McCoy, R Thomas

Address		**Email** tom.mccoy@jhu.edu (update 2021/11/26)	
612 Victoria Lane Wexford, PA 15090		**Home Phone** **Office Phone**	
Current Institution	Johns Hopkins University	**Department**	
Location	3400 N Charles St, Baltimore, MD 21218		
Highest Degree		**Institution**	**Date**
Research Interests	**Primary** Natural Language Processing/Computational Linguistics		
Secondary	AI; Machine Learning		
Discipline(s)	Natural Language Processing; Computer Science; Linguistics		
Position(s) applied	FACITHACA (complete)		

1. Tal Linzen, New York University, linzen@nyu.edu (2021/11/26)	file (PDF, PDF, 2021/11/26, tailored)
2. Paul Smolensky, Johns Hopkins University, Microsoft, paul.smolensky@gmail.com (2021/11/26)	file (PDF, PDF, 2021/11/28)
3. Asli Celikyilmaz, Facebook AI Research, aslic@fb.com (2021/11/26)	file (PDF, PDF, 2021/11/28)
4. Jacob Andreas, Massachusetts Institute of Technology, jdandreas@gmail.com (2021/11/26)	file (PDF, PDF, 2021/12/01)
Faculty contacts: Similar interests: Yoav Artzi, Lillian Lee, Alexander Rush	
US Citizen: Yes, **Visa Type:**	
Pub1: Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL). 2019. **Pub2:** RNNs implicitly implement tensor-product representations. R. Thomas McCoy, Tal Linzen, Ewan Dunbar, and Paul Smolensky. International Conference on Learning Representations (ICLR). 2019.	
Understand1: I Understand	

Received Materials	**FACITHACA**	*Cover Letter:* file (PDF, PDF, 2021/11/26) *Curriculum Vitae:* file (PDF, PDF, 2021/11/28) *Research Statement:* file (PDF, PDF, 2021/11/25) *Teaching Statement:* file (PDF, PDF, 2021/11/25) *Statement of Contribution to Diversity, Equity, and Inclusion:* file (PDF, PDF, 2021/11/26) *PDF of Most Significant Paper:* file (PDF, PDF, 2021/11/25) *PDF of Other Most Significant Paper:* file (PDF, PDF, 2021/11/25)

Tom McCoy
Department of Cognitive Science
Johns Hopkins University
3400 N Charles St
Baltimore, MD 21218

Department of Computer Science
402 Gates Hall
Cornell University
Ithaca, NY 14853-7501

Dear members of the search committee,

I am writing to apply for the position of Tenure-Track Professor in Computer Science. I am a PhD candidate in the Department of Cognitive Science at Johns Hopkins University, expecting to defend my dissertation in May 2022. Both as a researcher and as an instructor, I would be excited to contribute to the cross-disciplinary strength of Cornell's AI community.

My research combines natural language processing and cognitive science. I focus on analyzing the linguistic representations and learning behavior of neural network models and people. By investigating these topics, I aim to enable NLP models to achieve the rapid learning and robust generalization that humans display. Over the course of my PhD, I have published 16 papers on these topics, including 8 as first author and 2 as last author (with an undergraduate mentee as the first author). The venues of these papers include ACL, ICLR, TACL, and the CogSci conference, and several of these papers are collaborations with industry researchers at Microsoft, Google, and Facebook. Much of this work was funded by the competitive NSF Graduate Research Fellowship Program (GRFP).

I am excited by the prospect of working at Cornell due to its great strength in natural language processing and other language-related areas. Within the Department of Computer Science, I see potential for collaboration with Dr. Yoav Artzi, Dr. Lillian Lee, and Dr. Alexander Rush. In addition, I would be interested in building cross-departmental collaborations with faculty in Linguistics, such as Dr. Mats Rooth and Dr. Marten van Schijndel.

I also look forward to mentoring students, both as a course instructor and as a research advisor. Across the five courses for which I have been a lab instructor or teaching assistant, 79% of student evaluations (45/57) have given my teaching the top score of *Excellent*, and 100% of scores were *Satisfactory* or above. As a research mentor, I have supervised or co-supervised the research of two Master's students and two undergraduate students, leading to publications in top venues, and I plan to continue this focus on mentorship.

