different values of α. **c**, Halftime, i.e. time until the model reaches a margin of 0.5, as a function of α.

Here, the first sum has similarity κ_o if $i \in \{j+1, j-1\}$ and κ_d otherwise. The second sum has similarity κ_s if $i = j$ and κ_d otherwise. Thus,

$$\dot{\overline{g}}^{(j)} = b\kappa_d + (g^{(j+1)} + g^{(j-1)})(\kappa_o - \kappa_d) - \overline{g}^{(j)}(\kappa_s - \kappa_d). \quad [S120]$$

Initially $g^{(j)}(0) = \overline{g}^{(j)}(0) = 0$. Because the differential equations are symmetric, this equality keeps holding: $g^{(j)}(t) = \overline{g}^{(j)}(t)$ and $b \equiv 0$. Thus, we can write

$$\dot{g}^{(j)} = -\delta_s g^{(j)} + \delta_o(g^{(j+1)} + g^{(j-1)}). \quad [S121]$$

This equation, in turn is symmetric across all j. Because $g^{(j)}(0) = -\ell'(0)$ is equal for all j, this means that $g^{(j)}(t)$ remain equal. We thus express them as a single scalar $g(t)$:

$$\dot{g} = -(\delta_s - 2\delta_o)g = -\delta_s \alpha g. \quad [S122]$$

The margin is given by $m = 1 - g$. Its speed of change therefore depends on both the overall magnitude of the representation and the conjunctivity factor α. In particular, when $\delta_s = 1$,

$$m(t) = 1 - \exp(-\alpha t). \quad [S123]$$

In particular, setting $m(t) = 0.5$ reveals the margin's halftime:

$$\log(0.5) = -\alpha t, \quad [S124]$$

and therefore

$$t = -\frac{\log(0.5)}{\alpha} = \frac{\log(2)}{\alpha}. \quad [S125]$$

S2. Experimental details

All experiments and figures can be reproduced from the public Github repository: https://github.com/sflippl/relational-generalization-in-ti. This repository is also attached as Software S1.

S2.A. Data analysis and visualization. We analysed the data in R (11) and provide an R package containing all datasets listed below. In addition, we provide infrastructure to load these datasets into Python. The visualizations were created using `ggplot2` (12) and `patchwork` (13).

S2.B. TI behavior in models with non-exchangeable representations. To explore whether our theory can capture the behavior of a non-exchangeable representation, we explored the behavior of models with fixed, randomly sampled ReLU features. For a sufficiently small number of units, this stochasticity results in a drop in accuracy (Fig. S2a), though the margin remains the same (Fig. S2b). Further, because the representations violate exchangeability, the individual models' behavior is not fully captured by the theoretical account. For 1000 units, for example, the models exhibited considerable spread around the theoretical prediction, though they were still well approximated (Fig. S2c,e). Notably, our account captured the average behavior across many models much better, even for a small number of units (Fig. S2d,e).

S2.C. Deep neural network training. We trained deep neural networks with standard He initialization (10), transitioning between the lazy and rich regime by scaling the output of the networks. An output scaling of σ is equivalent to scaling the initialization by $\sigma^{1/(l+1)}$, where l is the depth of the network. (Indeed, the main text describes the transition between the lazy and rich regime in terms of scaling the initialization.) In the lazy regime, the network adheres to a linear readout model with an additional random intercept given by its initial values $f_{\theta_0}(x)$ (see Section S1.E.1). As this complicates analysis, but does not qualitatively change behavior, we subtract this initial intercept. The complete network was therefore given by

$$\tilde{f}_\theta(x) := \sigma(f_\theta(x) - f_{\theta_0}(x)). \quad [S126]$$

We implemented network training in Pytorch (14), using stochastic gradient descent without momentum. The learning rate was chosen automatically by the following process. We started at an extremely high learning rate of 10^{37}. This is because we considered extremely big ranges of scalings, which accordingly tampered down the network output by factors of up to 10^{-32} and had to compensate the learning rate accordingly.) If network training at some point yielded a loss of over 1, we divided the learning rate by 10 and restarted training. This is not an efficient way to determine the optimal learning rate, but suffices for our purposes. Further, we trained the networks until they 1) had a loss under 10^{-5} and 2) did not improve their loss in the last 50 epochs.

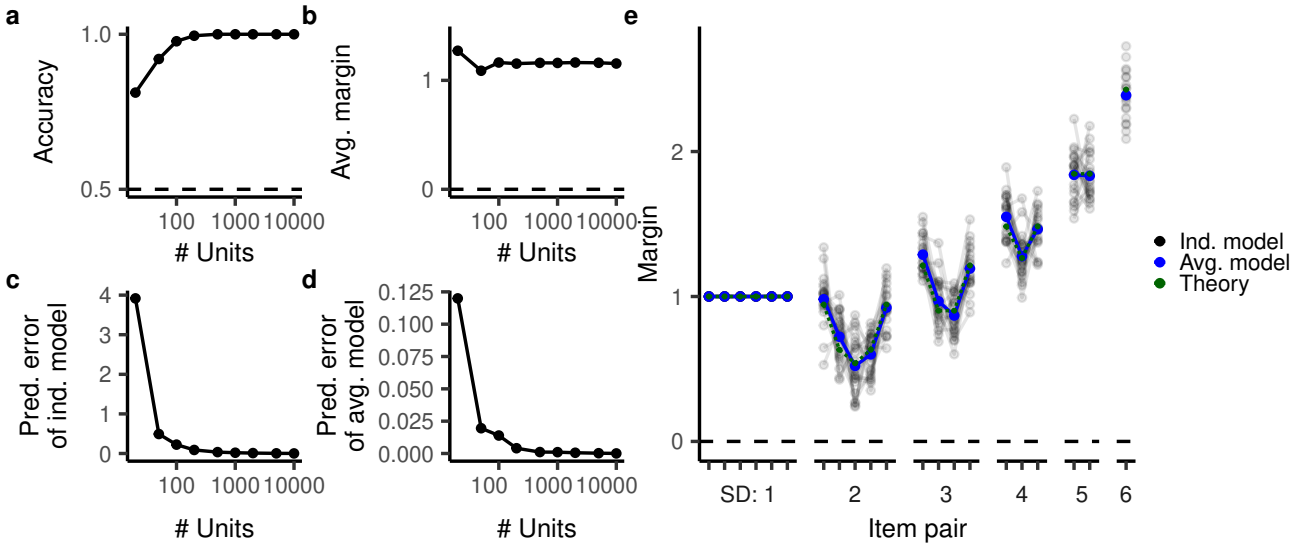


Fig. S2. Behavior of models with fixed, randomly sampled ReLU representations with different numbers of units. **a-b**, Accuracy (a) and average margin (b). **c**, Error of the theoretical prediction compared to the individual model prediction. **d**, Error of the theoretical prediction compared to the average model prediction. **e** Margins for individual ReLU networks with 1000 units (black translucent lines), their average (blue line), and the theoretical prediction (green line).

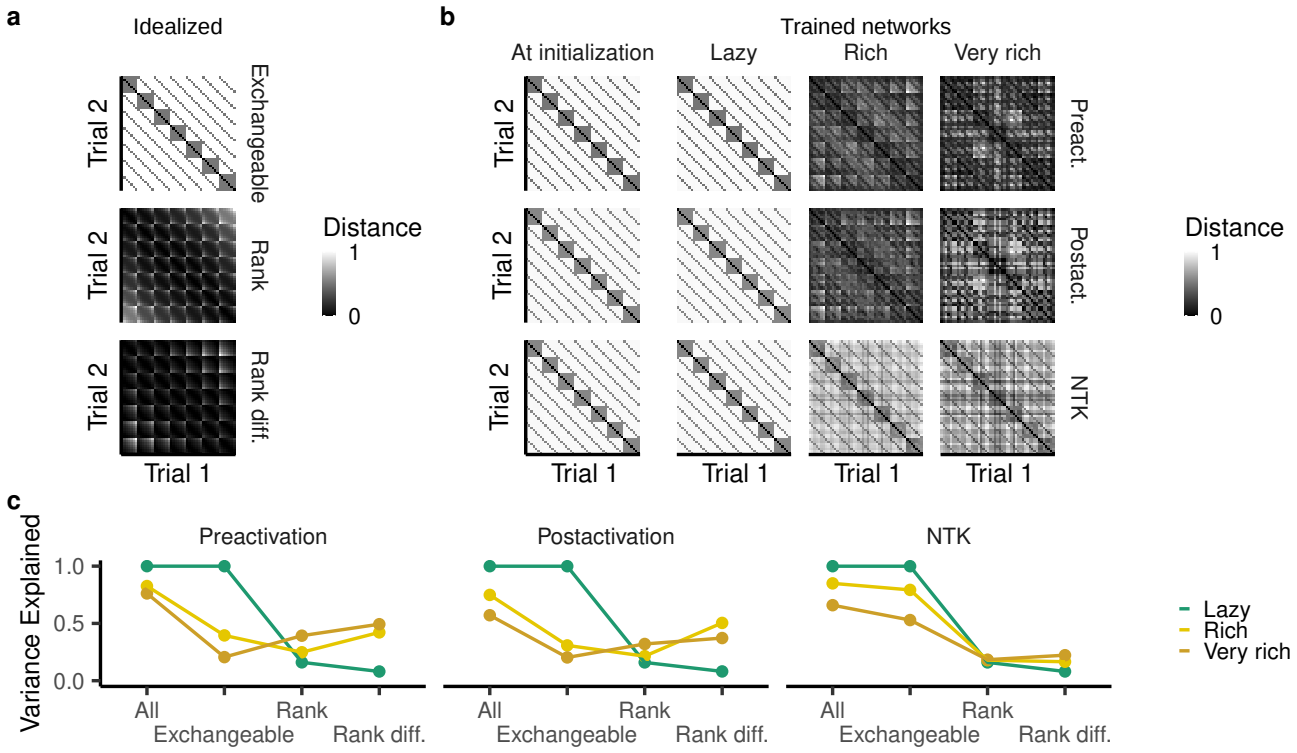


Fig. S3. Representational dissimilarity of the hidden-layer representation, computed as the squared Euclidean distance. **a**, Trial-by-trial representational dissimilarity matrix (RDM) of an idealized hidden layer with either an exchangeable representation, a representation of both items' rank or a representation of the rank difference. The trials are arranged in lexical ordering, i.e. AA, AB, ..., AG, BA, ..., BG, ..., GG. **b**, RDM of the networks' preactivations, postactivations, and neural tangent features at initialization and after lazy or rich training. **c**, Variance explained by a model which predicts the RDM in terms of a exchangeable representation, a representation of rank, a representation of rank difference, or a mixture of all three ("All").

S2.D. Analysis of the hidden-layer representation. Fig. 6 presented the representational dissimilarity matrix (RDM) for a subset of item pairs. Fig. S3 depicts the full trial-by-trial RDM. Candidates for a useful hidden-layer representation are an exchangeable representation (Fig. 6a, top row) or a rank representation; such a rank representation may represent both items' ranks individually (Fig. 6a, middle row) or it may represent the difference between the items' ranks (Fig. 6a, bottom row). We compare the hypothetical representations to those empirically found in the network (Fig. S3b). While Fig. 6 presented those of the postactivation, $\phi(W^{(0)}x)$, we here additionally analyse the preactivation, $W^{(0)}x$ and the neural tangent kernel (NTK; Section S1.E). All of them have an exchangeable representation in the lazy regime. In the rich regime, the representation exhibits some similarity to an exchangeable representation (in particular for the NTK) and perhaps a rank representation. There is also some structure, however, that appear not to be captured by any of the hypothetical RDMs.

For quantitative comparison, we try to predict the RDM by fitting an exchangeable representation, a representation of each items' rank, a representation of rank difference, and a mixture of all three representations. The free parameters of the exchangeable representation are the three distinct similarities. The free parameters of the rank representation are distinct ranks for each item, where we allow for different ranks depending on whether an item was presented first or second. We fit these parameters using gradient descent and measure the proportion of variance explained (Fig. S3c). This confirms our previous interpretation: the lazy regime is perfectly captured by an exchangeable representation, whereas the rich regime is partly captured by an exchangeable representation. In addition, the rich regime is partly captured by a representation of rank difference. However, even a mixture of all three hypothetical representations cannot fully explain the representational structure.

S2.E. Detailed analysis of deep neural network behavior on transitive inference. This section presents a more detailed analysis of the deep network behavior. The results below demonstrate, in particular, that changing the activation function or considering deeper networks does not straightforwardly improve TI behavior. Indeed, these extended results further demonstrate the fundamental strangeness of the deep networks' behavior on TI.

S2.E.1. Rank representability. The "rank representability" measures how well TI behavior can be captured by a rank representation, a value of 0 indicating that it can be perfectly captured. Specifically, a linear model is fit to approximate

$$\hat{f}(i,j) = r_1(i) + r_2(j), \qquad [S127]$$

where r_1 and r_2 are individually estimated for each i and j. The rank representability is then given by the mean squared error of this approximation. As expected, in the lazy regime, the rank representability is zero. As the scaling decreases (and training enters the rich regime), the error increases, indicating that network behavior cannot be described by a rank representation anymore (Fig. S4a). This confirms the picture observed in Fig. 6e.

