

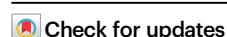
Modeling rapid language learning by distilling Bayesian priors into artificial neural networks

Received: 26 October 2023

R. Thomas McCoy^{1,2}  & Thomas L. Griffiths^{3,4} 

Accepted: 7 May 2025

Published online: 20 May 2025



Humans can learn languages from remarkably little experience. Developing computational models that explain this ability has been a major challenge in cognitive science. Existing approaches have been successful at explaining how humans generalize rapidly in controlled settings but are usually too restrictive to tractably handle naturalistic data. We show that learning from limited naturalistic data is possible with an approach that bridges the divide between two popular modeling traditions: Bayesian models and neural networks. This approach distills a Bayesian model's inductive biases—the factors that guide generalization—into a neural network that has flexible representations. Like a Bayesian model, the resulting system can learn formal linguistic patterns from limited data. Like a neural network, it can also learn aspects of English syntax from naturally-occurring sentences. Thus, this model provides a single system that can learn rapidly and can handle naturalistic data.

Across a remarkably wide range of settings, people make rich generalizations from limited experience. This ability is particularly apparent in the case of language, making it a classic setting for debates about learning. From a small number of examples, people can learn new word meanings^{1–3}, new syntactic structures^{4–7}, and new phonological rules^{8–11}. A central challenge in cognitive science is understanding how people can infer so much about language from so little evidence^{12,13}. This puzzle is so extensively discussed that it has accumulated a number of different names, including the poverty of the stimulus¹⁴, Plato's problem¹⁵, and the logical problem of language acquisition¹⁶.

One popular approach for explaining rapid learning is to use probabilistic models based on Bayesian inference^{17–21}. These models make strong commitments about how hypotheses are represented and selected, resulting in strong inductive biases—factors that determine how a learner generalizes beyond its experience²². Bayesian models are thus well-suited for capturing the ability to learn from few examples. For instance, a recent Bayesian model introduced by Yang and Piantadosi²³ showed that it is possible to learn many important aspects of syntax from 10 or fewer examples. However, when Bayesian

models are applied to larger datasets, they face significant challenges in specifying hypotheses that are flexible enough to capture the data yet remain computationally tractable.

Another influential modeling approach is to use neural networks^{24–26}, which make few high-level commitments, giving them the flexibility to capture the nuances of realistic data. These systems represent hypotheses with matrices of numerical connection weights, and they use data-driven learning to find the best connection weights for the task at hand. When data are plentiful, this approach is highly successful, yielding state-of-the-art systems such as the recent language model ChatGPT²⁷. Nonetheless, the flexibility of neural networks is accompanied by weak inductive biases, making them perform poorly in settings with little available data.

We argue that accounting for rapid learning from naturalistic data requires disentangling representations and inductive biases. These two factors are in principle distinct, yet historically certain types of inductive biases have always been paired with certain types of representations (Fig. 1a): strong inductive biases—which are important for rapid learning—have historically come with strong representational commitments (as in Bayesian models), while weak representational

¹Department of Linguistics, Yale University, 370 Temple St, New Haven, CT 06511, USA. ²Wu Tsai Institute, Yale University, 100 College St, New Haven, CT 06510, USA. ³Department of Psychology, Princeton University, South Drive, Princeton, NJ 08540, USA. ⁴Department of Computer Science, Princeton University, 35 Olden Street, Princeton, NJ 08540, USA. ✉e-mail: tom.mccoy@yale.edu

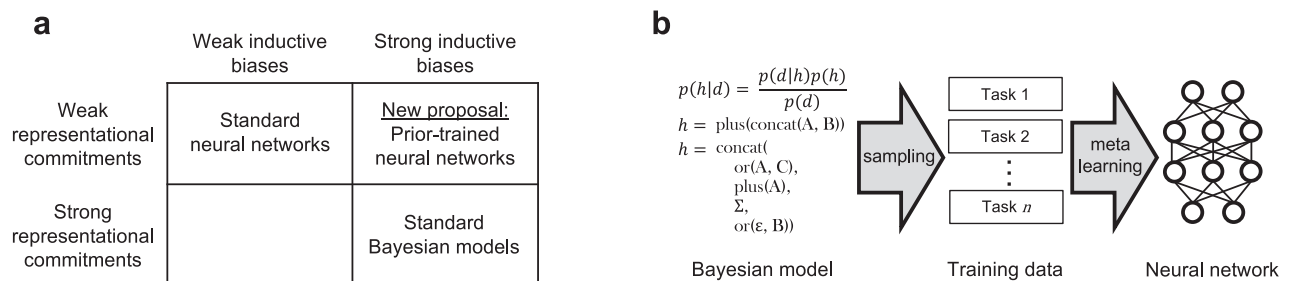


Fig. 1 | Overview of our approach. a Standard models of learning conflate strength of inductive biases with strength of representational commitments: Bayesian models have strong biases and strong representational commitments, while standard neural networks have weak biases and weak representational commitments. In this work, we create prior-trained neural networks—neural networks that have strong biases yet flexible representations. **b** The process of inductive bias

distillation that we use to give strong inductive biases to neural networks. First, a target inductive bias is instantiated in a Bayesian model which gives a prior over hypotheses. Then, hypotheses are sampled from that prior to create tasks that instantiate the inductive bias in data. Finally, a neural network meta-learns from the data, a step which transfers the inductive bias into the network.

commitments—which provide the flexibility needed to process complex naturalistic data—have historically come with weak inductive biases (as in neural networks). In principle, decoupling these factors would make it possible to create a system that has strong inductive biases yet weak representational commitments, enabling it—like humans—to learn rapidly without sacrificing the ability to develop more complex hypotheses. In practice, however, it remains far from obvious what sort of system could have both of these traits.

In this work, we show how the inductive biases of a Bayesian model can be distilled into a neural network. Our approach makes use of recent^{28,29} technical advances in meta-learning, a machine learning technique in which a system is shown a variety of tasks from which it automatically finds an inductive bias that enables it to learn new tasks more easily^{30,31}. In our application of meta-learning, the tasks are sampled from a Bayesian model, thereby distilling inductive biases from the Bayesian model into the neural network. The result of this procedure, which we call inductive bias distillation, is a system that has the strong inductive biases of a Bayesian model but the flexibility of a neural network.

We use this approach to create a model of language learning. We chose this case study because language learning is a classic problem that has long seemed to require structured symbolic representations, making it a challenging test for neural-network-based approaches. In a setting with limited data (learning artificial formal languages from a small number of examples), our model's performance is close to that of Yang & Piantadosi's Bayesian learner, which was the first model shown to be able to learn such languages from limited data without being substantially tailored to specific linguistic phenomena. Thus, even though our model is a neural network, its distilled inductive biases enable it to succeed in an environment where neural networks typically struggle, achieving a level of performance previously achieved only by a model using symbolic representations. In addition, the fact that our model is a neural network makes it flexible enough to handle a setting that is intractable for the Bayesian model: learning elements of English syntax from a corpus of 8.6 million words. Our results illustrate the possibility—and the benefits—of combining the complementary strengths of Bayesian models and neural networks.

Results

Model: inductive bias distillation

As illustrated in Fig. 1b, inductive bias distillation uses three steps to distill an inductive bias (called the target bias) into a model (called the student model). First, the target bias is defined with a Bayesian model, whose prior gives a distribution over tasks. Second, many tasks are sampled from this distribution. Finally, the student model meta-learns from these sampled tasks to gain inductive biases that allow it to learn

new tasks more easily. By controlling the Bayesian model, we control the student model's meta-learned inductive biases.

This approach is very general: the target bias can be characterized with any distribution that can be sampled from, and the student model can be any system that is able to perform meta-learning. In our specific case, each task is a language so that the inductive bias being distilled is a prior over the space of languages³². Our student model is a neural network, meaning that we distill the linguistic priors of a Bayesian model into a neural network. This approach extends the method from our previous proof-of-concept work³³ by using a structured probabilistic model to define the inductive bias and by testing the model in both artificial and naturalistic scenarios. In the rest of this section, we describe the specific form of inductive bias distillation that we perform for our language case study.

Step 1: Characterizing the inductive bias

Our starting point is the model that Yang and Piantadosi proposed for creating a prior over formal languages²³. A formal language^{34–37} is a set of strings defined by an abstract rule. For example, the set $\{AB, ABAB, ABABAB, \dots\}$ is a formal language defined by the expression $(AB)^+$, meaning one or more copies of AB . The mechanisms used to define formal languages are inspired by the structure of natural language. The case of $(AB)^+$ parallels tail recursion as seen in nested English prepositional phrases: if we take A to stand for a preposition and B to stand for a noun phrase, then $(AB)^+$ captures strings that alternate between prepositions and noun phrases, such as *under the vase on the table in the library*. By translating linguistic structure into precise abstractions, formal languages have long facilitated mathematical analyses of language^{38–41}.

In our case, the mathematical nature of formal languages makes them useful for defining a distribution over languages. Following the general approach used by Yang and Piantadosi, we specify a set of formal primitives and construct a model which probabilistically combines these primitives to create definitions of languages. The primitives that we use are mainly drawn from standard components of regular expressions⁴², one particular formal language notation. Examples of these primitives include concatenation and the aforementioned recursion primitive *plus* meaning one or more copies of. An example language defined by our primitives is $\text{concat}(A, \text{plus}(C), \text{or}(F, B))$, the language of strings made of an A followed by one or more C 's followed by either F or B : $\{ACF, ACB, ACCF, ACCB, ACCCF, \dots\}$. Regular expressions are limited in their expressive power: they are provably unable to capture certain aspects of natural language syntax⁴³. To overcome these limitations, we augment the basic regular expression primitives in ways that increase the system's expressivity. See “Methods” and Supplementary Methods for full descriptions of our primitives.