Finally, I am committed to improving the diversity of our field. In my current department, I am actively involved in initiatives aimed at recruiting and retaining a diverse group of PhD students. I am also the national co-program-chair of the North American Computational Linguistics Open competition (NACLO), a contest aimed at introducing a broad range of high school students to computational linguistics. At Cornell, I would continue such PhD-recruitment initiatives and would participate in organizing the Cornell NACLO site.

Please do not hesitate to contact me (tom.mccoy@jhu.edu) if I can provide any other helpful information. Thank you for your time and consideration.

Sincerely,
Tom McCoy

R. Thomas McCoy

Department of Cognitive Science
141 Krieger Hall
Johns Hopkins University
3400 N. Charles Street
Baltimore, MD 21218-2685

Email: tom.mccoy@jhu.edu
Last updated: November 24, 2021

Education

2017–present	Johns Hopkins University: Ph.D. in Cognitive Science. GPA: 4.0. *Advisors:* Tal Linzen, Paul Smolensky
2013–2017	Yale University: B.A. in Linguistics, *summa cum laude*, distinction in the major. GPA: 4.0. *Advisor:* Robert Frank
Summer 2016	Institute on Collaborative Language Research (CoLang), University of Alaska Fairbanks
Summer 2015	Linguistic Summer Institute, University of Chicago

Employment

Summer 2020	Microsoft Research intern Supervisor: Asli Celikyilmaz *Evaluation methods for neural text generation systems*
Summer 2018	Jelinek Summer Workshop on Speech and Language Technology (JSALT) sentence representations team Team leaders: Sam Bowman, Ellie Pavlick *Analysis techniques for learned sentence representations.*
Summers 2013–2017	Summer research projects in natural language processing and linguistics Supervisors: Chris Dyer, Lori Levin, Patrick Littell, Claire Bowern, Ryan Bennett, Jim Wood, Raffaella Zanuttini *Projects: creating finite-state morphological analyzers for Oromo and Kinyarwanda; developing automatic semantic processing techniques for a database of Australian languages; collecting lip rounding measurements from images of Irish speakers; writing a sketch grammar of Kuwarra; editing web pages about regional variation in syntax*

Awards

1. Fellowships

2018–2021 NSF Graduate Research Fellowship
Project title: Assessing the capacity of computational models to make linguistic generalizations

2020 Finalist: Facebook Fellowship
One of four finalists in the Natural Language Processing category; two of the four finalists received fellowships.

2021 Sweitzer Fellow
Fellowship awarded by the Johns Hopkins Department of Cognitive Science to one graduate student.

2017–2020 Owen Scholars Fellowship
Fellowship for outstanding incoming Johns Hopkins PhD students in the natural sciences.

2017 Finalist, Rhodes Scholarship

2017 Finalist, Marshall Scholarship

2. Prizes

2017 Alpheus Henry Snow Prize
Award for the graduating Yale senior who is "adjudged by the faculty to have done the most for Yale by inspiring in his or her classmates an admiration and love for the best traditions of high scholarship."

2016 Hart Lyman Prize
Award for the Yale junior who "has made through his/her own efforts the best record intellectually and socially."

2016 Phi Beta Kappa
One of 13 Yale students admitted as juniors.

2013 World champion team at the International Linguistics Olympiad
Member of the four-person U.S. team selected by the North American Computational Linguistics Olympiad.

2013 United States Presidential Scholar
One of two for Pennsylvania.

3. Grants

2019 NeurIPS Travel Grant
Grant to fund travel to present work at the NeurIPS workshop on Context and Compositionality in Biological and Artificial Neural Systems.

2019 ICLR Travel Grant
Grant to fund travel to present two projects at the 2019 ICLR conference.

2018–2019 Johns Hopkins University Center for Educational Resources Technology Fellowship Grant
Co-Grantee: Tal Linzen
Grant to develop interactive visualizations of concepts in computational cognitive science.