S2.E.2. Time trajectory. Fig. S4b-e depict the behavior of the neural network with one hidden layer over training. Notably, learning in the rich regime occurs much more suddenly (Fig. S4b) than in the lazy regime. Further, the TI behavior over training exhibits several curious intermediate stages (Fig. S4e). Specifically, the terminal items are learned essentially completely before the intermediate item pairs even improve slightly. In fact, the margin of the item pairs CE, BE, and CF decreases considerable before increasing again.

S2.E.3. Effect of the number of hidden units. At a fixed initialization, neural networks approach the lazy regime as they become wide (15, 16). Neural networks with few hidden units therefore tend to be in the rich regime at standard initialization. Further, the random initialization leads to deviations from the NTK predictions (which are generated from the expected neural tangent kernel) even in the lazy regime; these deviations are larger for fewer hidden units. We therefore decided to explore how the number of hidden units affects the behavior of the network at different initialization scales. We find that at 100 and 1000 hidden units, the network behavior deviates substantially from our theoretical prediction even at standard initialization (Fig. S5a). Part of this is due to the random noise of individual initializations. As a result, the prediction of twenty models averaged together are much more accurate and, in fact, close to zero for 1,000 hidden units (Fig. S5b). At 100 units, in contrast, we do observe a systematic bias (Fig. S5d). Further, at 100 units, model behavior cannot be entirely described by a rank representation, whereas even individual models with 1,000 units do implement a rank representation (Fig. S5c). Taken together, this suggests that at 100 hidden units, the neural network is in the rich regime even at standard initialization, but the effect of the rich regime can be exacerbated by further decreasing the scale of initialization.

S2.E.4. Effect of the nonlinearity. Here we explore the effect of different nonlinearities: both piecewise linear functions and a tanh function as an instance of a saturating nonlinearity. Generally, as the difference in slopes ρ of the piecewise linear function increases (see Section S1.C for a definition of ρ), test margin and test accuracy decrease (Fig. S6a,b) and their rank representation error increases (Fig. S6c). In fact, for $\rho \geq 1.2$, the network performs *below chance-level*. While TI behavior is better at $\rho = 0.5$, it is still non-naturalistic, as an inverse terminal item effect is consistently observed (Fig. S6g). For the tanh nonlinearity, it is not as clear how the rich regime can be approached, as a small output scaling can only be overcome by making the readout large due to the saturating nonlinearity. Here we explore two different scalings: 10^{-6} and 10^{-22}. We find that the tanh nonlinearity has better performance, though it is still characterized by an extremely sharp terminal item effect at a symbolic distance of two (Fig. S6g).

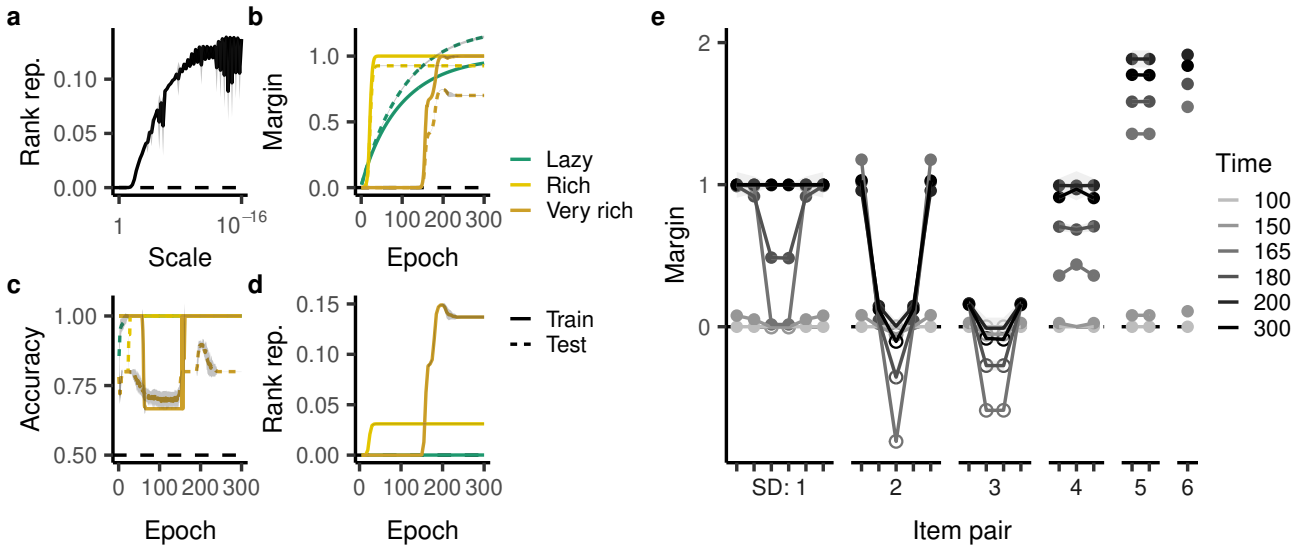


Fig. S4. TI behavior of deep networks as a function of initialization scale and training duration. **a**, Rank representability at the end of training as a function of initialization scale. **b**-**d**, Margin, accuracy, and rank representability over the first 300 epochs. Margins (i.e. $\hat{y}y$, where $\hat{y}$ is the model prediction and $y \in \{-1, 1\}$ is the trial label) are averaged over all trials within the training and test set. **e**, Margin at the different item pairs as a function of training duration (very rich regime).

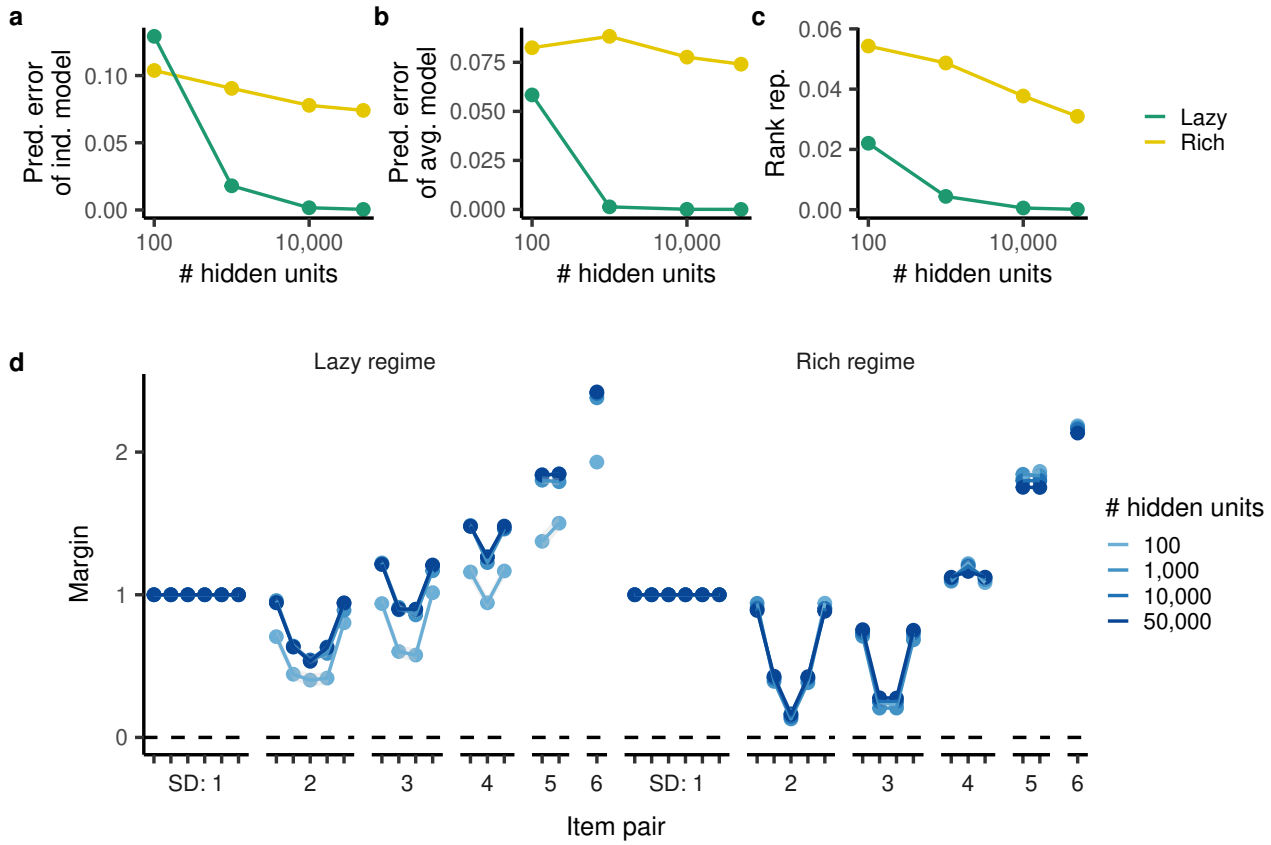


Fig. S5. TI behavior for different numbers of hidden units and standard initialization ("lazy") as well as smaller initialization (scale 10^{-3}, "rich"). **a**, Error of the theoretical prediction compared to the individual model prediction. **b**, Error of the theoretical prediction compared to the average model prediction. **c**, Rank representability as a function of the number of hidden units. **d**, Predictions of the networks in different regimes and for different numbers of hidden units.

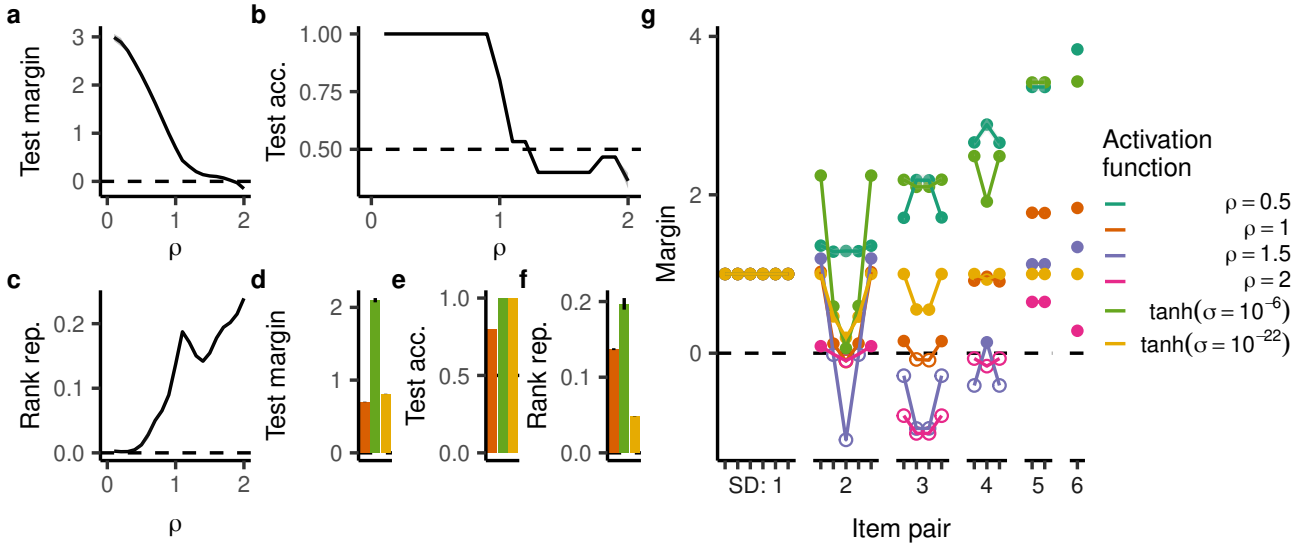


Fig. S6. TI behavior for different activation functions. **a**-**c**, Margin (a), accuracy (b), and rank representability (c) as a function of ρ (see (S11)). **d**-**f**, Margin (d), accuracy (e) and rank representability (f) for the tanh activation function at two different output scalings σ (as well as, for comparison, the ReLU activation function ($\rho = 1$)). **g**, Margin of the different item pairs for selected activation functions.