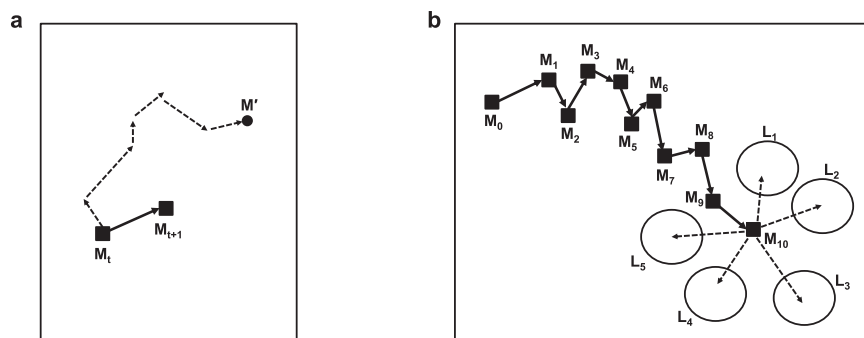


Fig. 2 | Meta-learning using MAML. We illustrate a model M as a point in parameter space. **a** A single episode of meta-learning. We sample one language L from a space of synthetic languages that we have defined and then sample two sets of sentences from this language: a training set and a test set. The model begins the episode with parameter values M_t . These parameter values are copied into a temporary model M' , which then learns from the training set for L using standard (non-meta) learning (dotted trajectory). The trained M' is then evaluated on the test set of language L . Based on the errors that M' makes on this test set, the parameters of the original model M_t are adjusted (following the solid arrow) to create a new set of parameters M_{t+1} , such that if M_{t+1} were duplicated into a new M' the new M' would learn this language more effectively than if it had been copied from M_t . **b** A complete meta-

learning process encompassing 10 episodes. This diagram shows the model starting with random initial parameters M_0 and then going through 10 episodes (solid arrows) until arriving at its final parameters M_{10} ; note that our experiments actually use 25,000 meta-learning episodes, not 10 episodes. If meta-learning has succeeded, these final parameter values should serve as a useful initialization from which the model can easily learn any language in our space of languages. Here, each oval represents the region of parameter space that would lead to effective processing of a single language such as L_2 . The meta-learned parameter values M_{10} indeed position the model in a region where it can readily reach any of the languages shown through a small amount of standard learning (dotted arrows).

Our complete distribution over languages is specified via a probabilistic model (structured similarly to a probabilistic context-free grammar) that defines a probability distribution over all possible combinations of our primitives. This approach assigns high probability to languages defined with few primitives and low probability to languages with more complex descriptions. Therefore, the inductive bias that we aim to distill using this model is one that favors languages that can be simply expressed using our chosen primitives. By specifying the target bias with a probabilistic model, we make this target bias interpretable and controllable—properties that would not hold if we instead defined the target bias with a neural network, as done by Abnar et al.'s work about transferring inductive biases from one type of neural network to another⁴⁴.

Step 2: Sampling data

Now that we have characterized the inductive bias as a distribution over languages, the next step is to sample languages from this distribution so that the student model can meta-learn from these languages. This step is straightforward because the distribution is defined as a generative model, which automatically allows us to sample languages from this distribution and to then sample specific strings from each language. Although it is simple, this step is conceptually important. It bridges the divide between our probabilistic model and our neural network by instantiating our target bias in data—something that can act as common ground between two otherwise very different models.

Step 3: Applying meta-learning

The final step in inductive bias distillation is to have the student model meta-learn from our sampled data in order to give it the target bias. The type of system that we use as a student model is a long short-term memory neural network (LSTM; ref. 45). LSTMs have been formally shown to be capable of processing many types of formal languages⁴⁶, and empirically they have been very successful in processing natural language^{47–49}. We also tried using Transformers⁵⁰—another type of neural network that is effective for language—but we found that the distillation was not as effective for Transformers as for LSTMs, likely because LSTMs perform better than Transformers at capturing some of the formal language mechanisms underlying our set of primitives⁵¹.

The task that our LSTMs perform is next-word prediction⁵², also known as language modeling: Given a sequence, the LSTM aims to

predict each word in the sequence conditioned on the previous words. For example, if the sequence is *ABA*, it would be expected to first predict what the first token is (*A*); then to predict what the second token is (*B*) given that the first token is *A*; then to predict what the third token is (*A*) conditioned on the prefix *AB*; and finally to produce a special end-of-sequence token conditioned on the prefix *ABA*. For most languages, this task cannot be solved perfectly; e.g., in English, there are many next words that could follow *The*. Therefore, the model's prediction for the next word is a probability distribution over all possible tokens (ideally assigning the highest probability to the most likely next words). We use the next-word prediction task because prior work has found it to be effective in teaching neural networks the grammatical properties of languages^{53–56} and has argued that it plays a central role in human language processing^{57,58}.

Before we describe meta-learning, it is helpful to first describe standard learning. A neural network is defined by a large number of numerical parameters, such as connection weights. In standard learning, the network starts with some initial parameter values (usually random ones). The network is then shown many examples of the target task. After each example, the network's parameters are adjusted such that, if the network saw the same example again, it would perform slightly better on it. After many such updates, the network should have parameter values that enable it to perform the task effectively.

Various types of meta-learning have been shown to improve the generalization abilities of neural networks^{59–64}. The form of meta-learning that we use is model-agnostic meta-learning (MAML; ref. 28). MAML can be viewed as a way to perform hierarchical Bayesian modeling⁶⁵, making it a natural choice for our purpose of distilling Bayesian priors. Intuitively, in our application of MAML, a network is shown many languages and thereby learns how to learn new languages more easily. What is updated throughout the MAML process is the network's initialization—the parameter values that it starts with before learning a specific language. If MAML is successful, the initialization that results from it should encode an inductive bias that enables the model to learn any language in our distribution from relatively few examples. Because we control the distribution of languages, we also control the inductive bias that is meta-learned. See Fig. 2 for a more detailed illustration of this process, and see Supplementary Methods for the complete MAML algorithm that we use. We refer to a neural network that has undergone inductive bias distillation as a prior-trained neural network because it has been given a particular prior via

training. Prior-training is superficially similar to another approach called pre-training, but there are important differences in what these methods achieve; see “Discussion”.

In inductive bias distillation, meta-learning is not a hypothesis about how humans might have come by their inductive biases. Though humans certainly perform meta-learning in at least some cases^{66–68}, we do not claim that humans’ linguistic inductive biases must arise via meta-learning, nor do we claim that these inductive biases are encoded in the form that MAML encodes them (i.e., via the initial settings of connection weights). Instead, we use meta-learning purely as a tool for creating a model that has specific inductive biases. See ref. 69 for discussion of meta-learning as a source of priors in humans.

Our goal in using inductive bias distillation is to combine the strong inductive biases of a Bayesian model with the representational flexibility of a neural network. To test whether our model captures the strengths of both approaches, we evaluate it in two settings: one setting that has traditionally been well-handled by Bayesian models but not neural networks, and another setting where the reverse is true.

Learning formal languages

We first evaluate our model on its ability to learn formal languages from few examples, an area where Bayesian models perform well but standard neural networks perform poorly. We use the same 56 formal languages that Yang and Piantadosi used to evaluate their Bayesian learner. For each evaluation language, we train our model on n strings drawn from that language, with n ranging from 1 to 10,000 on a logarithmic scale. To quantify how well the trained model has learned the intended language, we compute the model’s F-score, the same metric used by Yang and Piantadosi. The F-score quantifies how closely the set of strings assigned highest probability by the model matches the set of strings that have the highest probability in the true language (see “Methods”). We also compare prior-trained networks to standard neural networks, which have the same architecture as the prior-trained networks but which have their weights initialized randomly rather than via inductive bias distillation.

This setting poses a substantial challenge for a neural network because the formal languages are defined in discrete symbolic terms. Neural networks have long been viewed as fundamentally different from symbolic processing. Indeed, a major puzzle in cognitive science is the fact that the human mind is instantiated in a neural network yet is able to perform symbolic functions^{70–74}—a fact so puzzling that Smolensky and Legendre refer to it as “the central paradox of cognition”⁷⁵. Therefore, this setting provides a challenging test of the claim that strong inductive biases can be distilled into a neural network.

Although it is a neural network, our prior-trained model displays a data efficiency similar to that of Yang and Piantadosi’s symbolic Bayesian learner (Fig. 3). In contrast, a standard neural network is substantially more data-hungry: To reach a given level of performance, it needs about 10 times as many examples as the Bayesian learner. The standard and prior-trained neural networks are identical in their architecture and the procedure that they use to learn a given formal language. They only differ in that the prior-trained network has undergone inductive bias distillation while the standard one has not. Therefore, the distillation process has succeeded at giving our model inductive biases that are useful for learning formal languages. Although neural networks are typically associated with learning slowly, these results show that learning slowly is not a necessary aspect of being a neural network.

In addition to coming close to the Bayesian learner’s data efficiency, the prior-trained network exceeds the Bayesian learner’s time efficiency. To learn one formal language, the Bayesian learner takes from 1 min to 7 days. Our neural network takes at most 5 min, and sometimes as little as 10 ms. The Bayesian learner is by no means slow: indeed, considering the complexity of its hypothesis space, it is blazingly fast for a learner of this sort—Yang & Piantadosi’s software

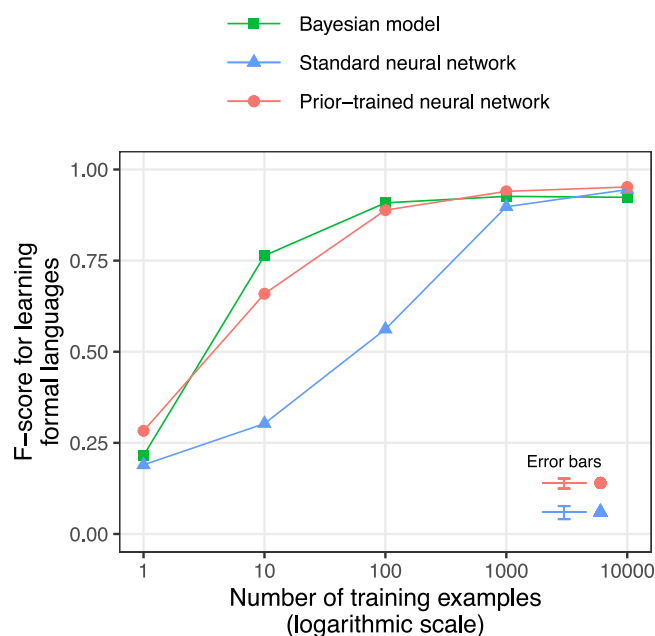


Fig. 3 | Assessing the ability of our model to learn formal languages. This plot averages over the 56 formal languages used by Yang and Piantadosi²³; see Supplementary Note 2 for results for individual languages. The Bayesian model results are taken from Yang and Piantadosi. The prior-trained neural network is our model that has undergone inductive bias distillation; the standard neural network has the same architecture but has not undergone distillation. For each neural network condition, the plot shows the mean over 40 re-runs, with the bottom right showing error bars (giving the full range) averaged over the five training set sizes.

package is appropriately named Fleet. Nonetheless, the flexible parallel processing performed by neural networks enables a substantial speedup over even a very fast Bayesian learner. See Supplementary Methods for more details about these time comparisons.