Peer-Reviewed Publications

2022 Paul Smolensky, **R. Thomas McCoy**, Roland Fernandez, Matthew Goldrick, and Jianfeng Gao. Neurocompositional computing: From the central paradox of cognition to a new generation of AI systems. Accepted to *AI Magazine*.

2021 **R. Thomas McCoy**, Jennifer Culbertson, Paul Smolensky, and Géraldine Legendre. Infinite use of finite means? Evaluating the generalization of center embedding learned from an artificial grammar. In *Proceedings of the 43rd Annual Conference of the Cognitive Science Society*.

2021 Paul Soulos, Sudha Rao, Caitlin Smith, Eric Rosen, Asli Celikyilmaz, **R. Thomas McCoy**, Yichen Jiang, Coleman Haley, Roland Fernandez, Hamid Palangi, Jianfeng Gao and Paul Smolensky. Structural Biases for Improving Transformers on Translation into Morphologically Rich Languages. In *Proceedings of the 4th Workshop on Technologies for Machine Translation of Low Resource Languages (LoResMT2021)*.

2020 **R. Thomas McCoy**, Erin Grant, Paul Smolensky, Thomas L. Griffiths, and Tal Linzen. Universal linguistic inductive biases via meta-learning. In *Proceedings of the 42nd Annual Conference of the Cognitive Science Society*.

2020 **R. Thomas McCoy**, Robert Frank, and Tal Linzen. Does syntax need to grow on trees? Sources of hierarchical inductive bias in sequence-to-sequence networks. *Transactions of the Association for Computational Linguistics (TACL)*.

2020 Michael Lepori and **R. Thomas McCoy**. Picking BERT's brain: Analyzing contextualized embeddings using Representational Similarity Analysis. In *Proceedings of the 28th International Conference on Computational Linguistics (COLING)*.

2020 Paul Soulos, **R. Thomas McCoy**, Tal Linzen, and Paul Smolensky. Uncovering the compositional structure of vector representations with Role Learning Networks. In *BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*.

2020 Michael Lepori, Tal Linzen, and **R. Thomas McCoy**. Representations of Syntax [MASK] Useful: Effects of Constituency and Dependency Structure in Recursive LSTMs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL).*

2020 Junghyun Min, **R. Thomas McCoy**, Dipanjan Das, Emily Pitler, and Tal Linzen. Syntactic data augmentation increases robustness to inference heuristics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL).*

2020 **R. Thomas McCoy**, Junghyun Min, and Tal Linzen. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP.*

2019 Najoung Kim, Roma Patel, Adam Poliak, Alex Wang, Patrick Xia, **R. Thomas McCoy**, Ian Tenney, Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, Ellie Pavlick. Probing What Different NLP Tasks Teach Machines about Function Word Comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019).*
Best paper award at *SEM 2019.

2019 **R. Thomas McCoy**, Tal Linzen, Ewan Dunbar, and Paul Smolensky. RNNs implicitly implement tensor-product representations. *International Conference on Learning Representations (ICLR).*

2019 Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, **R. Thomas McCoy**, Najoung Kim, Benjamin Van Durme, Samuel R. Bowman, Dipanjan Das, and Ellie Pavlick. What do you learn from context? Probing for sentence structure in contextualized word representations. *International Conference on Learning Representations (ICLR).*

2019 **R. Thomas McCoy**, Ellie Pavlick, and Tal Linzen. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL).*

2019 **R. Thomas McCoy**. Touch *down* in Pittsburghese. *Yale Working Papers in Grammatical Diversity.*

2019 Samuel R. Bowman, Ellie Pavlick, Edouard Grave, Benjamin Van Durme, Alex Wang, Jan Hula, Patrick Xia, Raghavendra Pappagari, **R. Thomas McCoy**, Roma Patel, Najoung Kim, Ian Tenney, Yinghui Huang, Katherin Yu, Shuning Jin, and Berlin Chen. Can You Tell Me How to Get Past Sesame Street? Sentence-Level Pretraining Beyond Language Modeling. *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL).*

2019 **R. Thomas McCoy** and Tal Linzen. Non-entailed subsequences as a challenge for natural language inference. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019.*