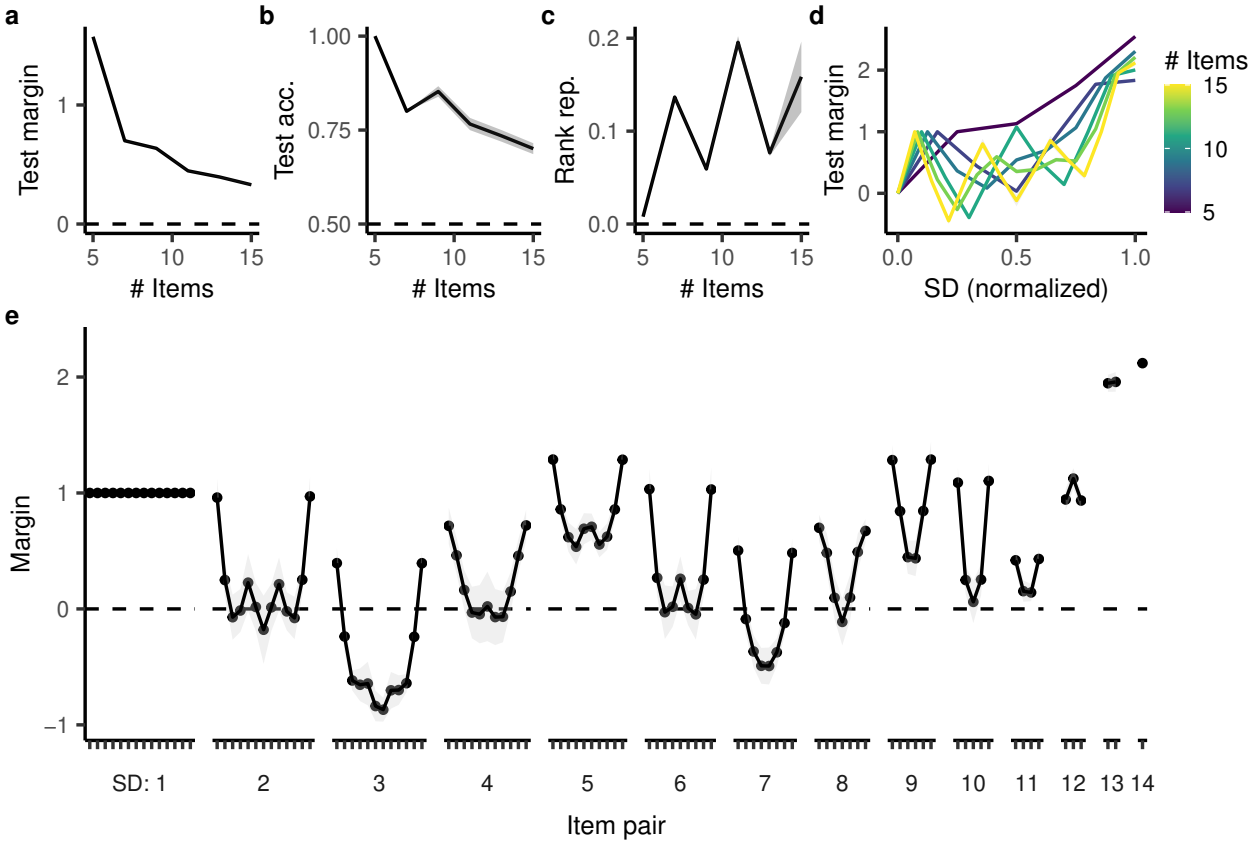


Fig. S7. TI behavior as a function of the number of items. **a**-**c**, Margin, accuracy, and rank representability as a function of the number of items. **d**, The average test margin for a given symbolic distance (normalized by $n-1$). **e**, Margin at the different item pairs for 15-item TI.

S2.E.5. Effect of the number of items. We next determined the behavior of ReLU networks with one hidden layer in the very rich regime for different numbers of items. Generally, both test margin and accuracy decreased as the number of items increased (Fig. S7a,b). The rank representability exhibited a more non-monotonic pattern. In particular, none of the networks except for those trained on 5-item TI exhibited a symbolic distance effect (Fig. S7d). Instead their margin oscillated at intermediate symbolic distances before eventually increasing at large symbolic distances.

S2.E.6. Networks trained on crossentropy. Neural networks are often trained on crossentropy rather than mean squared error. We therefore study the inductive bias of neural networks trained on TI with crossentropy. Notably, the learning trajectory of neural networks trained on crossentropy can generally not be approximated by their neural tangent kernel as the weights must diverge to minimize the loss function. This is because crossentropy is minimized as the margins diverge towards infinity (see Remark S1.7). Nevertheless, we consider networks with standard (scaling 1) or small (scaling 10^{-3}) initialization and find that they behave differently, presumably because we train for a finite number of epochs. Further, we consider both learning by backpropagation and only adjusting the linear readout.

While the margins grow over the duration of learning (Fig. S8a), we find that the normalized margins of the linear readout model are well approximated by our theory. This confirms our theoretical results (Remark S1.7). More surprisingly, the neural network trained by backpropagation with standard initialization is also well approximated by our theory (Fig. S8b). Only those networks trained with backpropagation and initialized with small weights exhibit behavior similar to that of neural network trained on MSE in the rich regime (Fig. S8a-c). There are two potential reasons for this: it is possible that after longer amounts of training, the neural network with standard initialization may exhibit similar behavior to that with smaller initialization (17). It is also possible that different initializations genuinely affect the inductive bias of neural networks trained with crossentropy. This is an interesting possibility especially in light of the fact that known theoretical results are largely silent on such an effect (17, 18).

S2.E.7. Deeper networks. While neural networks with one hidden layer already exhibit representation learning, the superior generalization of deep neural networks is often tied to their depth. We therefore explore how increasing depth changes the networks' inductive bias on TI. We use an initialization scale of 10^{-3} and train networks with one to three hidden layers. (Deeper networks do not converge to a minimal loss within six hours and we leave their exploration to future work.) We find little difference between the behavior of networks with different depth (Fig. S8d), suggesting that simply increasing the depth of the networks is not sufficient to improve their TI performance.

S2.F. Symmetric weights in the input layer. In related work, Nelli *et al.* (19) also compare the TI behavior of deep neural networks in the lazy and rich regime, finding that neural networks trained in the rich regime have an emergent rank representation in their hidden layer. Notably, their networks implement a weight symmetry between the two input items. Specifically, given two items $x_1, x_2 \in \mathbb{R}^d$, their network computes the output as

$$f(x_1, x_2) = w_2^T \phi(W_1 x_1 - W_1 x_2), \quad \phi(u) = \max(u, 0), \quad w_2 \in \mathbb{R}^h, \quad W_1 \in \mathbb{R}^{h \times d}. \tag{S128}$$

At first glance, this network is not additive – after all, ϕ mixes the two items nonlinearly. However, only the directional bias can be nonlinear. The average choice of the network between either order of presentation,

$$\overline{f}(x_1, x_2) := f(x_1, x_2) - f(x_2, x_1) = w_2^T (W_1 x_1 - W_1 x_2), \tag{S129}$$

is additive, as $\phi(u) - \phi(-u) = u$.

Indeed, weight symmetry strongly improves test performance in the rich and lazy regime (Fig. S9a). In line with the parameters used in (19), the rich regime corresponds to an initialization scaling of 0.025 in the hidden layer and a scaling of 1 in the readout layer, whereas the lazy regime corresponds to a scaling of 1 in both layers. Note that training performance is suboptimal because the network is only trained for 40 epochs. The RDM of the postactivation (Fig. S9b) corresponds to a rank difference representation (Fig. S9c), whereas such a representation leaves a substantial proportion of variance unexplained in the lazy regime as well as without weight symmetry. Notably, neither the preactivation nor the NTK can be explained as a rank difference representation.

In addition to weight symmetry, Nelli *et al.* implement training with some differences to our implementation: they run stochastic gradient descent with a batch size of one, use networks with only twenty units, run training for 40 epochs rather than until convergence, and initialize the hidden-layer weights with a smaller variance than the output weights (specifically multiplying them by a factor of 0.025). To investigate whether these differences are responsible for the emergence of a rank difference representation, we changed each of these hyperparameters independently. We found that weight symmetry is necessary and sufficient for the emergence of a rank difference representation. Note that in the wide network, the rank difference representation is weaker because the network is leaving the rich regime due to its increasing width. To re-enter the rich regime, we would have to initialize the weights by a smaller factor.

S3. Analysis of the rich-regime behavior

For hidden weights $W \in \mathbb{R}^{H \times d}, W = (w_h)_{h=1}^H$ and readout weights $v \in \mathbb{R}^H$, the ReLU network is expressed by

$$f(x; W, v) = \sum_{h=1}^{H} v_h \phi(w_h^T x), \quad \phi(u) = \max(u, 0). \tag{S130}$$

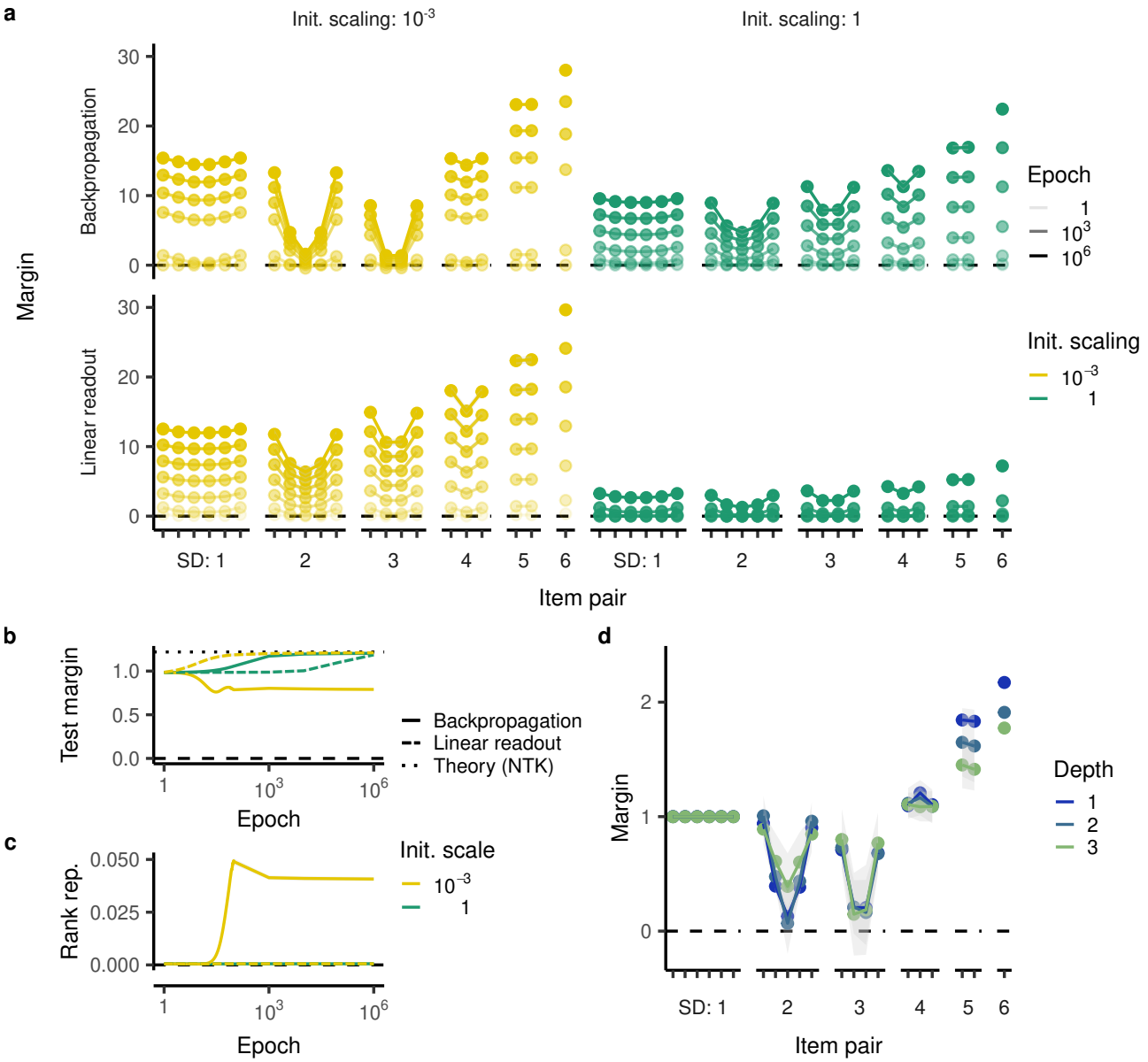


Fig. S8. **a**, TI behavior of networks trained on crossentropy. The top row depicts a network that has been trained by backpropagation; the bottom row a network whose linear readout has been trained. Further, the left column is initialized with smaller weights than the right column. **b**, Test margin of the networks over training. **c**, Rank representability of the networks over training. **d**, TI behavior of networks with different depths at an initialization scale of 10^{-3}.