Learning natural language

We next evaluate our model on its ability to learn natural language from an 8.6-million word corpus of English text⁷⁶. This corpus, which is drawn from the CHILDES database⁷⁷, is composed of sentences that English-speaking parents spoke to their children. It is thus the type of linguistic input that humans receive when acquiring the grammatical structure of English. Due to the size of this dataset and the complexity of natural language, Yang & Piantadosi’s Bayesian learner cannot tractably be applied in this setting. However, the fact that our model has greater time efficiency makes processing this dataset feasible, since neural networks are well-suited for handling large naturalistic datasets, as evidenced by the success of recent large language models such as ChatGPT²⁷.

We evaluate performance on this corpus by computing perplexity on a withheld test set. Perplexity is the standard metric for evaluating next-word prediction: the lower the perplexity is, the better the model is doing at predicting the next word given its context. A perplexity value is difficult to interpret in absolute terms, so to contextualize our model’s performance we use the strong baseline of a smoothed 5-gram model (the best known non-neural system for performing next-word prediction). On this dataset, as reported in ref. 76, a smoothed 5-gram model’s perplexity is 24.4.

Our prior-trained neural network achieves a perplexity of 19.66, substantially outperforming the 5-gram baseline. As shown in Fig. 4a, this perplexity of 19.66 slightly improves on the perplexity of 19.75 achieved by our standard neural network (two-sided t -statistic (77.4 degrees of freedom) = 13.87, $p < 0.001$, Cohen’s $d = 3.10$, 95% confidence interval for the difference of means = [0.073, 0.097]), as well as

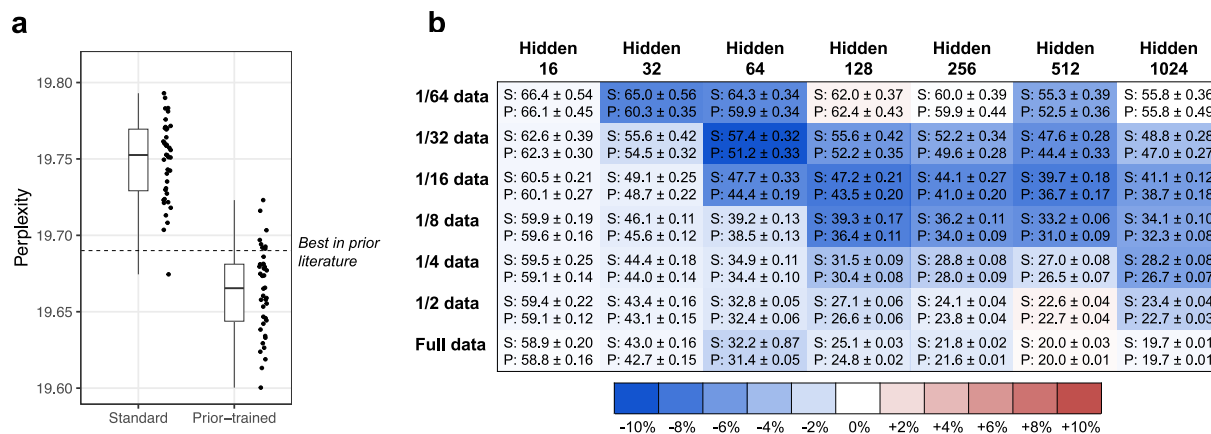


Fig. 4 | Perplexity of neural networks trained on English text. For perplexity, lower is better. **a** Results for our largest model size (1024 hidden units) trained on the full dataset. The boxplots show summary statistics (center = median, bounds of box = first and third quartiles, whiskers = minimum and maximum) across 40 runs, while the dots show the values for the individual re-runs. The dotted line is the best model from prior literature⁷⁶, which was a Transformer. **b** Effect of varying model

size and amount of training data. Each cell shows the mean perplexity and a 95% confidence interval for a standard model (S) and prior-trained model (P), across 20 runs for each model type in each cell. The shading shows the proportion by which inductive bias distillation changes the perplexity (a negative value—i.e., blue—indicates an improvement).

the perplexity of 19.69 achieved by the best-performing neural network from prior literature⁷⁶. These results show that, despite its strong inductive biases, our model retains the flexibility needed to learn well from a naturalistic dataset.

Do our model's strong inductive biases have any human-interpretable effect on how it learns natural language? The previous paragraph might make it seem like the answer is no, since the perplexity of the prior-trained network is only slightly better than the perplexity of the standard network. However, even if the distilled inductive biases had a substantial effect on learning, the evaluation in the previous paragraph would be unlikely to illustrate it. Inductive biases are what guide a learner when the training set is lacking. In the evaluation described so far, the test set was drawn from the same distribution as the training set, and the training set was large (8.6 million words). Therefore, the training data may already give strong signals regarding how to handle the test set, leaving little room for inductive biases to affect the results. To better pinpoint the effects of inductive biases, we should evaluate models on situations where the training set is not as informative. The rest of this section discusses two such situations: when the learner has access to less training data or when the learner must perform out-of-distribution generalization (generalizing to examples drawn from a different distribution than the training set).

Limiting the amount of training data

To test whether the distilled inductive biases have a clearer effect when the amount of CHILDES training data is smaller, we trained models on varying proportions of the dataset—from one sixty-fourth of the dataset to the full dataset. In neural networks, data quantity can interact with model size to determine a model's performance: it is generally true that models with more parameters generalize better, but smaller models sometimes perform better when there is too little training data for a large model to learn effective values for all of its parameters. Therefore, we also varied the number of parameters by varying the hidden layer size (the size of the network's internal vector representations).

The results (Fig. 4b) show that, in many cases, inductive bias distillation substantially improves the perplexity of models trained on English, whereas it never substantially worsens the performance. The full pattern of results is complex, displaying a rough diagonal band along which inductive bias distillation provides the greatest benefit: it helps most in small models with small amounts of data or in large

models with large amounts of data. See Supplementary Discussion for discussion of this pattern.

Testing out-of-distribution generalization

One remarkable aspect of human language acquisition is that we learn rules for which our experience gives little or no direct evidence. Consider the following sentences. In English, a declarative sentence, such as (1a), can be converted to a question by replacing one of the phrases in the sentence (*the banker*) with *who* and moving it to the start of the sentence, as in (1b). There are exceptions to this general rule⁷⁸: it is ungrammatical to form a question of this sort when *who* corresponds to a word inside a conjunction, as is the case in (2b). Though situations that would give rise to (2b) are rare in standard conversation, English speakers reliably learn this constraint.

- (1) a. The judge and the spy will visit the banker.
b. Who will the judge and the spy visit?
- (2) a. The judge will visit the spy and the banker.
b. *Who will the judge visit the spy and?

The evaluation set that we have used so far is a sample of naturally-occurring text. Therefore, for many linguistic phenomena, this evaluation set likely contains few sentences for which capturing that phenomenon is important. As a consequence, a model's performance on this evaluation set does not tell us whether the model has learned the phenomena that linguists typically focus on.

To test whether models have learned particular linguistic phenomena, prior work^{79,80} has proposed an evaluation paradigm based on minimal pairs—pairs of sentences that highlight the rule being investigated. For example, if a learner recognizes that sentence (1b) is better-formed than (2b), that is evidence that the learner has learned the constraint on questions discussed above. The neural networks that we consider in this work are next-word prediction models, which assign a probability to every possible sequence of words. Therefore, we can apply minimal pair evaluations to our models by seeing which sentence in the pair is assigned a higher probability by the model. We use four datasets of minimal pairs, described in “Methods”. Each dataset targets a number of linguistic phenomena, such as the question constraint described above. For this analysis, we return to the setting where both the standard and prior-trained networks achieved the best perplexity (namely, training on the full dataset with a hidden layer size of 1024).

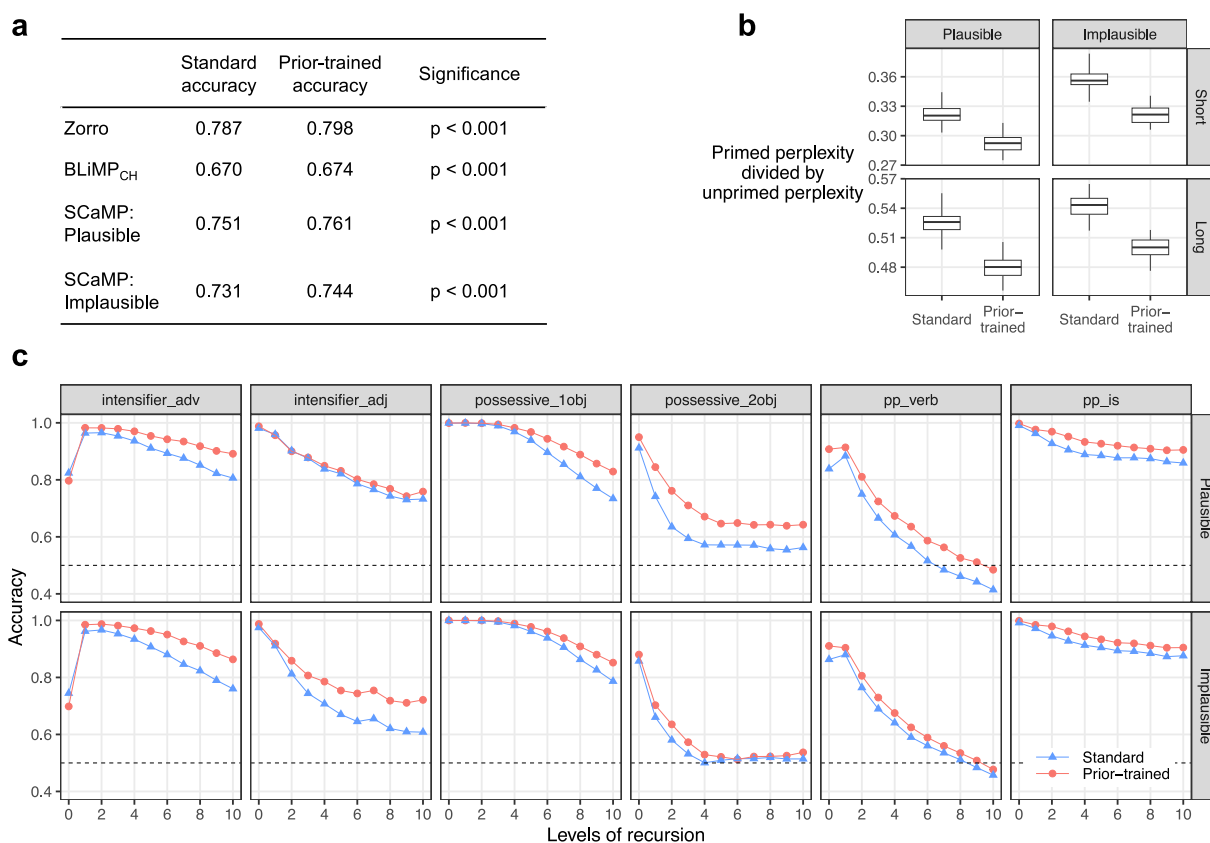


Fig. 5 | Results on targeted linguistic evaluations. **a** Accuracy on four minimal pair datasets that each cover a broad range of syntactic phenomena, averaging across 40 re-runs. The p -values are based on two-sided two-sample t -tests; see “Methods” for details. **b** The extent to which models display priming in sentences that are either short or long and either semantically plausible or semantically implausible. The lower the value on the y-axis is, the more extensively priming has

occurred. The boxplots show summary statistics (center = median, bounds of box = first and third quartiles, whiskers = minimum and maximum) across 40 re-runs. **c** Evaluations of recursion averaged across 40 re-runs; data are presented as mean values. The two model types score similarly when there are few levels of recursion, but at higher levels the prior-trained model often has a higher accuracy than the standard one.