2018 **R. Thomas McCoy**, Robert Frank, and Tal Linzen. Revisiting the poverty of the stimulus: hierarchical generalization without a hierarchical bias in recurrent neural networks. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society.*

2018 Patrick Littell, **R. Thomas McCoy**, Na-Rae Han, Shruti Rijhwani, Zaid Sheikh, David Mortensen, Teruko Mitamura, and Lori Levin. Parser combinators for Tigrinya and Oromo morphology. In *Language Resources and Evaluation Conference (LREC) 2018.*

2018 **R. Thomas McCoy** and Robert Frank. Phonologically Informed Edit Distance Algorithms for Word Alignment with Low-Resource Languages. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2018.*

2017 Jungo Kasai, Robert Frank, **R. Thomas McCoy**, Owen Rambow, and Alexis Nasr. TAG parsing with neural networks and vector representations of supertags. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP).*

2017 Dan Friedman*, Jungo Kasai*, **R. Thomas McCoy***, Robert Frank, Forrest Davis, and Owen Rambow. Linguistically Rich Vector Representations of Supertags for TAG Parsing. In *Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms.*
*Equal contribution.

2017 **R. Thomas McCoy**. English comparatives as degree-phrase relative clauses. In *Proceedings of the Linguistic Society of America 2.*

Work in preparation

R. Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. How much do language models copy from their training data? Evaluating linguistic novelty in text generation using RAVEN. In preparation to submit to *Transactions of the Association for Computational Linguistics.*

Aditya Yedetore, Tal Linzen, Robert Frank, and **R. Thomas McCoy**. How poor is the stimulus? Evaluating hierarchical generalization in neural networks trained on child-directed speech. In preparation to submit to *Proceedings of the Annual Conference of the North American Association for Computational Linguistics.*

R. Thomas McCoy, Tal Linzen, and Paul Smolensky. DISCOVER: A framework for dissecting compositionality in vector representations. In preparation to submit to the *Journal of Artificial Intelligence Research.*

Unpublished conference presentations

2018 R. Thomas McCoy, Robert Frank, and Tal Linzen. Investigating hierarchical bias in the acquisition of English question formation with recurrent neural networks. Poster presentation, *2018 Legrain conference: Learning Language in Humans and in Machines*, Paris, France, July 5-6.

2018 Robert Frank, R. Thomas McCoy, and Tal Linzen. Neural network syntax in the age of deep learning: the case of question formation. Oral presentation, *Society for Computation in Linguistics*, Salt Lake City, Utah, January 5.

2017 Patrick Littell, R. Thomas McCoy, and Lori Levin. The North American Computational Linguistics Olympiad. Oral presentation, in Datablitz: Getting High School Students into Linguistics: Current Activities and Future Directions, *Linguistic Society of America Annual Meeting*, Austin, Texas, January 7.

Invited talks

2021 Montreal Computational and Quantitative Linguistics Lab at McGill (MCQLL). October 26, 2021.
Discovering implicit compositional representations in neural networks

2021 Edinburgh Centre for Language Evolution. September 28, 2021.
How do neural networks represent compositional symbolic structure?

2021 USC ISI Natural Language Seminar. April 15, 2021.
Universal linguistic inductive biases via meta-learning.

2020 DeepMind language reading group. December 7, 2020.
Analyzing the syntactic inductive biases of sequence-to-sequence networks.

2020 Berkeley NLP Seminar. October 16, 2020.
Analyzing the syntactic inductive biases of sequence-to-sequence networks.

2020 NLP With Friends seminar series. August 12, 2020.
Universal linguistic inductive biases via meta-learning.

2019 Workshop on Gradient Symbolic Computation. Johns Hopkins University. September 19, 2019.
Tensor product decomposition of continuous vector representations

2018 Microsoft Research, Redmond. December 11, 2018.
Discovering the compositional structure implicitly learned by neural networks

Teaching

Across the five Johns Hopkins courses for which I have been a lab instructor or teaching assistant, 57 students have completed course evaluations rating my teaching on a 5-point scale. 79% scored my teaching as "Excellent" (the top score), 14% scored my teaching as "Good" (a score of 4/5), and the remaining 7% scored my teaching as "Satisfactory" (a score of 3/5). None have given scores of "Weak" or "Poor."