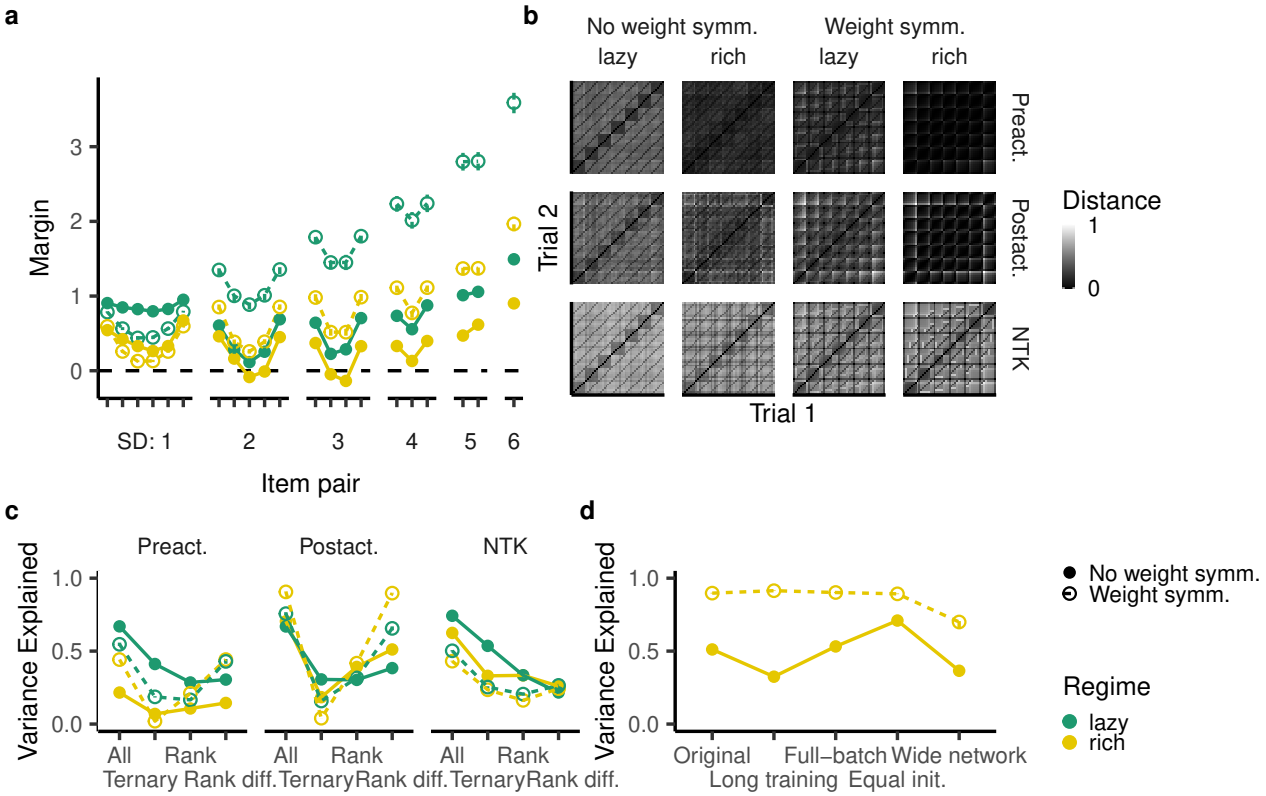


Fig. S9. Weight symmetry restores generalization in the rich regime (19). Specifically, weight symmetry is implemented as $W^{(1)} = -W^{(2)}$, where $W^{(1)}$ are the hidden weights corresponding to the first item and $W^{(2)}$ are the hidden weights corresponding to the second item. **a**, Generalization behavior in the lazy and rich regime with and without weight symmetry. **b**, Trial-by-trial RDM of the preactivation, postactivation and NTK in the lazy and rich regime with and without weight symmetry. **c**, Variance of those RDMs explained by a exchangeable representation, a rank representation, a representation of rank difference, and a mixture of all representations. **d**, Variance explained of the RDMs that introduce one of the changes introduced by our training protocol. Long training indicates training until convergence and equal init. indicates that both the hidden layer and the readout layer were initialized with a scaling of 0.025. Finally, the wide network indicates a network with 1000 units.

Because the ReLU nonlinearity ϕ is positively homogeneous ($\phi(su) = s\phi(u)$ for all $s > 0$), we can rewrite this network using the hidden weight directions $\hat{w}_h = w_h/\|w_h\|_2 \in \mathbb{R}^d$, the magnitude $m_h = \|w_h\|_2|v_h| \in \mathbb{R}$, and the sign $\sigma_h = \text{sign}(v_h) \in \{-1, 1\}$:

$$f(x; \hat{W}, m, \sigma) = \sum_{h=1}^{H} \sigma_h m_h \phi(\hat{w}_h^T x). \quad \text{[S131]}$$

Prior work has found that minimizing the ℓ_2-norm of all weights in the network is equivalent to minimizing $2\|m\|_1$ (where the weight directions can be freely chosen) (17, 20, 21). Because the ℓ_1-norm induces sparsity, this norm incentivizes a sparse set of hidden units, in addition to keeping the magnitude m_h of those hidden units low.

Below we describe the periodic coding schema for TI with arbitrary numbers of items n and compute the network's weight norm. We then compute the weight norm of a rank-network, which instead implements a ranking system, finding that it is much larger. We then compare these hand-constructed networks to neural network simulations, finding that they capture important properties of empirically trained networks and describing specific differences.

S3.A. Cooperative coding schema. The network we describe here uses four units (H_{1+}, H_{2+}, H_{1-}, and H_{2-}) to fit the training trials on TI. H_{1+} and H_{2+} have positive readout weights and H_{1-} and H_{2-} have negative readout weights:

$$f(x) = \phi(H_{1+}^T x) + \phi(H_{2+}^T x) - \phi(H_{1-}^T x) - \phi(H_{2-}^T x). \quad \text{[S132]}$$

These units partition the dataset $\mathcal{D}$ (see (S36)) into four disjoint subsets:

$$\mathcal{D}_{1+} := \left\{x^{(i,i+1)} | i \in \{4k+1, 4k+2\} \text{ for some } k\right\}, \quad \mathcal{D}_{2+} := \left\{x^{(i,i+1)} | i \in \{4k+3, 4k+4\} \text{ for some } k\right\}. \quad \text{[S133]}$$

$\mathcal{D}_{1-}$ and $\mathcal{D}_{2-}$ are defined analogously for $x^{(i+1,i)}$. For example, for $n = 7$,

$$\mathcal{D}_{1+} = \left\{x^{(1,2)}, x^{(2,3)}, x^{(5,6)}, x^{(6,7)}\right\}, \quad \mathcal{D}_{2+} = \left\{x^{(3,4)}, x^{(4,5)}\right\}. \quad \text{[S134]}$$

Each unit H_Δ, $\Delta \in \{1+, 2+, 1-, 2-\}$ then outputs one on its own data points and zero on all others:

$$\phi(H_\Delta^T x) = \begin{cases} 1 & \text{if } x \in \mathcal{D}_\Delta, \\ 0 & \text{else.} \end{cases} \quad \text{[S135]}$$

Below we describe how its weights allow it to do so. $H_\Delta \in \mathbb{R}^{2n}$ has one weight for each item and position. We write $H_\Delta(i, s)$ for item $i \in \{1, \ldots, n\}$ in position $s \in \{1, 2\}$. Now consider a data point $x^{(i,i+1)} \in \mathcal{D}_\Delta$ (i.e. $\Delta \in \{1+, 2+\}$). The unit then distributes its weights equally across those two items:

$$H_\Delta(i, 1) = H_\Delta(i + 1, 2) = 0.5. \quad \text{[S136]}$$

However, this also means that it would response positively to the data points $x^{(i,i-1)}, x^{(i+2,i+1)} \notin \mathcal{D}_\Delta$. To avoid this, the unit requires a negative weight in response to the other item for those trials:

$$H_\Delta(i - 1, 2) = H_\Delta(i + 2, 1) = -0.5. \quad \text{[S137]}$$

This same pattern can be repeated for two consecutive trials, e.g. $x^{(i,i+1)}, x^{(i+1,i+2)}$. However, encoding $x^{(i+2,i+3)}$ into the same unit would conflict with (S137). For this reason, the specialization of units switches every two trials.

Note that for $\Delta \in \{1-, 2-\}$, the exact roles of the first and second entry are reversed, i.e.

$$H_{k+}(i, s) = H_{k-}(i, 3 - s), k \in \{1, 2\}. \quad \text{[S138]}$$

This logic allows us to define a formula that constructs this network for arbitrary numbers of trials. We show the example coding scheme for $n = 7$ in Fig. S10a,b. The above intuition also helps us get a rough estimate of the magnitude of these weights. The total number of training trials is $2(n - 1)$. Each unit H_Δ is responsible for roughly a quarter of those. For each trial, these units require four weights at a magnitude of 0.5. This means that each unit's ℓ_2-norm (i.e. its magnitude) is roughly:

$$\|H_\Delta\|_2^2 \approx 4 \cdot \tfrac{1}{2}(n - 1)0.5^2 = \tfrac{1}{2}(n - 1). \quad \text{[S139]}$$

Thus, the networks overall weight norm is approximately

$$\|w\|_2^2 = 2 \sum_\Delta \|H_\Delta\|_2 \approx 4\sqrt{2}\sqrt{n - 1}. \quad \text{[S140]}$$

To determine the exact weight norm, we need to treat different numbers of trials slightly differently. This gives rise to a slightly more complicated formula that is still on the order of $\sqrt{n}$ (Fig. S10d, details in Section S4.F).

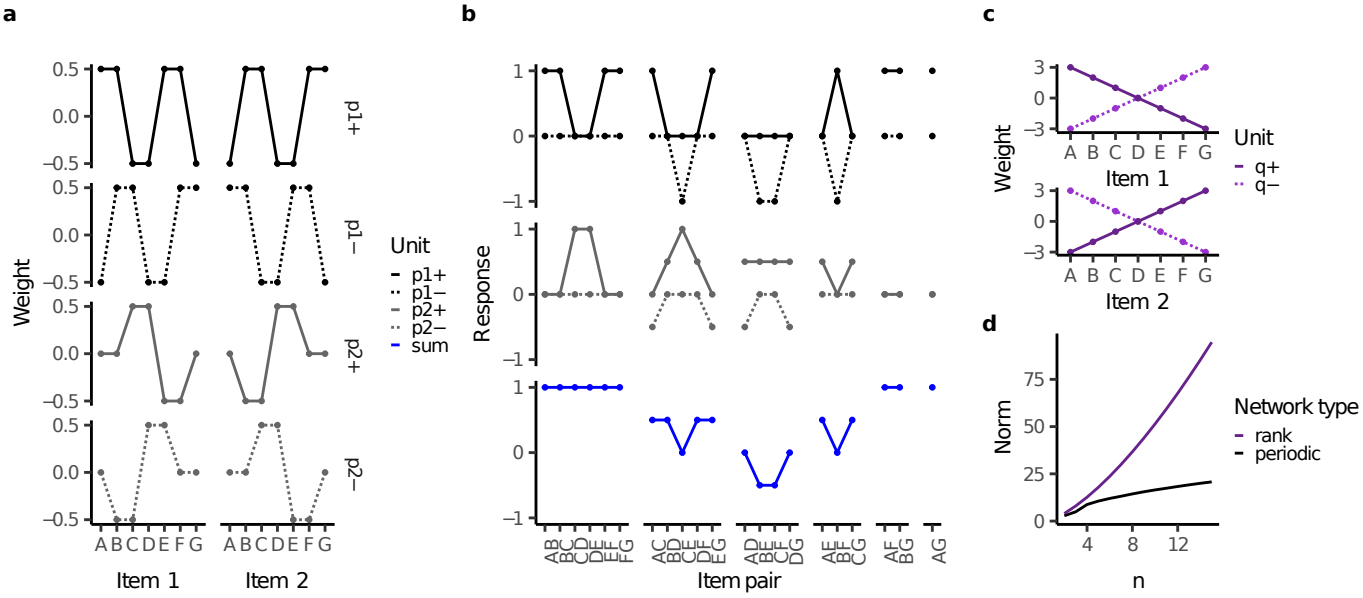


Fig. S10. Hand-constructed networks. **a**, Weights of the network implementing the cooperative coding schema for $n = 7$. **b**, Responses to all trials with positive outputs. First row depicts the units H_{1+}, H_{1-}, second row depicts H_{2+} and H_{2-} and the third row depicts their sum. **c**, Weights of the rank network for $n = 7$. **d**, Norm of the two networks as a function of the number of items n.

S3.B. Rank network. We contrast this cooperative coding schema with a simple ranking system in the hidden layer (Fig. S10c). Specifically, such a network would consist of two units q_+, q_- and compute its response as

$$f(x) = \phi(q_+^T x) - \phi(q_-^T x). \quad [S141]$$

Furthermore, $q_+ = -q_-$ and therefore

$$f(x) = q_+^T x. \quad [S142]$$

q_+ then simply encodes the rank of each item, centered around zero:

$$q_+(i,1) = \frac{n+1}{2} - i, \quad q_+(i,2) = i - \frac{n+1}{2}. \quad [S143]$$

We now compute the magnitude of these units:

$$\|q_+\|_2^2 = 2\sum_{i=1}^{n}(i - \frac{n+1}{2})^2 = 2\sum_{i=1}^{n} i^2 - 4\frac{n+1}{2}\sum_{i=1}^{n} i + 2n\frac{(n+1)^2}{4}, \quad [S144]$$

where we used Bernoulli's formula. Note that (see e.g. 22)

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}, \quad \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}. \quad [S145]$$

Thus,

$$\|q_+\|_2^2 = \frac{n(n+1)(2n+1)}{3} - n(n+1)^2 + \frac{n(n+1)^2}{2} = \frac{n(n+1)(2n+1)}{3} - \frac{n(n+1)^2}{2}. \quad [S146]$$

Simplifying this implies

$$\|q_+\|_2^2 = \frac{2}{3}n^3 + n^2 + \frac{n}{3} - \frac{n^3}{2} - n^2 - \frac{n}{2} = \frac{1}{6}n(n^2-1). \quad [S147]$$

Thus, the network has a total weight norm of

$$\|w\|_2^2 = 2(\|q_+\|_2 + \|q_-\|_2) = 4\|q_+\|_2 = \frac{4}{\sqrt{6}}\sqrt{n(n^2-1)} = \frac{2\sqrt{6}}{3}\sqrt{n(n^2-1)}. \quad [S148]$$

This weight norm is considerably bigger than that of the network implementing a cooperative coding schema. In particular, the norm is $O(n^{1.5})$, while the latter network has a norm of $O(\sqrt{n})$ (Fig. S10d).