On all four datasets, the prior-trained neural network achieves a small but statistically significant improvement over the standard network (Fig. 5a). Supplementary Note 4 provides results for the individual phenomena that make up each dataset; in general, there are some phenomena where the prior-trained network substantially outperforms the standard one, but there are other phenomena where the reverse is true, and it is difficult to discern clear patterns governing which phenomena are better handled by which model (with one exception—recursion—that is discussed in the next subsection).

Recursion and priming

The minimal pair results in the previous subsection were somewhat challenging to interpret. This fact is perhaps unsurprising because most of the phenomena tested in those evaluations did not have a clear connection to the inductive biases that we distilled. Therefore, we do not have a clear reason to expect that the distillation would help or hurt on those phenomena. In this section, we now consider two phenomena that connect more clearly to our target bias: recursion and priming.

One of the primitives we used—the *plus* primitive—enables syntactic recursion by allowing units to be repeated unboundedly many times. For instance, *plus*(AB) describes the set of strings containing one or more copies of AB: {AB, ABAB, ABABAB, ...}. We therefore might expect that our distilled inductive bias should improve how models process recursion in English, such as handling multiple intensifiers (*the mountain is very very very tall*) or possessives (*my cousin's friend's sister's neighbor*). (Note that some authors distinguish two types of

repetition—recursion and iteration—based on hypothesizing that different mechanisms are used to produce the relevant sentences^{81,82}. In this work, we discuss only surface strings, not the algorithms used to produce them, and therefore consider both types of repetition together under the heading of recursion).

Two of the minimal pair evaluation sets (SCaMP: Plausible and SCaMP: Implausible) contain stimuli that target recursion, such as the examples below (see Supplementary Note 5 for more examples). Each stimulus contains a pair of sentences that end in the same way (underlined), but where the underlined portion is a valid ending for the sentence in one example (the first example in each pair) but not the other. We compute the probability that each model assigns to the underlined portions; the model is said to be correct if it assigns higher probability to the valid case than the invalid one. Each pair involves some degree of recursion (in the examples below, each level adds an additional prepositional phrase). If a model handles recursion well, its accuracy should not degrade much when more levels of recursion are added.

- (3) 1 level:
 - a. The book on the chair is blue.
 - b. *The book was on the chair is blue.
- (4) 2 levels:
 - a. The book on the chair in the room is blue.
 - b. *The book was on the chair in the room is blue.
- (5) 3 levels:
 - a. The book on the chair in the room by the kitchen is blue.
 - b. *The book was on the chair in the room by the kitchen is blue.

Across most of our twelve recursion evaluations, the prior-trained network handles deep recursion better than the standard network (Fig. 5c), supporting the hypothesis that the distilled inductive bias helps models learn recursion in English. Indeed, the recursion subsets of the SCaMP datasets are largely responsible for why the prior-trained network outperforms the standard network on these datasets in Fig. 5a. When the recursion subsets are excluded, the SCaMP_{plausible} scores become 0.731 for the prior-trained network and 0.733 for the standard network ($p = 0.237$), and the SCaMP_{implausible} scores become 0.718 for the prior-trained network and 0.713 for the standard network ($p < 0.001$); see “Methods” for more details about these statistics.

The other primitive that we consider here is our synchrony primitive, which enables multiple parts of a sequence to be synchronized. Most relevantly for our analysis, this primitive can capture a formal language where each sequence contains two copies of some string—e.g., ACCDACCDD or BDABDA. English does not have any such patterns in the syntax of individual sentences, but this type of pattern does occur in neighboring pairs of sentences: in our corpus, 2.8% of sentences are identical to the sentence before them. (Recall that our corpus contains sentences spoken by parents to their children; apparently, parents commonly repeat sentences). For instance, the first 6 sentences in the corpus are:

- (6) you had a what ?
 the basketball had eyes ?
 the basketball had eyes ?
 eyes ?
 you're very excited.
 you're very excited.

Such tendencies are not just statistical properties of corpora; they are also leveraged by language users during sentence processing, as evidenced by priming—the tendency for language users to produce^{83,84} and expect^{85,86} sentences that are similar to others they have recently encountered. Like humans, neural network language models also display priming effects^{87–89}.

Because our synchrony primitive facilitates the parallelism that underlies priming, we hypothesize that our distilled inductive bias should increase the extent to which models display priming. To test this hypothesis, we compute the perplexity that models assign to sentences (underlined) in an unprimed setting where the sentence appears in isolation, as in (7a), and a primed setting where the sentence is preceded by a copy of itself, as in (7b). The more a model undergoes priming, the more its perplexity should decrease from the unprimed setting to the primed setting. This analysis was created to test our hypothesis about priming and is not part of any of the minimal pair datasets in Fig. 5a.

- (7) a. The lady helped the boy in the bank.
 b. The lady helped the boy in the bank. The lady helped the boy in the bank.

We find that, across all four conditions we studied, the prior-trained neural network displays a greater degree of priming than the standard network (Fig. 5b). This result supports our hypothesis that the inductive bias we have distilled predisposes models toward being primed.

Analyzing the distilled inductive bias

Our goal for inductive bias distillation is to give a neural network inductive biases that match those of a target Bayesian model. Our experiments so far have shown that the distillation process indeed imparts useful inductive biases, but the possibility remains that these biases are not the intended ones—they may be useful without matching the Bayesian model. To investigate this possibility, we ran additional experiments in which we varied the target bias to see if the prior-trained network's behavior varied accordingly. We considered three different target biases. First, the all primitives case is the one we have

used throughout the paper, in which neural networks are meta-trained on formal languages defined with a set of primitives that includes recursion and synchrony. The other two cases are based on modified versions that remove one primitive: the no recursion setting uses all primitives except recursion, and the no synchrony setting uses all primitives except synchrony.

When we evaluated these three types of prior-trained networks on learning formal languages, the results varied in ways that paralleled the differences in the distributions they were meta-trained on (Fig. 6a). First, we evaluated these systems on a set of 8 formal languages that require recursion but not synchrony; these 8 languages were a subset of the 56 formal languages evaluated on above. The no recursion case performed much worse than the all primitives and no synchrony cases, as shown by the fact that it needed a greater number of training examples to attain a strong F-score. We then evaluated these systems on a set of 8 formal languages that require both recursion and synchrony. Now, no recursion and no synchrony performed similarly to each other and substantially worse than all primitives. (Note that we also considered evaluating on languages that require synchrony but not recursion, but there were no such languages in the set from which we drew evaluations, and practical challenges prevented the expansion of this set; see Supplementary Methods). These results support the conclusion that inductive bias distillation has imparted the target bias because removing a primitive from the target bias results in a prior-trained system that performs worse on languages featuring that primitive.

We then evaluated these modified prior-trained systems when they were applied to natural language by repeating the natural-language recursion and priming evaluations from above. On the recursion evaluations, as expected, the no recursion case performs worse on average than the all primitives case (Fig. 6b), though note that there are some individual recursion evaluations on which no recursion outperforms all primitives (see Supplement Fig. S1). Unexpectedly, the no synchrony case also performs worse than the all primitives case, showing that the synchrony primitive is helpful on these recursion evaluations; this might be because the recursion evaluations not only involve recursion but also involve long-distance relationships between phrases (e.g., in the sentences in example (5) above, the relationship between the phrase *the book* and *is blue*), and synchrony may help with such long-distance relationships because synchrony creates the opportunity for elements in widely-separated parts of a sequence to depend on each other.

On the priming evaluation, we found that all three prior-trained networks performed similarly to each other and better than the standard network (Fig. 6c). This result suggests that the increased priming observed in the prior-trained systems is not due to the synchrony primitive as we had hypothesized above but rather arises from some other aspect of the prior-training distribution, such as (for example) a general predisposition for discrete, symbolic patterns.

In sum, when we evaluated prior-trained models on formal languages, the nature of the target bias modulated performance in exactly the ways we would expect. When we evaluated on natural language, the results were less clear: the recursion results largely matched our expectations, but the priming results did not. Note that our target bias was defined over formal languages, meaning that natural language is far outside the distribution used during the meta-training phase. We believe these results are consistent with the following conclusion: Inductive bias distillation robustly imparts the target bias within the distribution used during the meta-training process (in our case, the distribution over formal languages), but the effects of that target bias are less predictable when it is applied outside of the meta-training distribution (such as when we evaluated systems on natural language)—a conclusion consistent with prior work finding that neural networks perform in consistent ways within their training distributions but are less predictable when generalizing out-of-distribution^{90,91}.