Spring 2020	Johns Hopkins University Role: Teaching Assistant Course: Foundations of Cognitive Science Lecture Instructor: Paul Smolensky *Led one seminar discussion and graded assignments.*
Fall 2019	Johns Hopkins University Role: Teaching Assistant, Lab Instructor Course: Computational Psycholinguistics Lecture Instructor: Tal Linzen *Led lab sessions and graded assignments.*
Spring 2019	Johns Hopkins University Role: Teaching Assistant Course: Syntax I Lecture Instructor: Géraldine Legendre *Led review sessions and graded assignments.*
Fall 2018	Johns Hopkins University Role: Teaching Assistant Course: Introduction to Computational Cognitive Science Lecture Instructor: Tal Linzen *Created educational simulations, tutorials, and homeworks in Javascript and Jupyter and taught lectures using these resources.*
Spring 2018	Johns Hopkins University Role: Fieldwork Instructor Course: World of Language Lecture Instructor: Géraldine Legendre *Led two sections of weekly fieldwork sessions complementing lectures.*
Summer 2015	Linguistic Society of America Summer Institute Role: Workshop Co-Instructor Course: Linguistic Enigmatography Co-Instructor: Lori Levin *Developed and co-taught a one-week workshop on creating linguistic puzzles.*

Mentoring

Master's students

2019–2020 Junghyun Min
Co-supervised with Tal Linzen.

2019–2020 Paul Soulos
Co-supervised with Paul Smolensky.

Undergraduate students

2019–present Aditya Yedetore
Co-supervised with Tal Linzen.

2019–2020 Michael Lepori
Co-supervised with Tal Linzen.

Outreach and Contributions to Diversity

2020–present Johns Hopkins Cognitive Science Representation and Diversity Committee
Co-created and co-organized a program for giving feedback on PhD applications to prospective students who belong to underrepresented groups. Served as a mentor for 6 prospective students.

2020 Johns Hopkins Cognitive Science syllabus section for raising awareness about research and graduate school.
With one other graduate student, wrote a statement that faculty members added to their syllabi and course discussions describing how to pursue research opportunities and graduate school, in order to raise awareness of these opportunities among a broader group of undergraduates.

2020 Public talk for the National Museum of Language: *Language Squared: The Linguistics of Crosswords.*

2013–present North American Computational Linguistics Olympiad (NACLO).
Contest that introduces high school students to computational linguistics, with 1000 to 1500 students participating each year. Last year, 42% of participants were female, a high proportion for a computational initiative. National level: Co-Program Chair; problem writer (20 problems to date). Local level: Co-founder and co-organizer of the Yale contest site (2013-2017); co-organizer of the Johns Hopkins contest site (2017-present); organizer of pre-contest practice sessions at both sites.

2018–2019 International Linguistics Olympiad (IOL): Problem writer.

2016 Yale Grammatical Diversity Project
Authored two webpages describing regional grammatical phenomena (all the further, subject contact relatives).

2013–2017	Linguistics teaching initiatives *Designed and taught a one-lecture linguistics class to high school students in New Haven in connection with the programs Splash, Sprout, and Math Mornings. Presented 8 times to groups ranging from 25 to 50 students.*

Service

2019–present	Departmental representative for the Department of Cognitive Science in the Johns Hopkins Graduate Representative Organization.
2016–2017	Computational Linguistics at Yale (CLAY) reading group: Co-organizer.
2015–2017	Yale Undergraduate Linguistics Society: Co-founder (2015), president (2015–2016), treasurer (2016–2017).

Reviewing

2021	Workshop reviewer: SCiL 2022
2021	Workshop reviewer: BlackboxNLP 2021.
2021	Journal reviewer: Natural Language Engineering.
2021	Conference reviewer: EMNLP 2021. Recognized as an outstanding reviewer.
2020	Conference reviewer: CoNLL 2020.
2020	Conference reviewer: EMNLP 2020. Recognized as an outstanding reviewer.
2020	Conference reviewer: ACL 2020.
2019	Conference reviewer: CoNLL 2019.
2018	Conference reviewer: CoNLL 2018.
2018	Conference reviewer: ACL 2018. Recognized as a top reviewer.