S3.C. Comparison with empirical networks. Here we provide a detailed comparison between the hand-constructed networks and the empirically computed deep networks.

S3.C.1. Norm comparison. First, we compute the empirical networks' norm. The hand-constructed networks have a lower norm than their empirical counterpart, but the values are generally close together (Fig. S11a). Further, the norms follow the same trends with respect to n. In particular, our theory would predict that $\|w\|_2^2/n$ should increase for the symmetric network (which should be of order $O(n^{1.5})$) but decrease for the dense network (of order $O(n^{0.5})$). Indeed, this is the case (Fig. S11b). Note that the symmetric network's norm appears to increase a bit faster than that of the rank network.

The suboptimalities in the empirical networks may have arisen because they found a local stationary point rather than a global minimum, because the initialization scale of 10^{-16} was still not small enough, or because training on mean squared error may not exactly minimize the ℓ_2-norm (23) (though we note that networks trained on crossentropy, which provably converge to a stationary point, implement the same mechanism).

S3.C.2. Clustering analysis. We assessed the network's sparsity in hidden units by performing k-means clustering over the normalized weights $(\hat{w}_h)_{h=1}^H$, weighting each sample by its respective magnitude m_h. We perform this clustering analysis separately for units with positive and negative readout weights, for the sake of easier interpretability. After identifying k clusters, we replace each hidden weight by its respective cluster centroid, giving rise to a network with k hidden units.

We analyzed networks trained on TI with $n = 5, \ldots, 15$. We consistently found that these networks were well explained by six clusters in total (Fig. S11c). To compare the units extracted from networks with different initializations, we identified the unit-to-unit mapping that minimized the deviation between the different networks' units. Fig. S11d,e depict the three units with positive and negative signs. Notably, their weights are symmetric in that each positive unit responds to item 1 in the same way that the corresponding negative unit responds to item 2 and vice versa. Finally, Fig. S11f depicts the six units extracted from the networks trained on 15 items. Again, these units are remarkably consistent.

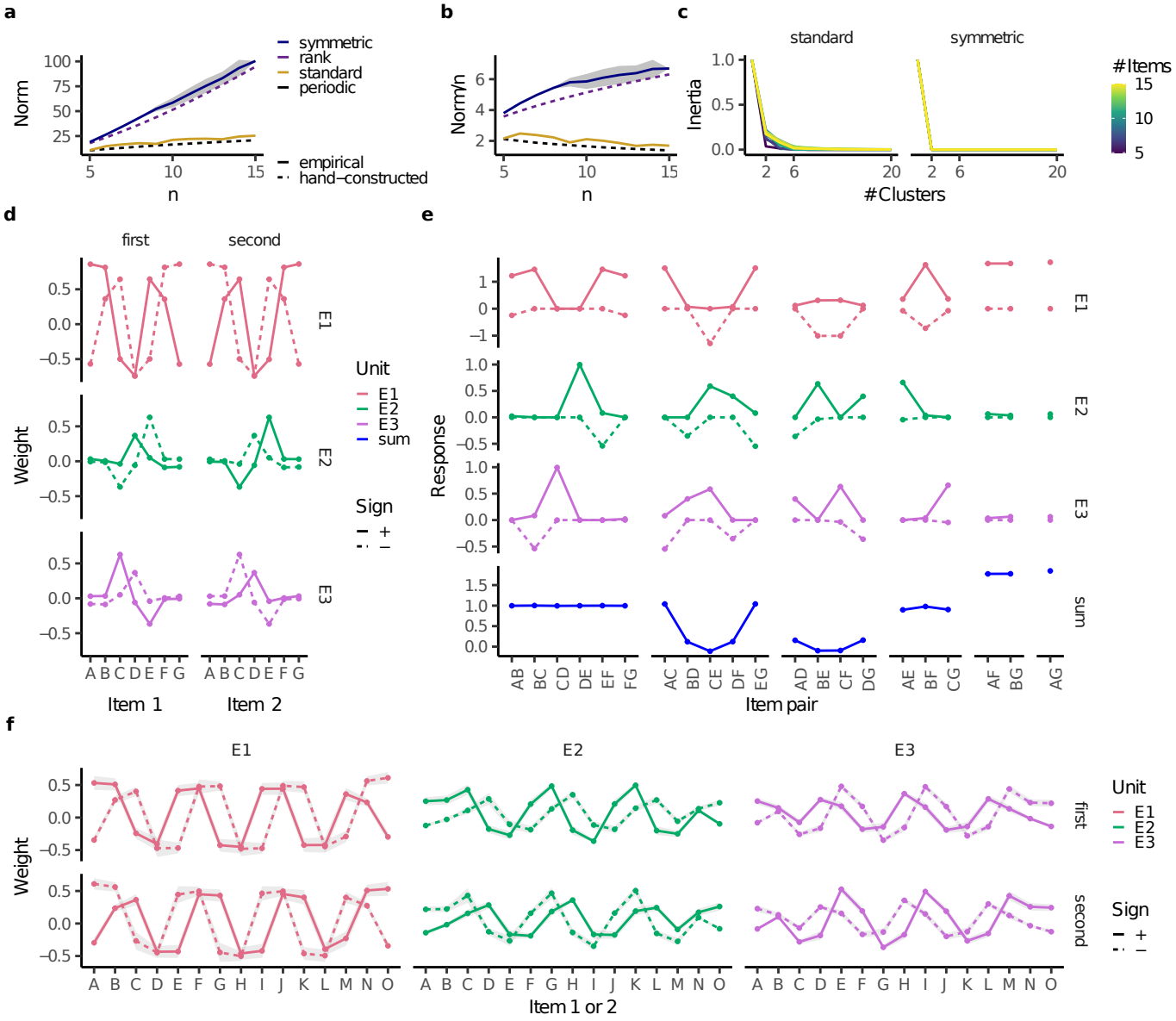


Fig. S11. Extended analysis of empirical rich-regime networks (initialization scale 10^{-16}). All figures depicted are averaged across twenty random initializations; shaded areas show $\pm$ one standard deviation. **a**,**b**, (a) Norm and (b) Norm divided by the number of items of standard and symmetric networks trained on different numbers of items (solid lines), compared to their corresponding hand-constructed network. **c** Inertia (i.e. proportion of unexplained variance) as a function the number of clusters, the network structure, and the number of items the networks were trained on. **d**, Weights of the three positive and negative centroid units. **e**, Response of the different units to all item pairs. (Negative item pairs look analogously, though with the role of positive and negative units switched. This is apparent from the symmetry in panel d.) **f**, Weights of the three positive and negative units for networks trained on 15 items. They indeed have a periodic structure approximately described by the periodic network and E2 and E3 continue to share the role of H2.

S3.C.3. Differences between empirical and hand-constructed networks. We observe a few differences between the empirical and hand-constructed networks, as highlighted in Fig. 6h in the main text. We hand-construct two networks that interpolate between the solution described in the main text (HCN) and the empirical networks (for $n = 7$) (Table S1). In modification 1, the network does not silence the second unit on the trials BC/CB and EF/FE. Instead, the weights of the first unit are increased to compensate for this change. This, in turn, causes other weights to change as well to prevent this change from having ramifications. As a result, this network has a lower weight norm on unit 2 ($\frac{1}{2}\sqrt{6}$) but a strongly increased weight norm on unit 1 ($\sqrt{8(\frac{3}{4})^2 + 6(\frac{1}{2})^2}$). Keeping in mind that the network has corresponding negative units (and that, to compute its norm, the total ℓ_2-norm of its units needs to be multiplied by 2), its norm is

$$4\left(\sqrt{8(\tfrac{3}{4})^2 + 6(\tfrac{1}{2})^2} + \tfrac{1}{2}\sqrt{6}\right) = 2\left(\sqrt{2\cdot 9 + 6} + \sqrt{6}\right) = 2\left(\sqrt{24} + \sqrt{6}\right) = 6\sqrt{6} \approx 14.70, \tag{S149}$$

which is higher than the periodic network's norm of $4 \cdot 0.5\left(\sqrt{14} + \sqrt{8}\right) = 2(\sqrt{14} + \sqrt{8}) \approx 13.14$.

The second modification then turns unit 2 into to units with non-overlapping units. Even though the total number of weights doesn't increase, the decrease in sparsity yields a higher weight norm. Specifically, each of those units has a norm of $\frac{1}{2}\sqrt{3}$ and their sum, $\sqrt{3}$ is higher than the previous unit's norm $\frac{1}{2}\sqrt{6}$. In total, this network has a norm of

$$4\left(\sqrt{8(\tfrac{3}{4})^2 + 6(\tfrac{1}{2})^2} + \sqrt{3}\right) = 2\left(\sqrt{24} + 2\sqrt{3}\right) = (4 + 2\sqrt{2})\sqrt{6} \approx 16.73. \tag{S150}$$

This is very close to the networks average weight norm, $\|w\|_2^2 \approx 16.64 \pm 0.036$.

S4. Detailed mathematical analysis

S4.A. Prerequisites.

S4.A.1. Results on Tridiagonal Matrices.

Theorem S4.1 (Hu & O'Connell (8))**.** *Let $a, b \in \mathbb{R}$ such that $\frac{a}{b} \leq -2$ and*

$$B \in \mathbb{R}^{m\times m}, \quad B_{ij} = \begin{cases} a & \text{if } i = j, \\ b & \text{if } |i-j| = 1, \\ 0 & \text{else.} \end{cases} \tag{S151}$$

Then

$$B^{-1}_{ij} = -\frac{\cosh\left((m+1-|j-i|)\lambda\right) - \cosh\left((m+1-i-j)\lambda\right)}{2b\sinh(\lambda)\sinh\left((m+1)\lambda\right)}, \tag{S152}$$

where

$$\lambda := \operatorname{arccosh}\left(-\frac{a}{2b}\right). \tag{S153}$$

Proof. We define $\tilde{B} := \frac{1}{b}B$. Using (8),

$$\tilde{B}^{-1}_{ij} = -\frac{\cosh\left((m+1-|j-i|)\lambda\right) - \cosh\left((m+1-i-j)\lambda\right)}{2\sinh(\lambda)\sinh\left((m+1)\lambda\right)}, \tag{S154}$$

and thus

$$B^{-1} = \tfrac{1}{b}\tilde{B}^{-1}. \tag{S155}$$

□

Theorem S4.2 (Böttcher & Grudsky (24))**.** *A tridiagonal matrix*

$$B \in \mathbb{R}^{m\times m}, \quad B_{ij} = \begin{cases} a & \text{if } i = j, \\ b & \text{if } |i-j| = 1, \\ 0 & \text{else,} \end{cases} \tag{S156}$$

has the eigenvalue decomposition

$$\begin{aligned} A &= U\Lambda U, \\ \Lambda_{jj} &= a + 2|b|\cos\left(\frac{\pi j}{m+1}\right), \\ U_{jk} &= \sqrt{\frac{2}{m+1}}\sin\left(\frac{jk\pi}{m+1}\right). \end{aligned} \tag{S157}$$

	Hand-constructed network				Modification 1				Modification 2				
	p1+		p2+		Unit 1		Unit 2		Unit 1		Unit 2		Unit 3
1	0.5	-0.5			**0.75**	-0.5			0.75	-0.5			
2	0.5	0.5		-0.5	**0.75**	0.5		**0**	0.75	0.5			
3	-0.5	0.5	0.5	-0.5	-0.5	**0.75**	0.5	-0.5	-0.5	0.75	0.5		-0.5
4	-0.5	-0.5	0.5	0.5	**-0.75**	**-0.75**	0.5	0.5	-0.75	-0.75		0.5	0.5
5	0.5	-0.5	-0.5	0.5	**0.75**	-0.5	-0.5	0.5	0.75	-0.5	-0.5		0.5
6	0.5	0.5	-0.5		0.5	**0.75**	**0**		0.5	0.75			
7	-0.5	0.5			-0.5	**0.75**			-0.5	0.75			

Table S1. Weights associated with the hand-constructed network described in the main text on $n = 7$ as well as two modifications that account for the differences empirically observed. Bolded weights emphasize changes from the periodic network to modification 1.