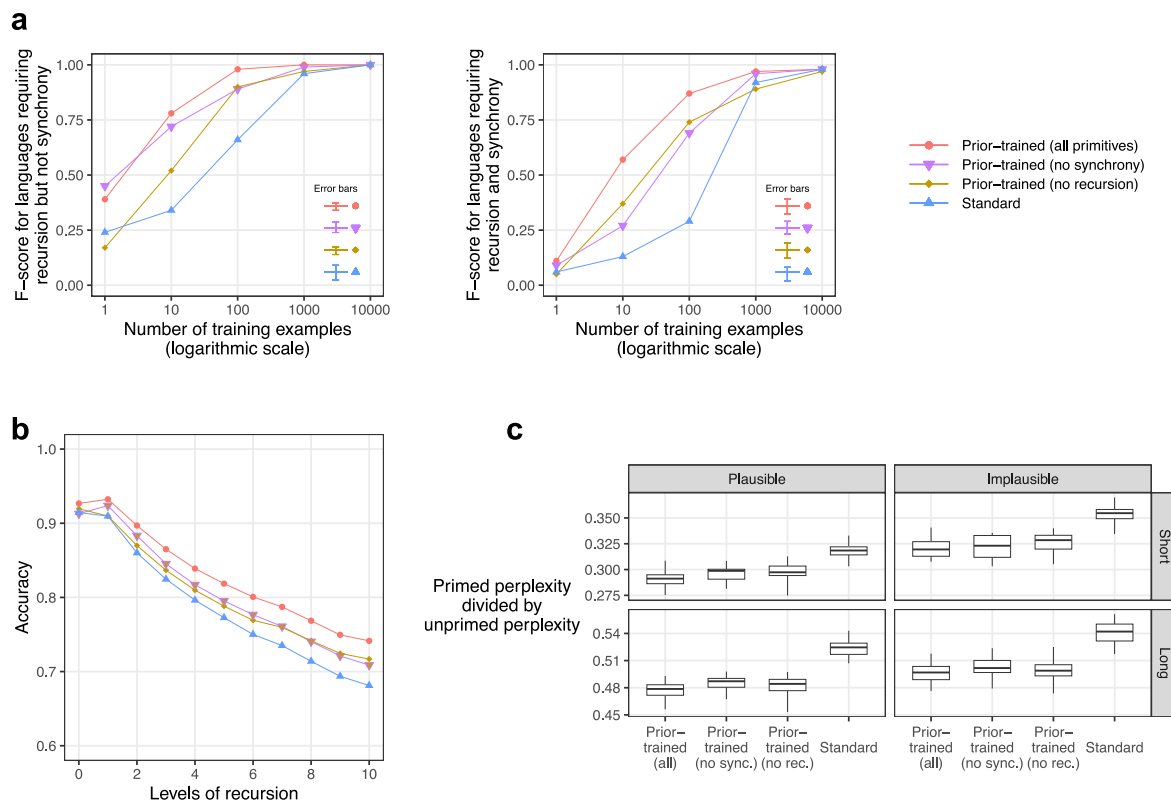


Fig. 6 | Results from varying the set of primitives used during inductive bias distillation. We tested the use of all primitives, all primitives except synchrony (no synchrony), and all primitives except recursion (no recursion). **a** Performance on learning formal languages that require recursion but not synchrony (left) or both recursion and synchrony (right). Each point shows the mean over 20 re-runs, with error bars in the bottom right corner showing the full range averaged over the five training set sizes. **b** Performance on the natural-language recursion evaluation set,

averaged over 20 re-runs and over all sub-conditions of this dataset (see Supplementary Fig. S1 for individual conditions). Data are presented as mean values. **c** Performance on the natural-language priming evaluation set, expressed using boxplots over 20 re-runs (center = median, bounds of box = first and third quartiles, whiskers = minimum and maximum). The lower the value on the y-axis is, the more extensively priming has occurred. no rec. = no recursion; no sync. = no synchrony.

Discussion

We have shown that prior-trained neural networks (created by distilling Bayesian priors into a neural network) can learn effectively from few examples or from complex naturalistic data. Standard Bayesian models and standard neural networks are effective in only one of these settings but not both. Our results illustrate both the possibility and the importance of disentangling strength of inductive bias from strength of representational commitments: our models have strong inductive biases instantiated in continuous vector representations, a combination that enables them—like humans—to learn both rapidly and flexibly.

Inductive bias distillation provides a way to bridge different levels at which cognition can be analyzed. Marr⁹² proposed that cognitive science consider three levels of analysis: the computational level, which provides an abstract characterization of the problem that the mind solves and the solution that it uses; the algorithmic level, which describes the algorithm that the mind uses to execute this solution; and the implementation level, which describes how this algorithm is implemented. Bayesian models are usually meant as proposals at the computational level, characterizing the inductive biases that people have (i.e., which hypotheses do people select given which data?) but remaining agnostic about how these inductive biases are realized^{93–95}. Neural networks instead align more with the algorithmic level (and, in some cases, the implementation level). Therefore, our experiments give an illustration of how inductive bias distillation can connect inductive biases proposed at the computational level to models proposed at the algorithmic level.

In our case study, the work of Yang and Piantadosi²³ provided a natural inspiration for the inductive bias that we distilled. In the more general case, how should we identify appropriate biases to transfer to neural networks? One valuable source of inductive biases is Bayesian models of cognition, which capture aspects of human learning by explicitly defining a prior distribution that captures human inductive biases¹⁷. Tasks for meta-learning can be sampled from these priors, providing a simple route for extracting human inductive biases and transferring them into machines. Binz et al.⁹⁶ recently pointed out that meta-learning can be used to adapt neural networks to their environments, providing a way to extend rational models of cognition to more complex settings. Inductive bias distillation provides a complementary strategy for achieving this goal, in which we define an inductive bias through a prior distribution and then create an approximation to a rational model by distilling that prior into a neural network.

There are several other modeling approaches that are related to inductive bias distillation. We briefly touch on these approaches here; see Supplementary Discussion for detailed discussion. First, prior-training is superficially similar to the popular existing approach of pre-training, in which a network is trained on a large quantity of general data before being further trained on a specific task^{97,98}. Pre-training does influence a model's inductive biases^{99–102}, but we found that pre-training did not work well in our setting; see Supplementary Note 1. Some large pre-trained models, such as ChatGPT, might be able to perform well on our evaluations, but these systems are poorly suited to being models of language learning because they are pre-trained on unrealistically large quantities of natural language. Second, Prior-Data Fitted Networks (PFNs; refs. 103–106) are a type of neural network

trained to approximate Bayesian inference; however, PFNs differ from our approach because they are based on learning, rather than meta-learning, and because they have not been applied to sequential, symbolic domains such as language. In concurrently-developed work, Lake and Baroni⁶³ and Zhou et al.⁶⁴ also used meta-learning as a way to incorporate inductive biases from probabilistic models into neural networks. Our work differs from these approaches in the type of meta-learning that we use (gradient-based, rather than memory-based, meta-learning), in the domain we study (language rather than instructions or visual concepts), and in the fact that we provide a general recipe for using meta-learning to distill the inductive biases of probabilistic models into neural networks; Lake and Baroni and Zhou et al. show how meta-learning with specific task distributions can result in specific inductive biases but do not provide this general framing. Finally, the approaches known as Bayesian neural networks and Bayesian deep learning^{107–110} may sound related to inductive bias distillation, but they in fact have a different goal—namely, enhancing neural networks with explicit estimates of uncertainty over model parameters.

Using inductive bias distillation, we have shown that it is possible to combine the representations of a neural network with the inductive biases of a Bayesian model. Like a Bayesian model, the resulting system can learn formal linguistic patterns from a small number of examples. Like a neural network, it can also learn with much greater time efficiency than standard Bayesian approaches, enabling us to study our target inductive bias in a setting that is larger-scale than previously possible (namely, learning aspects of English syntax from millions of words of natural language). We hope that bridging the divide between these modeling approaches will enable us to account for both the rapidity and the flexibility of human learning.

Methods

Formal language primitives

Our distribution over formal languages is mainly defined using standard regular expression primitives⁴²:

- Atomic alphabet symbols (A, B, \dots)
- Σ : any symbol in the alphabet
- ϵ : the empty string
- `concat`: concatenation
- `or`: randomly selecting one of two strings
- `plus`: Kleene plus, which produces one or more instances of an expression

To overcome formal limitations in the expressive power of regular expressions³⁴, we make two enhancements to the basic regular expression primitives. First, the standard Kleene plus primitive enables tail recursion, in which multiple instances of an expression are joined sequentially (e.g., repeating AB to give $ABAB$). However, it does not enable nested recursion (also known as center embedding), in which multiple instances of an expression are nested inside each other (e.g., nesting AB inside AB to yield $AABB$). We generalize the Kleene plus by incorporating an index argument that specifies where recursed material is inserted: `plus($AB, 0, 0.5$)` inserts new copies of AB at index 0 (the start of the string), yielding tail recursion: $\{AB, ABAB, ABABAB, \dots\}$. The expression `plus($AB, 1, 0.5$)` instead creates nested recursion by inserting new copies of AB in between the existing A and B : $\{AB, AABB, AAABBB, \dots\}$. The final argument in this expression is the probability of continuing to insert new copies of AB : setting this value to 0.5 means that, in this language, the string AB has probability 0.5, the string $AABB$ has probability $0.5 \times 0.5 = 0.25$, etc.

The second enhancement that we make to our set of primitives is the addition of a synchrony mechanism—inspired by synchronous grammars^{111–113}—which allows different parts of a sequence to be

synchronized. For example, the following defines a language in which each sequence has three parts:

```
Synchrony pattern: 0, 1, 0
0: plus(or(A/B, B/D), 0, 0.5)
1: concat(C, C)
```

The synchrony pattern shows that the first and third parts are synchronized (with ID 0), while the middle part is independent (ID 1). The middle part is always the string CC . The first and third parts are sequences made of A, B , and D , where everywhere that there is an A in the first part, there is a B in the third part, and everywhere there is a B in the first part, there is a D in the third part. Example strings in this language include $ACCB$ and $AABACCBDB$.

With these primitives defined, we can sample a formal language by probabilistically combining the primitives to form a language description, with probabilities chosen in a way motivated by Chi¹¹⁴. See Supplementary Methods for the specific probabilistic model we use for this purpose.

We use a different set of primitives from Yang and Piantadosi because we found that, although their primitives were very effective for what Yang and Piantadosi use them for (choosing between hypotheses), they are not well-suited for inductive bias distillation. Specifically, in inductive bias distillation, the distribution over languages is distilled into a learner by showing samples from this distribution to the learner. In a sample of 10,000 languages from Yang and Piantadosi's prior distribution, we found that most languages were degenerate: 94.4% contained only one unique string, and 98.6% contained no strings with a length greater than 1. Therefore, distilling this distribution into a learner would require an unrealistically large number of samples in order to show sufficiently many examples of non-trivial languages, so we instead chose primitives that result in a greater proportion of non-trivial languages.

We tried running Yang and Piantadosi's code with our primitives, but we found that it performed worse with these primitives than with Yang and Piantadosi's primitives, potentially because our synchrony mechanism makes the hypothesis space difficult for their learner to search through. Therefore, in order to present each approach in the most favorable light possible, the results we present with Yang and Piantadosi's model use their set of primitives; for each language, we used the highest-posterior hypothesis among the four candidates listed in their supplementary materials.