Proof. This is a special case of (24, Theorem 2.4). All that remains is proving the scaling factor. We use

$$\sin^2(x) = \frac{1-\cos(2x)}{2}, \tag{S158}$$

to infer

$$\begin{aligned}\sum_{k=1}^{m} \sin^2\left(\frac{jk\pi}{m+1}\right) &= \frac{m}{2} - \frac{1}{2}\sum_{k=1}^{m} \cos\left(2\frac{jk\pi}{m+1}\right) \\ &= \frac{1}{2}\left(m - \frac{\sin\left((2m+1)\frac{j\pi}{m+1}\right) - \sin\left(\frac{j\pi}{m+1}\right)}{2\sin\left(\frac{j\pi}{m+1}\right)}\right) \\ &= \frac{1}{2}\left(m + \frac{2\sin\left(\frac{j\pi}{m+1}\right)}{2\sin\left(\frac{j\pi}{m+1}\right)}\right) = \frac{m+1}{2},\end{aligned} \tag{S159}$$

where the last equality holds because

$$\sin\left((2m+1)\tfrac{j\pi}{m+1}\right) = \sin\left((2m+1-2(m+1))\tfrac{j\pi}{m+1}\right) = \sin\left(-\tfrac{j\pi}{m+1}\right). \tag{S160}$$

Note that we know that the eigenvectors are orthogonal because the matrix is symmetric and thus normal. □

S4.A.2. Hyperbolic identities.

Lemma S4.3. *For any x, and $m, n \in \mathbb{N}$,*

$$\sum_{k=m}^{n} \cosh(kx+b) = \frac{\sinh\left(\frac{2n+1}{2}x+b\right) - \sinh\left(\frac{2m-1}{2}x+b\right)}{2\sinh(\frac{1}{2}x)}, \tag{S161}$$

$$\sum_{k=m}^{n} \sinh(kx) = \frac{\cosh\left(\frac{2n+1}{2}x\right) - \cosh\left(\frac{2m-1}{2}x\right)}{2\cosh\left(\frac{1}{2}x\right)}, \tag{S162}$$

Proof. We prove (S161) (as in this StackExchange response) by

$$\begin{aligned}\sum_{k=0}^{n} \cosh(kx+b) &= \frac{1}{2}\sum_{k=0}^{n}\left(e^b e^{kx} + e^{-b}e^{-kx}\right) = \frac{1}{2}\left(e^b\frac{1-e^{(n+1)x}}{1-e^x} + e^{-b}\frac{1-e^{-(n+1)x}}{1-e^{-x}}\right) \\ &= \frac{1}{2}\left(\frac{e^{\frac{n+1}{2}x+b}\left(e^{-\frac{n+1}{2}x} - e^{\frac{n+1}{2}x}+\right)}{e^{x/2}(e^{-x/2}-e^{x/2})} + \frac{e^{-(\frac{n+1}{2}x+b)}\left(e^{\frac{n+1}{2}x} - e^{-\frac{n+1}{2}x}\right)}{e^{-x/2}(e^{x/2}-e^{-x/2})}\right) \\ &= \frac{1}{2}\left(e^{nx/2+b} + e^{-(nx/2+b)}\right)\frac{\sinh\left(\frac{n+1}{2}x\right)}{\sinh(x/2)} = \cosh\left(\tfrac{n}{2}x+b\right)\frac{\sinh\left(\frac{n+1}{2}x\right)}{\sinh(x/2)} \\ &= \frac{\sinh\left(\frac{2n+1}{2}x+b\right) + \sinh\left(\frac{x}{2}-b\right)}{2\sinh(x/2)}.\end{aligned}$$

Analogously, we prove (S162) by

$$\sum_{k=0}^{n} \sinh(kx) = \sinh\left(\tfrac{n}{2}x\right)\frac{\sinh\left(\frac{n+1}{2}x\right)}{\sinh(x/2)} = \frac{\cosh\left(\frac{2n+1}{2}x\right) - \cosh\left(\frac{1}{2}x\right)}{2\cosh\left(\frac{1}{2}x\right)}.$$

□

S4.A.3. Fixed-point theorem.

In Section S4.E, we use a fixed-point theorem that works with a sequence of functions:

Theorem S4.4 (Gill (25))**.** *Let $(f_n)_n$ be a sequence of functions that are analytic on a simply connected region S and continuous on the closure S' of S. Define the outer compositional sequence as*

$$F_n(z) := f_n \circ \cdots \circ f_1(z). \tag{S163}$$

Suppose there exists a compact set Ω contained in S such that $\Omega \supset f_n(S)$ for all n. Furthermore, suppose the sequence of fixed points $(\alpha_n)_n$ of $(f_n)_n$ (in Ω) converges to a number α. Then $F_n(z) \to \alpha$ uniformly on S'.

S4.B. Upper bounds on α. We here construct a set of features that encodes overlapping item pairs as less similar than distinct item pairs. To do so, we modify a onehot encoding to contain negative loadings at a trial's overlapping trial loadings. Specifically, our features

$$\phi(i,j) \in \mathbb{R}^{n\times n},$$

are defined as

$$\phi_{kl}(i,j) = \begin{cases} 1 & \text{if } k = i \wedge l = j, \\ -a & \text{if } k = i \text{ xor } l = j, \\ 0 & \text{else.} \end{cases} \quad [S164]$$

a is a hyperparameter that will depend on n.

We compute the similarities as

$$s = 1 + 2(n-1)a^2, \quad [S165]$$
$$o = -2a + (n-2), \quad [S166]$$
$$d = 2a^2, \quad [S167]$$

and therefore

$$\alpha = 1 - 2\frac{n-2-2a-2a^2}{1+2(n-2)a^2}. \quad [S168]$$

As the determinant of the polynomial in the numerator, $4+8(n-2)$, is positive for all n, this polynomial can take on negative values for all n. As a result $\max_a \alpha > 1$ for all n, demonstrating that this construction encodes overlapping item pairs as less similar than distinct item pairs. Determining its value numerically, we find that it exponentially approaches 1 as n grows large. While we provide a lower bound on the upper bound on α, it remains unclear whether this is a tight lower bound or whether there is a construction that can attain higher values for α.

S4.C. Ridge regression. We recall Lemma S1.4:

Lemma S1.4. *For non-adjacent items,*

$$f(x_j, x_k) = r_j - r_k, \quad r_j = \frac{\sinh((\frac{n+1}{2} - j)\lambda)}{\sinh(\frac{n+1}{2}\lambda) - \sinh(\frac{n-1}{2}\lambda)}, \quad \lambda := \operatorname{arccosh}\left(\frac{1+\tilde{c}^{-1}}{1-\alpha}\right). \quad [S72]$$

For adjacent items,

$$f(x_j, x_{j+1}) = m \cdot 1 + (1-m)(r_j - r_{j+1}), \quad m := \frac{\alpha}{\alpha + \tilde{c}^{-1}}. \quad [S73]$$

Proof. We had established that

$$\overline{a} = \frac{1}{\delta_s} B^{-1}, \quad [S169]$$

where

$$B^{-1}_{ij} = \frac{\cosh\left((n - |j-i|)\lambda\right) - \cosh\left((n-i-j)\lambda\right)}{(1-\alpha)\sinh(\lambda)\sinh(n\lambda)}, \quad \lambda := \operatorname{arccosh}\left(\frac{1+\tilde{c}^{-1}}{1-\alpha}\right). \quad [S71 \text{ revisited}]$$

Thus

$$\overline{a}_i = \sum_{j=1}^{n-1} \frac{\cosh\left((n - |j-i|)\lambda\right) - \cosh\left((n-i-j)\lambda\right)}{(1-\alpha)\delta_s \sinh(\lambda)\sinh(n\lambda)}. \quad [S170]$$

Setting

$$b^{(1)}_i = \sum_{j=1}^{n-1} \cosh\left((n - |j-i|)\lambda\right), \quad b^{(2)}_i = \sum_{j=1}^{n-1} \cosh\left((n - (i+j))\lambda\right), \quad [S171]$$

we can express this as

$$\overline{a}_i = \frac{b^{(1)}_i - b^{(2)}_i}{2\delta_s \alpha \sinh(\lambda)\sinh(\lambda n)}. \quad [S172]$$

We simplify

$$\begin{aligned} b_i^{(1)} &= \sum_{l=n-i+1}^{n} \cosh(l\lambda) + \sum_{l=i+1}^{n-1} \cosh(l\lambda) \\ &= \frac{\sinh((n+\frac{1}{2})\lambda) - \sinh((n-i+\frac{1}{2})\lambda) + \sinh((n-\frac{1}{2})\lambda) - \sinh((i+\frac{1}{2})\lambda)}{\sinh(\lambda/2)}, \\ b_i^{(2)} &= \sum_{l=0}^{i-1} \cosh(l\lambda) + \sum_{l=1}^{n-i-1} \cosh(l\lambda) \\ &= \frac{\sinh(\lambda/2) + \sinh((i-\frac{1}{2})\lambda) + \sinh((n-i-\frac{1}{2})\lambda) - \sinh(\lambda/2)}{\sinh(\lambda/2)} \\ &= \frac{\sinh((i-\frac{1}{2})\lambda) + \sinh((n-i-\frac{1}{2})\lambda)}{\sinh(\lambda/2)}. \end{aligned} \tag{S173}$$

Moreover, note that

$$\frac{\sinh((m+\frac{1}{2})\lambda) + \sinh((m-\frac{1}{2})\lambda)}{2\sinh(\lambda/2)} = \frac{2\sinh(m\lambda)\cosh(\lambda/2)}{2\sinh(\lambda/2)} = \frac{\sinh(m\lambda)}{\tanh(\lambda/2)}. \tag{S174}$$

Applying this to $m \in \{n, n-i, i\}$ results in

$$\begin{aligned} \overline{a}_i &= \frac{\sinh(n\lambda) - (\sinh((n-i)\lambda) + \sinh(i\lambda))}{\delta_s(1-\alpha)\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)} \\ &= \frac{\sinh(n\lambda) - 2\sinh(\frac{n}{2}\lambda)\cosh((\frac{n}{2}-i)\lambda)}{\delta_s(1-\alpha)\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)}. \end{aligned} \tag{S175}$$

Note that the resulting boundary conditions are $\overline{a}_0 = \overline{a}_n = 0$, consistent with our definition.

Recalling that $r_j = \delta_s \frac{1-\alpha}{2}(\overline{a}_j - \overline{a}_{j-1})$, (S69), we find that

$$\begin{aligned} r_j &= -\frac{\sinh(n\lambda) - 2\sinh(\frac{n}{2}\lambda)\cosh((\frac{n}{2}-j)\lambda)}{2\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)} \\ &= \frac{\sinh(\frac{n}{2}\lambda)(\cosh((\frac{n}{2}+1-j)\lambda) - \cosh((\frac{n}{2}-j)\lambda))}{\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)} \\ &= \frac{2\sinh(\frac{n}{2}\lambda)\sinh((\frac{n+1}{2}-j)\lambda)\sinh(\lambda/2)}{\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)} \\ &= \frac{2\cosh(\lambda/2)\sinh(\frac{n}{2}\lambda)\sinh((\frac{n+1}{2}-j)\lambda)}{\sinh(\lambda)\sinh(\lambda n)} = C^{-1}\sinh((\tfrac{n+1}{2}-j)\lambda), \end{aligned} \tag{S176}$$

where

$$C := \frac{\sinh(\lambda)\sinh(\lambda n)}{2\cosh(\lambda/2)\sinh(\frac{n}{2}\lambda)}. \tag{S177}$$

Noting that

$$\begin{aligned} &\sinh(\tfrac{n+1}{2}\lambda) - \sinh(\tfrac{n-1}{2}\lambda) = 2\cosh(\tfrac{n}{2}\lambda)\sinh(\lambda/2) = \\ &\frac{2\cosh(\frac{n}{2}\lambda)\sinh(\lambda)}{\sqrt{2\cosh(\lambda)+1}} = \frac{\cosh(\frac{n}{2}\lambda)\sinh(\lambda)}{\cosh(\lambda/2)} = \\ &\frac{\sinh(\lambda)\sinh(\frac{n}{2}\lambda)\cosh(\frac{n}{2}\lambda)}{\cosh(\lambda/2)\sinh(\frac{n}{2}\lambda)} = C, \end{aligned} \tag{S178}$$

proves (S72).

Analysing the adjacent items, we observe that by definition,

$$\delta_s((1+\tilde{c}^{-1})\overline{a}_j - \tfrac{1-\alpha}{2}(\overline{a}_{j-1} + \overline{a}_{j+1})) = 1, \tag{S179}$$

and thus

$$f(x_j, x_{j+1}) = \delta_s(\overline{a}_j - \tfrac{1-\alpha}{2}(\overline{a}_{j-1} + \overline{a}_{j+1})) = 1 - \delta_s\tilde{c}^{-1}\overline{a}_j = 1 - \frac{\sinh(n\lambda) - 2\sinh(\frac{n}{2}\lambda)\cosh((\frac{n}{2}-j)\lambda)}{(1-\alpha)\tilde{c}\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)}. \tag{S180}$$

We split this up into two components: the memorization coefficient

$$m := 1 - \frac{\sinh(n\lambda)}{(1-\alpha)\tilde{c}\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)}, \tag{S181}$$

and the generalization coefficient g, which we pick such that

$$f(x_j, x_{j+1}) = m + g(r_j - r_{j+1}). \tag{S182}$$

We first simplify

$$\begin{aligned} m &= 1 - \frac{\tilde{c}^{-1} + 1 - 1}{(1-\alpha)\tanh(\lambda/2)\sinh(\lambda)} = 1 - \frac{(1-\alpha)\cosh(\lambda) - 1}{(1-\alpha)\tanh(\lambda/2)\sinh(\lambda)} \\ &= 1 - \frac{(1-\alpha)\cosh(\lambda) - 1}{(1-\alpha)(\cosh(\lambda) - 1)} = \frac{\alpha}{(1-\alpha)(\cosh(\lambda) - 1)} = \frac{\alpha}{\alpha + \tilde{c}^{-1}}. \end{aligned} \tag{S183}$$

To determine g, we solve the equation

$$g(r_j - r_{j+1}) = \frac{2\sinh(\frac{n}{2}\lambda)\cosh((\frac{n}{2} - j)\lambda)}{(1-\alpha)\tilde{c}\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)}. \tag{S184}$$

We observe

$$r_{j+1} - r_j = \frac{\sinh((j + 1 - \frac{n+1}{2})\lambda) - \sinh((j - \frac{n+1}{2})\lambda)}{\sinh((\frac{n+1}{2})\lambda) - \sinh((j + 1 - \frac{n-1}{2})\lambda)} = \frac{\cosh((j - \frac{n}{2})\lambda)}{\cosh(\frac{n}{2}\lambda)}. \tag{S185}$$

Thus,

$$g = \frac{2\sinh(\frac{n}{2}\lambda)\cosh(\frac{n}{2}\lambda)}{(1-\alpha)\tilde{c}\tanh(\lambda/2)\sinh(\lambda)\sinh(\lambda n)} = \frac{1}{(1-\alpha)\tilde{c}\tanh(\lambda/2)\sinh(\lambda)} = 1 - m, \tag{S186}$$

and we can write

$$f(x_j, x_{j+1}) = m + (1-m)(r_j - r_{j+1}), \tag{S187}$$

proving (S73). □

S4.D. Gradient descent dynamics for mean squared error. We recall Lemma S1.5:

Lemma S1.5. *Given the mean squared error as a loss function, g_j and r_j amount to*

$$g_j = \sum_{\substack{k=1 \\ k\ odd}} \frac{2\sin\left(\frac{jk\pi}{n}\right)\exp\left(-\delta_s\left(1 - 2\alpha\cos\left(\frac{k\pi}{n}\right)\right)t\right)}{n\tan\left(\frac{k\pi}{2n}\right)}. \tag{S101}$$

and

$$r_j(t) = (1-\alpha)\sum_{\substack{k=1 \\ k\ odd}} \frac{\left(\sin\left(\frac{jk\pi}{n}\right) - \sin\left(\frac{(j-1)k\pi}{n}\right)\right)\left(1 - \exp\left(-\delta_s\left(1 - (1-\alpha)\cos\left(\frac{k\pi}{n}\right)\right)t\right)\right)}{n\tan\left(\frac{k\pi}{2n}\right)\left(1 - (1-\alpha)\cos\left(\frac{k\pi}{n}\right)\right)}. \tag{S102}$$

Proof. We here solve the differential equation

$$\dot{g} = -Kg, \quad K_{ij} = \begin{cases} 1 & \text{if } i = j, \\ -\frac{1-\alpha}{2} & \text{if } |i-j| = 1, \\ 0 & \text{else.} \end{cases} \tag{S99 revisited}$$

Setting U as in Theorem S4.2 (meaning that $K = U\Lambda U$), we define

$$\tilde{g} := Ug, \tag{S188}$$

which results in the diagonalized equation

$$\dot{\tilde{g}}_j = -\delta_s\left(1 - (1-\alpha)\cos\left(\tfrac{j\pi}{m+1}\right)\right)\tilde{g}, \tag{S189}$$

We know that $g(0) = 1$ and so

$$\begin{aligned} \tilde{g}_j(0) = U_j 1 &= \sqrt{\frac{2}{n}}\sum_{k=1}^{n-1}\sin\left(\frac{jk\pi}{n}\right) = \frac{\cos\left(\frac{j\pi}{2n}\right) - \cos\left((n - \frac{1}{2})\frac{j\pi}{n}\right)}{\sqrt{2n}\sin\left(\frac{j\pi}{2n}\right)} \\ &= \frac{\cos\left(\frac{j\pi}{2n}\right) - \cos\left(j\pi - \frac{j\pi}{2n}\right)}{\sqrt{2n}\sin\left(\frac{j\pi}{2n}\right)}. \end{aligned} \tag{S190}$$

For odd j,

$$\cos\left(j\pi - \tfrac{j\pi}{2n}\right) = -\cos\left(\tfrac{j\pi}{2n}\right), \tag{S191}$$

and therefore

$$\tilde{g}_j(0) = \frac{\sqrt{2}}{\sqrt{n}\tan(\frac{j\pi}{2n})}, \tag{S192}$$

meaning that

$$\tilde{g}_j(t) = \frac{\sqrt{2}\exp\left(-\delta_s\left(1-(1-\alpha)\cos\left(\frac{j\pi}{n}\right)\right)t\right)}{\sqrt{n}\tan\left(\frac{j\pi}{2n}\right)}. \tag{S193}$$

For even j,

$$\cos\left(j\pi - \tfrac{j\pi}{2n}\right) = \cos\left(\tfrac{j\pi}{2n}\right), \tag{S194}$$

and therefore

$$\tilde{g}_j(0) = 0, \tag{S195}$$

meaning that

$$\tilde{g}_j(t) = 0. \tag{S196}$$

Therefore,

$$g_j(t) = \sum_{\substack{k=1\\ k\text{ odd}}} \frac{2\sin\left(\frac{jk\pi}{n}\right)\exp\left(-\delta_s\left(1-(1-\alpha)\cos\left(\frac{k\pi}{n}\right)\right)t\right)}{n\tan\left(\frac{k\pi}{2n}\right)}. \tag{S197}$$

This proves (S101).

As a reminder, the rank representation was defined as

$$r_i(0) := 0, \quad \dot{r}_i = \delta_s\tfrac{1-\alpha}{2}(g_i - g_{i-1}). \tag{S96 revisited}$$

As a result,

$$\begin{aligned} r_i(t) &= \delta_s\frac{1-\alpha}{2}\int_0^t g^{(i)}(s) - g^{(i-1)}(s)\mathrm{d}s \\ &= (1-\alpha)\sum_{\substack{j=1\\ j\text{ odd}}} \frac{\left(\sin\left(\frac{ij\pi}{n}\right) - \sin\left(\frac{(i-1)j\pi}{n}\right)\right)\left(1-\exp\left(-\delta_s\left(1-(1-\alpha)\cos\left(\frac{j\pi}{n}\right)\right)t\right)\right)}{n\tan\left(\frac{j\pi}{2n}\right)\left(1-(1-\alpha)\cos\left(\frac{j\pi}{n}\right)\right)}. \end{aligned} \tag{S198}$$

This proves (S102). □

S4.E. Deep neural networks' conjunctivity factor in the limit of infinite depth. We recall Lemma S1.3:

Lemma S1.3. *For a nonlinearity ϕ with relative slope ρ and a proportion of additive units λ, the infinite-depth conjunctivity factor*

$$\alpha^{(\infty)} := \lim_{l\to\infty}\alpha^{(l)}. \tag{S32}$$

is given by

$$\alpha^{(\infty)} = \begin{cases} 0 & \text{if } \lambda > 0 \vee \rho = 0, \\ 1 & \text{else.} \end{cases} \tag{S33}$$

Proof. Note that for $\rho = 0$, ϕ is the identity function and as a results $\alpha^{(l)} = 0$. We thus assume $\rho \neq 0$. As a reminder,

$$\hat{\kappa}_d^{(l+1)} = \tilde{k}(\hat{\kappa}_d^{(l)}), \quad \hat{\kappa}_o^{(l+1)} = (1-\lambda)\tilde{k}(\hat{\kappa}_o^{(l)}) + \frac{\lambda}{2}(\hat{\kappa}_d^{(l+1)} + 1), \tag{S199}$$

where

$$\tilde{k}(u) = u + \frac{\rho}{\pi}\left(\sqrt{1-u^2} - \cos^{-1}(u)u\right). \tag{S200}$$

First, $\tilde{k}([-1,1]) = [0,1]$. Further, $\tilde{k}$ has precisely one fixed point at $u = 1$. As a result,

$$\hat{\kappa}_d^{(l)} \to 1, \quad \hat{\kappa}_o^{(l)} \to 1. \tag{S201}$$

Note that

$$\alpha^{(l+1)} = (1-\lambda)\frac{1+\hat{\kappa}_d^{(l+1)} - 2\hat{\kappa}_o^{(l+1)}}{1-\hat{\kappa}_d^{(l+1)}} = (1-\lambda)\frac{1+\hat{\kappa}_d^{(l)} - 2\hat{\kappa}_o^{(l)} + h(\hat{\kappa}_d^{(l)}) - 2h(\hat{\kappa}_o^{(l)})}{1-\hat{\kappa}_d^{(l)} - h(\hat{\kappa}_d^{(l)})}, \tag{S202}$$

where

$$h(u) = \frac{\rho}{\pi}\left(\sqrt{1-u^2} - \cos^{-1}(u)u\right). \tag{S203}$$

We further simplify

$$\alpha^{(l+1)} = (1-\lambda)\frac{(1-\hat{\kappa}_d^{(l)})\alpha^{(l)} + h(\hat{\kappa}_d^{(l)}) - 2h(\hat{\kappa}_o^{(l)})}{1-\hat{\kappa}_d^{(l)} - h(\hat{\kappa}_d^{(l)})}. \tag{S204}$$

Noting that

$$\hat{\kappa}_o^{(l)} = \frac{1+\alpha}{2}\hat{\kappa}_d^{(l)} + \frac{1-\alpha}{2}, \tag{S205}$$

we can define updates to $\alpha^{(l)}$ by the iterated map

$$\alpha^{(l+1)} = f_l(\alpha^{(l)}) := f(\alpha^{(l)}, \hat{\kappa}_d^{(l)}), \tag{S206}$$

where

$$f(\alpha, d) := (1-\lambda)\frac{(1-d)\alpha + h(d) - 2h\left(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2}\right)}{1-d-h(d)}. \tag{S207}$$

Case $\lambda = 0$. For $\lambda = 0$, the fixed point equation amounts to

$$\alpha_d^*(1-d-h(d)) = (1-d)\alpha_d^* + h(d) - 2h\left(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2}\right), \tag{S208}$$

which is equivalent to

$$-h(d)\alpha_d^* = h(d) - 2h\left(\frac{1+\alpha_d^*}{2}d + \frac{1-\alpha_d^*}{2}\right), \tag{S209}$$

and thus

$$2h\left(\frac{1+\alpha_d^*}{2}d + \frac{1-\alpha_d^*}{2}\right) - h(d)(1+\alpha_d^*) =: g_d(\alpha_d^*) = 0. \tag{S210}$$

Clearly, $g_d(1) = 0$. Note that

$$h'(u) = \frac{\rho}{\pi}\left(-\frac{u}{\sqrt{1-u^2}} - \cos^{-1}(u) + \frac{u}{\sqrt{1-u^2}}\right) = -\frac{\rho}{\pi}\cos^{-1}(u). \tag{S211}$$

Therefore,

$$g_d'(\alpha) = \frac{\rho}{\pi}\cos^{-1}\left(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2}\right)(1-d) - h(d) \tag{S212}$$

As $\cos^{-1}$ is monotonically decreasing,

$$g_d'(\alpha) \geq g_d'(0) = \frac{\rho}{\pi}\cos^{-1}\left(\frac{1}{2}d + \frac{1}{2}\right)(1-d) - h(d). \tag{S213}$$

Visual inspection of this plot reveals $g_d'(0) \geq 0$ for all d. As a result, g_d is monotonically increasing on α and can therefore only have one zero. Put differently, the fixed point $\alpha_d^* = 1$ is unique. This means that each f_l has a unique fixed point at 1, and, by Theorem S4.4, $\alpha^{(l)} \to 1$.

Case $\lambda > 0$. Because $f(\cdot, d)$ maps $[0,1]$ onto $[0,1]$, it has a fixed point. If

$$\lim_{d\to 1}\frac{1-d}{1-d-h(d)} = 1, \quad \forall_{\alpha\in[0,1]}\lim_{d\to 1}\frac{h(d) - 2h(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2})}{1-d-h(d)} = 0, \tag{S214}$$

we can write

$$f(\alpha, d) = (1-\lambda)(1+\epsilon_d^{(1)})\alpha + \epsilon_d^{(2)}(\alpha), \quad \lim_{d\to 1}\epsilon_d^{(1)} = 0, \quad \lim_{d\to 1}\epsilon_d^{(2)} \equiv 0. \tag{S215}$$

For any fixed point α_d^*, we thus observe that

$$(\lambda - (1-\lambda)\epsilon_d^{(1)})\alpha_d^* = \epsilon_d^{(2)}(\alpha). \tag{S216}$$

As $(\lambda - (1-\lambda)\epsilon_d^{(1)}) \to \lambda > 0$, $\alpha_d^* \to 0$. This means that any sequence of fixed points given by f_l as $l \to \infty$ must converge to zero, and, as a result, $\alpha^{(l)} \to 0$.