Meta-training

For the meta-learning phase of inductive bias distillation, we used MAML²⁸. In our use of MAML, each episode involves one formal language sampled from our distribution over formal languages. In the basic version of MAML, each episode updates the model following the equation below, where M_i is the model's parameters after i episodes of meta-training, L_i is the i th language sampled from the distribution over languages, M'_i is a temporary model created by training M_i on L_i using standard learning, $\mathcal{L}_{L_i}(M'_i)$ is the loss that M'_i obtains on the test set for L_i , ∇_{M_i} is the gradient with respect to M_i , and γ is a learning rate. Crucially, the loss is computed based on M'_i , but the gradient is computed with respect to M_i , making it a second-order gradient that treats the weight updates performed to create M'_i as part of the computation graph to be backpropagated through.

$$M_{i+1} = M_i - \gamma \nabla_{M_i} \mathcal{L}_{L_i}(M'_i) \quad (1)$$

Note, however, that our use of MAML did not adhere to this basic equation because we adopted three additional optimization techniques that have been found in prior work to enable training to converge more rapidly, namely multi-step loss²⁹, the AdamW optimizer^{115,116}, and a cosine-based learning rate scheduler¹¹⁷. See Supplementary Methods

for a complete definition of the MAML algorithm we used that factors in these optimization techniques.

We used a meta-training set of 25,000 formal languages sampled from our meta-grammar, and a meta-validation set of 500 formal languages. Models went through one epoch on this dataset. For each formal language L_i , the model was trained using stochastic gradient descent with a learning rate of 1.0 on n batches of size 10 sampled from that language, where n was drawn uniformly from [1,20]; the result of this training was M'_i . We used multi-step loss²⁹: after each batch, a meta-loss term was computed based on a 1000-item test set sampled from the formal language L_i . The sum of all these meta-loss terms over all the batches gave the loss on this language $\mathcal{L}_{L_i}(M'_i)$. After all batches for the language had been processed, the model then underwent a meta-update using AdamW^{115,116} with weight decay of 0.1 and a learning rate that began at 0, increased linearly to 0.005 during a warmup period covering the first 5% of training, and then decayed following a single half-cycle (without restarts) of a cosine schedule. The model was evaluated on the meta-validation set after every 100 languages. The final trained model was the checkpoint with the lowest validation loss.

The model was a 2-layer LSTM⁴⁵ with dropout¹¹⁸ of 0.1, weight sharing between input and output word representations¹¹⁹, and a hidden layer size of 1024 (unless otherwise specified). We also tried simply pre-training our model on the same dataset (i.e., combining all 25,000 languages into a single next-word prediction dataset), but we found that this approach performed substantially worse than using MAML; see Supplementary Note 1. We implemented our models in PyTorch version 2.2.1+cu121¹²⁰, with meta-training facilitated by the package `higher` version 0.2.1¹²¹ and some training functions based on code from the Transformers library version 4.38.2¹²².

Formal language evaluation

We evaluated models on the set of 56 formal languages used by Yang and Piantadosi. All of these languages were withheld from the meta-learning phase of inductive bias distillation, ensuring that models had not encountered any of these languages before being evaluated on them. Following Yang and Piantadosi, we measured performance using the F-score defined as follows, where $S_{25}(D)$ is the 25 highest-probability strings in the formal language, $S_{25}(h)$ is the 25 highest-probability strings generated by the model, $S(D)$ is the set of all strings in the formal language, and $S(h)$ is the set of all strings generated by the model:

$$\text{precision} = \frac{|S_{25}(h) \cap S(D)|}{|S_{25}(h)|} \quad (2)$$

$$\text{recall} = \frac{|S_{25}(D) \cap S(h)|}{|S_{25}(D)|} \quad (3)$$

$$F\text{-score} = 2 \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (4)$$

We used F-score as our metric to make it possible to compare the performance of prior-trained networks to the numbers that Yang and Piantadosi report for their Bayesian learner, since F-score is the metric that Yang and Piantadosi use. To produce $S(h)$ and $S_{25}(h)$ from our model, we trained it on the relevant dataset then sampled 1 million sequences from it. In some cases, we reweighted these probabilities using a temperature of 0.5, as a measure for prioritizing the sequences that the model had the highest confidence in, and we also in some cases used nucleus sampling¹²³ to truncate the distribution for each next token to the top 0.99 probability mass as another measure for reducing noise (see Supplementary Methods for details of when these measures were used). These hyperparameters were tuned on a validation set of languages that were not in the 56-language evaluation set.

Natural language data: ethical considerations

Our experiments on natural language involved two datasets—the training corpus from ref. 76 and the Zorro dataset¹²⁴—that were in turn based on the CHILDES database⁷⁷, which contains transcripts of natural conversations between parents and children. Due to the possibility that CHILDES might contain private data, we consulted with a member of the Princeton Institutional Review Board about receiving retrospective ethical approval for using this dataset. He responded that the CHILDES database is freely available online and therefore does not meet the definition of private (i.e., information that has been provided for specific purposes by an individual and that the individual can reasonably expect will not be made public).

Training on natural language

During our meta-training phase, the model only used a vocabulary size of 10, but our English corpus had a vocabulary size of 17,096. Therefore, to apply our model to English, we discarded its initial embedding layer and its final output layer, replacing them with randomly-initialized layers of the appropriate size. Although the optimizer that we used within each episode of meta-training was stochastic gradient descent, we used the AdamW optimizer^{115,116} for all natural-language training (including the natural-language training phase that followed meta-learning in prior-trained networks) because, in preliminary experiments, we found that it outperformed stochastic gradient descent. To select the hyperparameters for training models on this dataset, for each cell in the plot in Fig. 4b, we performed an extensive search over the hyperparameters of learning rate, dropout, and number of epochs. We performed this hyperparameter search separately for the prior-trained network and the standard network (using exactly the same search for each type of network, to ensure fairness), and trained each type of model using the hyperparameters that worked best for it. See Supplementary Methods for the values of these hyperparameters, and see Supplementary Note 3 for further discussion of hyperparameters.

To evaluate models on next-word prediction, we use the perplexity. Perplexity is defined as follows, where W is the sequence of words being used to evaluate the model, and N is the length of W :

$$\text{perplexity}(W) = P(W)^{-\frac{1}{N}} \quad (5)$$

Targeted linguistic evaluations

The Zorro evaluation set was used unmodified from ref. 124. The original BLiMP dataset¹²⁵ included many words not present in our model's vocabulary, so we used the authors' code to regenerate the dataset using only the words in their vocabulary that appeared at least 10 times in the model's training set, resulting in the dataset we have labeled BLiMP_{CH} (short for BLiMP_{CHILDES}).

We also wished to compare our model's performance on sentences that were plausible vs. implausible. In the Zorro dataset, the sentences were deliberately designed to be semantically implausible, whereas the BLiMP sentences tend to be reasonably semantically plausible. However, these datasets differ in many other ways, so they cannot provide a controlled comparison on the dimension of plausibility. Instead, we generated two new datasets that are identical in structure but make different word choices to ensure a greater or lesser degree of plausibility. The result is a new dataset SCaMP (Selectional Category Minimal Pairs), which has a semantically-plausible version and a semantically-implausible version. Our additional evaluations targeting recursion and priming were generated from the same codebase as these two new minimal pair datasets.

Statistics

All statistics were computed using R (version 4.1.3). For the p -values shown in Fig. 5a, we used two approaches: model-level tests and item-

level tests. As described below, both types of tests agreed in yielding $p < 0.001$ in all cases (the result shown in Fig. 5a).

The model-level p -values are based on two-sided two-sample t -tests. For each of the four datasets, we obtained the accuracy on that dataset for each of the 40 re-runs of each model type, resulting in two vectors that each contained 40 accuracy values; those two vectors were then compared with a t -test. The Zorro comparison gives a t -value (77.9 degrees of freedom) = 5.30, $p < 0.001$, Cohen's $d = 1.19$, and 95% confidence interval for the difference in means = [0.007, 0.016]. The BLiMP_{CH} comparison gives a t -value (77.9 degrees of freedom) = 3.62, $p < 0.001$, Cohen's $d = 0.810$, and 95% confidence interval for the difference in means = [0.0016, 0.0054]. The SCaMP: Plausible comparison gives a t -value (73.4 degrees of freedom) = 4.41, $p < 0.001$, Cohen's $d = 0.986$, and 95% confidence interval for the difference in means = [0.006, 0.016]; with the recursion subsets removed, these results changed to t -value (74.4 degrees of freedom) = -1.19, $p = 0.237$, Cohen's $d = -0.267$, and 95% confidence interval for the difference in means = [-0.004, 0.001]. The SCaMP: Implausible comparison gives a t -value (68.2 degrees of freedom) = 5.35, $p < 0.001$, Cohen's $d = 1.20$, and 95% confidence interval for the difference in means = [0.008, 0.019]; with the recursion subsets removed, these results changed to t -value (77.1 degrees of freedom) = 3.79, $p < 0.001$, Cohen's $d = 0.847$, and 95% confidence interval for the difference in means = [0.003, 0.008].

The item-level p -values are based on paired two-sided two-sample t -tests. For each of the four datasets, we obtained the proportion of the 40 re-runs of each model type that got each item in that dataset correct, resulting in two vectors having a length equal to the number of items in the dataset; those two vectors were then compared with a paired t -test. The Zorro comparison gives a t -value (45,999 degrees of freedom) = 17.75, $p < 0.001$, Cohen's $d = 0.037$, and 95% confidence interval for the difference in means = [0.010, 0.013]. The BLiMP_{CH} comparison gives a t -value (68,999 degrees of freedom) = 6.49, $p < 0.001$, Cohen's $d = 0.0092$, and 95% confidence interval for the difference in means = [0.0024, 0.0045]. The SCaMP: Plausible comparison gives a t -value (66,999 degrees of freedom) = 22.72, $p < 0.001$, Cohen's $d = 0.033$, and 95% confidence interval for the difference in means = [0.010, 0.012]; with the recursion subsets removed, these results changed to t -value (48,999 degrees of freedom) = -3.06, $p = 0.002$, Cohen's $d = -0.005$, and 95% confidence interval for the difference in means = [-0.003, -0.001]. The SCaMP: Implausible comparison gives a t -value (66,999 degrees of freedom) = 27.11, $p < 0.001$, Cohen's $d = 0.041$, and 95% confidence interval for the difference in means = [0.013, 0.014]; with the recursion subsets removed, these results changed to t -value (48,999 degrees of freedom) = 8.70, $p < 0.001$, Cohen's $d = 0.014$, and 95% confidence interval for the difference in means = [0.004, 0.006].

Reporting summary

Further information on research design is available in the Nature Portfolio Reporting Summary linked to this article.