All that remains to show is (S214). We first note that $0 \leq h(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2}) \leq h(d)$ and thus

$$-\frac{h(d)}{1-d-h(d)} \leq \frac{h(d) - 2h(\frac{1+\alpha}{2}d + \frac{1-\alpha}{2})}{1-d-h(d)} \leq \frac{h(d)}{1-d-h(d)}. \tag{S217}$$

As the left limit in (S214) implies

$$\lim_{d\to 1}\frac{h(d)}{1-d-h(d)} = 0, \tag{S218}$$

it immediately implies the right half of the equation. All that remains to show is the left half. Note that it is equivalent to

$$\lim_{d\to 1} \frac{1-d-h(d)}{1-d} = 1, \tag{S219}$$

and thus

$$\lim_{d\to 1} -\frac{h(d)}{1-d} = 0. \tag{S220}$$

We observe that

$$-\frac{h(d)}{1-d} = \frac{h(1)-h(d)}{1-d}, \tag{S221}$$

and thus

$$\lim_{d\to 1} -\frac{h(d)}{1-d} = h'(1) = 0, \tag{S222}$$

which concludes the proof. □

S4.F. Details on the cooperative coding scheme. We describe the construction of the units with positive readout weights. We make sure that these units are silent on data points with negative labels. The units with negative readout weights are constructed analogously and therefore silent on data points with positive labels. Because the induction step is a bit complicated, we describe the first few cases ($n = 2, \dots, 6$) concretely. All unit weights will either be zero or 0.5. We give an example of the specified code for $n = 2, \dots, 7$ in Table S2. Overall, there are five cases to be considered. Our induction step for each case will be that all trials involving the first $n-1$ items are correctly classified. Note that from now on we will say that some unit u needs to encode x if we require that $\langle u, x\rangle$. Further, we say that u needs to be silent on x if we require that $\langle u, x\rangle \leq 0$.

Case 1: $n = 2$. H_{1+} needs to encode $(1, 2)$ and be silent on $(2, 1)$. It can do so simply by setting $\mathrm{H}_{1+}(1, 1) = \mathrm{H}_{1+}(2, 2) = 0.5$. Thus, the first unit has two weights added and the second unit has no weights added (as it is not needed at all).

Case 2: $n = 4k+3$ **(e.g.** $n = 3, 7$**).** This step adds the trial $(4k+2, 4k+3)$, which is encoded by H_{1+}, (as well as $(4k+3, 4k+2)$). Because H_{1+} also encodes $(4k+1, 4k+2)$ and therefore $\mathrm{H}_{1+}(4k+2, 2) = 0.5$, it needs to add a negative weight $\mathrm{H}_{1+}(4k+3, 1) = -0.5$. To encode $(4k+2, 4k+3)$, we add $\mathrm{H}_{1+}(4k+2, 1) = \mathrm{H}_{1+}(4k+3, 2) = 0.5$. Finally, to remain silent on $(4k+2, 4k+1)$, we add a negative weight $\mathrm{H}_{1+}(4k+1, 2) = -0.5$. In total there are **four weights added to** $\mathbf{H}_{1+}$. On the other hand, $\mathbf{H}_{2+}$ **needs no units added** as it is already silent on both added data points (we can check this by checking the two preceding cases 1 and 5).

Case 3: $n = 4k+4$ **(e.g.** $n = 4, 8$**).** This steps adds the trial $(4k+3, 4k+4)$, which is encoded by H_{2+} (as well as $(4k+4, 4k+3)$). We therefore need to add $\mathrm{H}_{2+}(4k+3, 1) = p_{2+}(4k+4, 2) = 0.5$. To maintain silence on $(4k+3, 4k+2)$, we also need to add $\mathrm{H}_{2+}(4k+2, 2) = -0.5$ (which was not needed so far, as can be checked from case 2). In total, we have added **three weights to** $\mathbf{H}_{2+}$. For H_{1+}, on the other hand, the positive weight at $\mathrm{H}_{1+}(4k+3, 2)$ (per case 2) needs to be counter-balanced by setting $\mathrm{H}_{1+}(4k+4, 1) = -0.5$. We therefore add **one weight to** $\mathbf{H}_{1+}$.

Case 4: $n = 4k+5$ **(e.g.** $n = 5$**).** This step adds the trial $(4k+4, 4k+5)$, which is encoded by H_{2+} (as well as $(4k+5, 4k+4)$). Note that it is somewhat analogous to case 2 except with the roles of H_{1+} and H_{2+} being exchanged. Specifically, because H_{2+} also encodes $(4k+3, 4k+4)$, we need to add a negative weight $\mathrm{H}_{2+}(4k+5, 1) = -0.5$ to maintain silence on $(4k+5, 4k+4)$. Additionally we need to add positive weights $\mathrm{H}_{2+}(4k+4, 1) = \mathrm{H}_{2+}(4k+5, 2) = 0.5$. To maintain silence on $(4k+4, 4k+3)$, we finally need to add $\mathrm{H}_{2+}(4k+3, 2) = -0.5$ (which had no weight assigned as per cases 2 and 3). In total we added **four weights to** $\mathbf{H}_{2+}$ and **zero weights to** $\mathbf{H}_{1+}$.

Case 5: $n = 4k+6$ **(e.g.** $n = 6$**).** This step adds the trial $(4k+5, 4k+6)$, which is encoded by H_{1+} (as well as $(4k+6, 4k+5)$). Note that it is analogous to case 3 with the roles of H_{1+} and H_{2+} being exchanged and we need to add **three weights to** $\mathbf{H}_{1+}$ and **one weight to** $\mathbf{H}_{2+}$.

In total, this means that we can define two recursive functions $c_1(n), c_2(n)$ that count the weights of $\mathrm{H}_{1\pm}$ and $\mathrm{H}_{2\pm}$, respectively. The baseline conditions are

$$c_1(2) = 2, \quad c_2(2) = 0. \tag{S223}$$

The increment is defined as

$$c_1(n) = c_1(n-1) + \begin{cases} 4 \text{ if } n = 4k+3, \\ 1 \text{ if } n = 4k+4, \\ 0 \text{ if } n = 4k+5, \\ 3 \text{ if } n = 4k+6, \end{cases} \qquad c_2(n) = c_2(n-1) + \begin{cases} 0 \text{ if } n = 4k+3, \\ 3 \text{ if } n = 4k+4, \\ 4 \text{ if } n = 4k+5, \\ 1 \text{ if } n = 4k+6. \end{cases} \tag{S224}$$

Note that $c_1(n), c_2(n) \approx 2(n-1)$.

We can then compute the norm as

$$2\sum_{\Delta} \|\mathrm{H}_{\Delta}\|_2 = 4\sqrt{\tfrac{1}{2}}\left(\sqrt{c_1(n)} + \sqrt{c_2(n)}\right) \approx 4\sqrt{2}\sqrt{n-1}. \tag{S225}$$

	n=2		n=3		n=4		n=4		n=5		n=5		n=6		n=6		n=7		n=7	
	p1+		p1+		p1+		p2+		p1+		p2+		p1+		p2+		p1+		p2+	
1	**0.5**		0.5	**-0.5**	0.5	-0.5			0.5	-0.5			0.5	-0.5			0.5	-0.5		
2		**0.5**	**0.5**	0.5	0.5	0.5		**-0.5**	0.5	0.5		-0.5	0.5	0.5		-0.5	0.5	0.5		-0.5
3			**-0.5**	**0.5**	-0.5	0.5	**0.5**		-0.5	0.5	0.5	**-0.5**	-0.5	0.5	0.5	-0.5	-0.5	0.5	0.5	-0.5
4					**-0.5**			**0.5**	-0.5		**0.5**	0.5	-0.5	**-0.5**	0.5	0.5	-0.5	-0.5	0.5	0.5
5											**-0.5**	**0.5**	**0.5**		-0.5	0.5	0.5	**-0.5**	-0.5	0.5
6														**0.5**	**-0.5**		**0.5**	0.5	-0.5	
7																	**-0.5**	**0.5**		

Table S2. Weights of the hand-constructed network for $n = 2, \ldots, 7$. Rows reference responses to the different units. First column for each unit is response to the item when in the first position, second column is response to the item when in the second position. Entries that are new between consecutive numbers of items are highlighted in bold.

S1 Software. Code used to implement simulations and their analysis.

References

1. K Fukushima, Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements. *IEEE Transactions on Syst. Sci. Cybern.* **5**, 322–333 (1969) Conference Name: IEEE Transactions on Systems Science and Cybernetics.
2. AL Maas, AY Hannun, AY Ng, others, Rectifier nonlinearities improve neural network acoustic models in *Proc. icml*. (Atlanta, Georgia, USA), Vol. 30, p. 3 (2013) Issue: 1.
3. Y Cho, L Saul, Kernel Methods for Deep Learning in *Advances in Neural Information Processing Systems*. (Curran Associates, Inc.), Vol. 22, (2009).
4. R Tsuchida, F Roosta-Khorasani, M Gallagher, Invariance of Weight Distributions in Rectified MLPs (2018) arXiv:1711.09090 [cs, stat].
5. R Tsuchida, F Roosta, M Gallagher, Richer priors for infinitely wide multi-layer perceptrons (2019) arXiv:1911.12927 [cs, stat].
6. I Han, et al., Fast Neural Kernel Embeddings for General Activations (2022) arXiv:2209.04121 [cs, stat].
7. B Schölkopf, The Kernel Trick for Distances in *Advances in Neural Information Processing Systems*. (MIT Press), Vol. 13, (2000).
8. GY Hu, RF O'Connell, Analytical inversion of symmetric tridiagonal matrices. *J. Phys. A: Math. Gen.* **29**, 1511 (1996).
9. D Soudry, E Hoffer, MS Nacson, S Gunasekar, N Srebro, The implicit bias of gradient descent on separable data. *The J. Mach. Learn. Res.* **19**, 2822–2878 (2018) Publisher: JMLR. org.
10. K He, X Zhang, S Ren, J Sun, Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. pp. 1026–1034 (2015).
11. R Core Team, *R: A Language and Environment for Statistical Computing*. (R Foundation for Statistical Computing, Vienna, Austria), (2022).
12. H Wickham, *ggplot2: Elegant Graphics for Data Analysis*. (Springer-Verlag New York), (2016).
13. TL Pedersen, *patchwork: The Composer of Plots*. (2022).
14. A Paszke, et al., PyTorch: An Imperative Style, High-Performance Deep Learning Library in *Advances in Neural Information Processing Systems 32*. (Curran Associates, Inc.), pp. 8024–8035 (2019).
15. A Jacot, F Gabriel, C Hongler, Neural tangent kernel: Convergence and generalization in neural networks. *Adv. neural information processing systems* **31** (2018).
16. J Lee, et al., Wide neural networks of any depth evolve as linear models under gradient descent. *Adv. neural information processing systems* **32** (2019).
17. L Chizat, F Bach, Implicit Bias of Gradient Descent for Wide Two-layer Neural Networks Trained with the Logistic Loss in *Proceedings of Thirty Third Conference on Learning Theory*. (PMLR), pp. 1305–1338 (2020) ISSN: 2640-3498.
18. K Lyu, J Li, Gradient Descent Maximizes the Margin of Homogeneous Neural Networks in *International Conference on Learning Representations*. (2020).
19. S Nelli, L Braun, T Dumbalska, A Saxe, C Summerfield, Neural knowledge assembly in humans and neural networks. *Neuron* (2023) Publisher: Elsevier.
20. P Savarese, I Evron, D Soudry, N Srebro, How do infinite width bounded norm networks look in function space? in *Proceedings of the Thirty-Second Conference on Learning Theory*, Proceedings of Machine Learning Research, eds. A Beygelzimer, D Hsu. (PMLR), Vol. 99, pp. 2667–2690 (2019).
21. G Ongie, R Willett, D Soudry, N Srebro, A Function Space View of Bounded Norm Infinite Width ReLU Nets: The Multivariate Case in *International Conference on Learning Representations*. (2020).
22. DE Knuth, Johann Faulhaber and sums of powers. *Math. Comput.* **61**, 277–294 (1993).
23. N Razin, N Cohen, Implicit Regularization in Deep Learning May Not Be Explainable by Norms in *Advances in Neural Information Processing Systems*. (Curran Associates, Inc.), Vol. 33, pp. 21174–21187 (2020).
24. A Böttcher, SM Grudsky, *Spectral properties of banded Toeplitz matrices*. (SIAM), (2005).
25. J Gill, The use of the sequence Fn(z)=fnf1(z) in computing fixed points of continued fractions, products, and series. *Appl. Numer. Math.* **8**, 469–476 (1991).