Data availability

All of the data created in this study have been deposited in GitHub: <https://github.com/tommccoy1/inductive-bias-distillation>; DOI URL: <https://doi.org/10.5281/zenodo.15161071>¹²⁶. All pre-existing datasets that we used are also publicly available online. The 56 formal language datasets²³ that we used for formal language evaluations are available at <https://github.com/piantado/Fleet>. The natural-language training dataset that we used⁷⁶ is available at <https://github.com/adityayedetore/lm-povstim-with-children/tree/master/data/CHILDES>. The Zorro evaluation dataset¹²⁴ is available at <https://github.com/phueb/Zorro/tree/master/sentences/babyberta>. Please see our GitHub repository (<https://github.com/tommccoy1/inductive-bias-distillation>) for detailed instructions on how to download all relevant data from these sources. Source data are provided with this paper.

Code availability

All of our code is publicly available on GitHub: <https://github.com/tommccoy1/inductive-bias-distillation>; DOI URL: <https://doi.org/10.5281/zenodo.15161071>¹²⁶.

References

- Carey, S. & Bartlett, E. Acquiring a single new word. *Pap. Rep. Child Lang. Dev.* **15**, 17–29 (1978).
- Bloom, P. *How Children Learn the Meanings of Words* (MIT Press, 2002).
- Xu, F. & Tenenbaum, J. B. Word learning as Bayesian inference. *Psychol. Rev.* **114**, 245 (2007).
- Reber, A. S. Implicit learning of artificial grammars. *J. Verbal Learn. Verbal Behav.* **6**, 855–863 (1967).
- Morgan, J. L. & Newport, E. L. The role of constituent structure in the induction of an artificial language. *J. Verbal Learn. Verbal Behav.* **20**, 67–85 (1981).
- Culbertson, J., Smolensky, P. & Legendre, G. Learning biases predict a word order universal. *Cognition* **122**, 306–329 (2012).
- Reeder, P. A., Newport, E. L. & Aslin, R. N. Distributional learning of subcategories in an artificial grammar: category generalization and subcategory restrictions. *J. Mem. Lang.* **97**, 17–29 (2017).
- Wilson, C. Learning phonology with substantive bias: an experimental and computational study of velar palatalization. *Cogn. Sci.* **30**, 945–982 (2006).
- Finley, S. & Badecker, W. Artificial language learning and feature-based generalization. *J. Mem. Lang.* **61**, 423–437 (2009).
- Moreton, E. & Pater, J. Structure and substance in artificial-phonology learning, part I: Structure. *Lang. Linguist. Compass* **6**, 686–701 (2012).
- Newport, E. L. & Aslin, R. N. Learning at a distance I. Statistical learning of non-adjacent dependencies. *Cogn. Psychol.* **48**, 127–162 (2004).
- Goldin-Meadow, S. *The Resilience of Language: What Gesture Creation in Deaf Children Can Tell Us About How All Children Learn Language* (Psychology Press, 2003).
- Pearl, L. Poverty of the stimulus without tears. *Lang. Learn. Dev.* **18**, 415–454 (2022).
- Clark, A. & Lappin, S. *Linguistic Nativism and the Poverty of the Stimulus* (John Wiley & Sons, 2010).
- Chomsky, N. *Knowledge of Language: Its Nature, Origin, and Use* (Praeger, 1986).
- Baker, C. L. & McCarthy, J. J. (eds.) *The Logical Problem of Language Acquisition* (MIT Press, 1981).
- Griffiths, T. L., Chater, N., Kemp, C., Perfors, A. & Tenenbaum, J. B. Probabilistic models of cognition: Exploring representations and inductive biases. *Trends Cogn. Sci.* **14**, 357–364 (2010).
- Tenenbaum, J. B., Kemp, C., Griffiths, T. L. & Goodman, N. D. How to grow a mind: Statistics, structure, and abstraction. *Science* **331**, 1279–1285 (2011).
- Perfors, A., Tenenbaum, J., Gibson, E. & Regier, T. In *Recursion and Human Language* 159–175 (2010).
- Perfors, A., Tenenbaum, J. B. & Regier, T. The learnability of abstract syntactic principles. *Cognition* **118**, 306–338 (2011).
- O'Donnell, T. J. *Productivity and Reuse in Language: A Theory of Linguistic Computation and Storage* (MIT Press, 2015).
- Mitchell, T. M. *Machine Learning* (McGraw Hill, 1997).
- Yang, Y. & Piantadosi, S. T. One model for the learning of language. *Proc. Natl Acad. Sci. USA* **119**, e2021865119 (2022).
- McClelland, J. L. et al. Letting structure emerge: connectionist and dynamical systems approaches to cognition. *Trends Cogn. Sci.* **14**, 348–356 (2010).
- Rumelhart, D. E., Hinton, G. E. & McClelland, J. L. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* Vol. 1, 26 (1986).

26. Churchland, P. S. & Sejnowski, T. J. *The Computational Brain* (MIT Press, 1992).
27. OpenAI. GPT-4 technical report. Preprint at <https://arxiv.org/abs/2303.08774> (2023).
28. Finn, C., Abbeel, P. & Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the International Conference on Machine Learning* (2017).
29. Antoniou, A., Edwards, H. & Storkey, A. How to train your MAML. In *International Conference on Learning Representations* (2019).
30. Schmidhuber, J. *Evolutionary Principles in Self-referential Learning*. Ph.D. thesis, Institut für Informatik, Technische Universität München (1987).
31. Thrun, S. & Pratt, L. *Learning to Learn* (Kluwer Academic Publishers, 2012).
32. Wang, D. & Eisner, J. The Galactic Dependencies Treebanks: getting more data by synthesizing new languages. *Trans. Assoc. Comput. Linguist.* **4**, 491–505 (2016).
33. McCoy, R. T., Grant, E., Smolensky, P., Griffiths, T. L. & Linzen, T. Universal linguistic inductive biases via meta-learning. In *Proceedings of the 42nd Annual Conference of the Cognitive Science Society* 737–743 (2020).
34. Chomsky, N. Three models for the description of language. *IRE Trans. Inf. Theory* **2**, 113–124 (1956).
35. Shieber, S. M. *An Introduction to Unification-Based Approaches to Grammar* (CSLI, 2003).
36. Joshi, A. K. & Schabes, Y. In *Handbook of Formal Languages: Volume 3 Beyond Words* 69–123 (1997).
37. Steedman, M. & Baldridge, J. In *Non-Transformational Syntax: Formal and Explicit Models of Grammar* 181–224 (2011).
38. Shieber, S. M. Evidence against the context-freeness of natural language. *Linguist. Philos.* **8**, 333–343 (1985).
39. Frank, R. & Hunter, T. Variation in mild context-sensitivity: derivational state and structural monotonicity. *Evolut. Linguist. Theory* **3**, 181–214 (2021).
40. Frank, R. & Satta, G. Optimality Theory and the generative complexity of constraint violability. *Comput. Linguist.* **24**, 307–315 (1998).
41. Rogers, J. et al. In *Formal Grammar*, 90–108 (Springer, 2013).
42. Partee, B. H., ter Meulen, A. & Wall, R. *Mathematical Methods in Linguistics* (Kluwer, 1993).
43. Chomsky, N. *Syntactic Structures* (Mouton de Gruyter, 1957).
44. Abnar, S., Dehghani, M. & Zuidema, W. Transferring inductive biases through knowledge distillation. Preprint at <https://arxiv.org/abs/2006.00555> (2020).
45. Hochreiter, S. & Schmidhuber, J. Long short-term memory. *Neural Comput.* **9**, 1735–1780 (1997).
46. Merrill, W. et al. A formal hierarchy of RNN architectures. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 443–459 (2020).
47. Bahdanau, D., Cho, K. & Bengio, Y. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations* (2015).
48. Kipewasser, E. & Goldberg, Y. Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Trans. Assoc. Comput. Linguist.* **4**, 313–327 (2016).
49. Peters, M. E. et al. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2227–2237 (2018).
50. Vaswani, A. et al. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30**, 5998–6008 (2017).
51. Deletang, G. et al. Neural networks and the Chomsky hierarchy. In *International Conference on Learning Representations* (2023).
52. Elman, J. L. Distributed representations, simple recurrent networks, and grammatical structure. *Mach. Learn.* **7**, 195–225 (1991).
53. Gulordava, K., Bojanowski, P., Grave, E., Linzen, T. & Baroni, M. Colorless green recurrent networks dream hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 1195–1205 (2018).
54. Hu, J., Gauthier, J., Qian, P., Wilcox, E. & Levy, R. A systematic assessment of syntactic generalization in neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1725–1744 (2020).
55. Wilcox, E. G., Futrell, R. & Levy, R. Using computational models to test syntactic learnability. *Linguist. Inq.* **55**, 805–848 (2022).
56. Mahowald, K. et al. Dissociating language and thought in large language models. *Trends Cogn. Sci.* **28**, 517–540 (2024).
57. Smith, N. J. & Levy, R. The effect of word predictability on reading time is logarithmic. *Cognition* **128**, 302–319 (2013).
58. Schrimpf, M. et al. The neural architecture of language: integrative modeling converges on predictive processing. *Proc. Natl Acad. Sci. USA* **118**, e2105646118 (2021).
59. Lake, B. M. Compositional generalization through meta sequence-to-sequence learning. In *Advances in Neural Information Processing Systems Vol. 32* (2019).
60. Nye, M. I., Solar-Lezama, A., Tenenbaum, J. B. & Lake, B. M. Learning compositional rules via neural program synthesis. In *Advances in Neural Information Processing Systems Vol. 33* 10832–10842 (2020).
61. Conklin, H., Wang, B., Smith, K. & Titov, I. Meta-learning to compositionally generalize. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Vol. 1*, 3322–3335 (2021).
62. Murty, S., Hashimoto, T. B. & Manning, C. DRaCa: A general task augmentation strategy for few-shot natural language inference. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1113–1125 (2021).
63. Lake, B. M. & Baroni, M. Human-like systematic generalization through a meta-learning neural network. *Nature* **623**, 115–121 (2023).
64. Zhou, Y., Feinman, R. & Lake, B. M. Compositional diversity in visual concept learning. *Cognition* **244**, 105711 (2024).
65. Grant, E., Finn, C., Levine, S., Darrell, T. & Griffiths, T. Recasting gradient-based meta-learning as hierarchical Bayes. In *International Conference on Learning Representations* (2018).
66. Lake, B. M., Ullman, T. D., Tenenbaum, J. B. & Gershman, S. J. Building machines that learn and think like people. *Behav. Brain Sci.* **40** (2017).
67. Griffiths, T. L. et al. Doing more with less: meta-reasoning and meta-learning in humans and machines. *Curr. Opin. Behav. Sci.* **29**, 24–30 (2019).
68. Wang, J. X. Meta-learning in natural and artificial intelligence. *Curr. Opin. Behav. Sci.* **38**, 90–95 (2021).
69. Hansen, S. S., Lampinen, A., Suri, G. & McClelland, J. L. Building on prior knowledge without building it in. *Behav. Brain Sci.* **40**, e268 (2017).
70. Dehaene, S., Meyniel, F., Wacongne, C., Wang, L. & Pallier, C. The neural representation of sequences: from transition probabilities to algebraic patterns and linguistic trees. *Neuron* **88**, 2–19 (2015).
71. Pater, J. Generative linguistics and neural networks at 60: foundation, friction, and fusion. *Language* **95**, e41–e74 (2019).
72. Papadimitriou, C. H., Vempala, S. S., Mitropolsky, D., Collins, M. & Maass, W. Brain computation by assemblies of neurons. *Proc. Natl Acad. Sci. USA* **117**, 14464–14472 (2020).

73. Dehaene, S., Al Roumi, F., Lakretz, Y., Planton, S. & Sablé-Meyer, M. Symbols and mental programs: a hypothesis about human singularity. *Trends in Cognitive Sciences* (2022).
74. Prickett, B., Traylor, A. & Pater, J. Learning reduplication with a neural network that lacks explicit variables. *J. Lang. Model.* **10**, 1–38 (2022).
75. Smolensky, P. & Legendre, G. Foundational implications of the ICS architecture: unification in cognitive science. *Harmonic Mind* **1**, 99–121 (2006).
76. Yedetore, A., Linzen, T., Frank, R. & McCoy, R. T. How poor is the stimulus? Evaluating hierarchical generalization in neural networks trained on child-directed speech. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 9370–9393 (2023).
77. MacWhinney, B. *The CHILDES Project: Tools for Analyzing Talk* (Lawrence Erlbaum Associates, 2000).
78. Ross, J. R. *Constraints on Variables in Syntax*. Ph.D. thesis, Massachusetts Institute of Technology (1967).
79. Linzen, T., Dupoux, E. & Goldberg, Y. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Trans. Assoc. Comput. Linguist.* **4**, 521–535 (2016).
80. Marvin, R. & Linzen, T. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 1192–1202 (2018).
81. Reich, P. A. The finiteness of natural language. *Language* 831–843 (1969).
82. Christiansen, M. H. The (non) necessity of recursion in natural language processing. In *Proceedings of the 14th Annual Conference of the Cognitive Science Society*, 665–670 (1992).
83. Bock, J. K. Syntactic persistence in language production. *Cogn. Psychol.* **18**, 355–387 (1986).
84. Mahowald, K., James, A., Futrell, R. & Gibson, E. A meta-analysis of syntactic priming in language production. *J. Mem. Lang.* **91**, 5–27 (2016).
85. Arai, M., Van Gompel, R. P. & Scheepers, C. Priming ditransitive structures in comprehension. *Cogn. Psychol.* **54**, 218–250 (2007).
86. Schuster, S. & Degen, J. I know what you're probably going to say: Listener adaptation to variable use of uncertainty expressions. *Cognition* **203**, 104285 (2020).
87. van Schijndel, M. & Linzen, T. A neural model of adaptation in reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 4704–4710 (Association for Computational Linguistics, 2018).
88. Prasad, G., van Schijndel, M. & Linzen, T. Using priming to uncover the organization of syntactic representations in neural language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 66–76 (2019).
89. Sinclair, A., Jumelet, J., Zuidema, W. & Fernández, R. Structural persistence in language models: Priming as a window into abstract language representations. *Trans. Assoc. Comput. Linguist.* **10**, 1031–1050 (2022).
90. Weber, N., Shekhar, L. & Balasubramanian, N. The fine line between linguistic generalization and failure in Seq2Seq-attention models. In *Proceedings of the Workshop on Generalization in the Age of Deep Learning*, 24–27 (2018).
91. McCoy, R. T., Min, J. & Linzen, T. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, 217–227 (2020).
92. Marr, D. *Vision: A Computational Approach* (Freeman, 1982).
93. Griffiths, T. L., Chater, N., Norris, D. & Pouget, A. How the Bayesians got their beliefs (and what those beliefs actually are): Comment on Bowers and Davis. *Psychol. Bull.* **138**, 415–422 (2012).
94. Goodman, N. D. & Frank, M. C. Pragmatic language interpretation as probabilistic inference. *Trends Cogn. Sci.* **20**, 818–829 (2016).
95. Degen, J. The rational speech act framework. *Annu. Rev. Linguist.* **9**, 519–540 (2023).
96. Binz, M. et al. Meta-learned models of cognition. *Behav. Brain Sci.* **47**, e147 (2024).
97. Donahue, J. et al. DeCAF: a deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, 647–655 (PMLR, 2014).
98. McCann, B., Bradbury, J., Xiong, C. & Socher, R. Learned in translation: contextualized word vectors. In *Advances in Neural Information Processing Systems* Vol. 30 (2017).
99. Papadimitriou, I. & Jurafsky, D. Learning music helps you read: using transfer to study linguistic structure in language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6829–6839 (2020).
100. Chan, S. et al. Data distributional properties drive emergent in-context learning in transformers. *Adv. Neural Inf. Process. Syst.* **35**, 18878–18891 (2022).
101. Papadimitriou, I. & Jurafsky, D. Injecting structural hints: using language models to study inductive biases in language learning. *Findings of the Association for Computational Linguistics: EMNLP 2023* 8402–8413 (2023).
102. Lindemann, M., Koller, A. & Titov, I. SIP: injecting a structural inductive bias into a Seq2Seq model by simulation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 6570–6587 (2024).
103. Müller, S., Hollmann, N., Arango, S. P., Grabocka, J. & Hutter, F. Transformers can do Bayesian inference. In *International Conference on Learning Representations* (2022).
104. Hollmann, N., Müller, S., Eggensperger, K. & Hutter, F. TabPFN: A transformer that solves small tabular classification problems in a second. In *The Eleventh International Conference on Learning Representations* (2023).
105. Müller, S., Feurer, M., Hollmann, N. & Hutter, F. PFNs4BO: In-context learning for Bayesian optimization. In *Proceedings of the 40th International Conference on Machine Learning*, vol. 202 of *Proceedings of Machine Learning Research*, 25444–25470 (PMLR, 2023).
106. Nagler, T. Statistical foundations of prior-data fitted networks. In *Proceedings of the 40th International Conference on Machine Learning*, vol. 202 of *Proceedings of Machine Learning Research*, 25660–25676 (PMLR, 2023).
107. MacKay, D. J. A practical Bayesian framework for backpropagation networks. *Neural Comput.* **4**, 448–472 (1992).
108. Wilson, A. G. & Izmailov, P. Bayesian deep learning and a probabilistic perspective of generalization. *Adv. Neural Inf. Process. Syst.* **33**, 4697–4708 (2020).
109. Yoon, J. et al. Bayesian model-agnostic meta-learning. In *Advances in Neural Information Processing Systems* Vol. 31 (2018).
110. Izmailov, P., Vikram, S., Hoffman, M. D. & Wilson, A. G. What are Bayesian neural network posteriors really like? In *International Conference on Machine Learning*, 4629–4640 (PMLR, 2021).
111. Lewis, P. M. & Stearns, R. E. Syntax-directed transduction. *J. ACM* **15**, 465–488 (1968).
112. Aho, A. V. & Ullman, J. D. Syntax directed translations and the pushdown assembler. *J. Comput. Syst. Sci.* **3**, 37–56 (1969).
113. Shieber, S. M. & Schabes, Y. Synchronous tree-adjointing grammars. In *COLING 1990 Volume 3: Papers presented to the 13th International Conference on Computational Linguistics* (1990).
114. Chi, Z. Statistical properties of probabilistic context-free grammars. *Comput. Linguist.* **25**, 131–160 (1999).
115. Kingma, D. & Ba, J. Adam: a method for stochastic optimization. In *International Conference on Learning Representations* (2015).

116. Loshchilov, I. & Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations* (2019).
 117. Loshchilov, I. & Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations* (2017).
 118. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
 119. Press, O. & Wolf, L. Using the output embedding to improve language models. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 157–163 (2017).
 120. Paszke, A. et al. PyTorch: an imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems* Vol. 32 (2019).
 121. Grefenstette, E. et al. Generalized inner loop meta-learning. Preprint at <https://arxiv.org/abs/1910.01727> (2019).
 122. Wolf, T. et al. Transformers: state-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 38–45 (2020).
 123. Holtzman, A., Buys, J., Du, L., Forbes, M. & Choi, Y. The curious case of neural text degeneration. In *International Conference on Learning Representations* (2020).
 124. Huebner, P. A., Sulem, E., Fisher, C. & Roth, D. BabyBERTa: learning more grammar with small-scale child-directed language. In *Proceedings of the 25th Conference on Computational Natural Language Learning*, 624–646 (2021).
 125. Warstadt, A. et al. BLiMP: the benchmark of linguistic minimal pairs for English. *Trans. Assoc. Comput. Linguist.* **8**, 377–392 (2020).
 126. McCoy, R. T. & Griffiths, T. L. Modeling rapid language learning by distilling Bayesian priors into artificial neural networks. GitHub. <https://doi.org/10.5281/zenodo.15161071> (2025).
- the National Science Foundation, the Office of Naval Research, or Princeton Research Computing.

Author contributions

R.T.M. and T.L.G. designed research, contributed new analytic tools, analyzed data, and wrote the paper. R.T.M. performed research.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41467-025-59957-y>.

Correspondence and requests for materials should be addressed to R. Thomas McCoy.

Peer review information *Nature Communications* thanks Roger Levy, Brandon Turner and the other, anonymous, reviewers for their contribution to the peer review of this work. A peer review file is available.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License, which permits any non-commercial use, sharing, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if you modified the licensed material. You do not have permission under this licence to share adapted material derived from this article or parts of it. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by-nc-nd/4.0/>.

© The Author(s) 2025

Acknowledgements

For helpful discussion, we are grateful to Adele Goldberg, Erin Grant, Tal Linzen, Paul Smolensky, and the members of the Princeton Computational Cognitive Science Lab. We thank Steven Piantadosi for assistance in using the Fleet software package. This material is based upon work supported by the National Science Foundation SBE Postdoctoral Research Fellowship under Grant No. 2204152 (R.T.M.) and the Office of Naval Research under grants No. N00014-18-1-2873 and N00014-23-1-2510 (T.L.G.), and the experiments were conducted using computing resources managed by Princeton Research Computing. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of