Professional Memberships

2015–present	Linguistic Society of America (LSA).
2017–present	Association for Computational Linguistics (ACL).
2018–present	Cognitive Science Society.

Skills

Programming languages	Python, PyTorch, JavaScript, Haskell, C, Java, R, Scheme.
Natural languages	English (native), Bahasa Indonesia (basic conversation), Old English (basic reading ability), Old Norse (basic reading ability), Latin (basic reading ability).

Coursework

Undergraduate GPA: 4.0 *Graduate GPA:* 4.0

Computational Linguistics: Language and Computation I, Language and Computation II, Formal Foundations of Linguistic Theories, Computing Meaning

Natural Language Processing: Natural Language Processing, Machine Learning: Linguistic and Sequence Modeling

Syntax: Syntax I, Syntax II, Grammatical Diversity in US English

Phonetics/Phonology: Phonetics, Phonology I, Phonology II, The Phonetics/Phonology Interface

Semantics: Semantics I, Semantics II

Computer Science: Data Structures and Programming Techniques, Computational Tools for Data Science

Mathematics: Multivariable Calculus, Discrete Mathematics, Probability and Statistics, Advanced Statistical Methods

Other relevant courses: Linguistic Field Methods, Foundations of Cognitive Science

Tom McCoy: Research statement

How can we create computational systems that learn language as rapidly and robustly as humans do? By studying this topic, I hope to both improve models used in AI and to advance cognitive science by showing which computational mechanisms underlie humans' linguistic abilities. My research has three main components. First is **rethinking how we measure progress in NLP**. The prevalent approach involves scoring a model on a test set that is similar to the model's training set, but improvements on such benchmarks do not always indicate meaningful progress because it is unclear what drives the improvements: enhanced linguistic abilities or brittle, dataset-specific heuristics. I instead develop hypothesis-driven techniques that allow us to *understand* NLP models, both illuminating how they perform so well and diagnosing areas where they still fall short, such as requiring far more data than humans do. My second focus is **understanding the linguistic inductive biases of humans**, the factors that guide learning and enable rapid generalization. That is, what strategies do people use to learn so much more effectively than current computational models? My final focus—which is informed by insights from the first two—is **improving the inductive biases of NLP models** so that they can learn more quickly and generalize better. In particular, I study how linguistically-motivated architectures and training techniques can improve models' generalization abilities.

1 Rethinking how we measure progress in NLP: Hypothesis-driven evaluation and analysis

i. Hypothesis-driven evaluation: Standard NLP test sets are sampled from the same distribution as the training set. This approach has a major flaw: a model can score well on the test set by learning heuristics that succeed for frequent types of examples but fail on rarer cases, meaning that the model has not actually solved the intended training task.

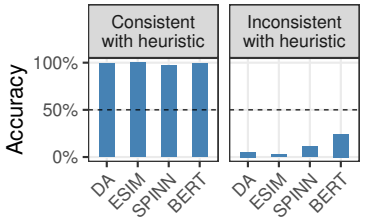


Figure 1: Inference models succeed on examples that can be solved with a shallow heuristic, but fail when attention to syntax is needed.

One focus of my research is addressing this possibility through a different evaluation paradigm: hypothesis-driven evaluations that illuminate what strategies models are using. We applied this approach to the task of natural language inference (NLI) to create the HANS dataset (McCoy, Pavlick, & Linzen 2019), which evaluates whether NLI models have adopted three syntactic heuristics that we hypothesized they were likely to adopt, such as assuming that sentence S entails any sentence whose words all appear in S (e.g., assuming that *the owl saw the fox* means the same thing as *the fox saw the owl*). Even a state-of-the-art model (BERT) performs poorly on HANS (Figure 1), consistent with the hypothesis that BERT has adopted this heuristic.

A major benefit of this approach is that, when a model fails on HANS, we have a clear hypothesis about what it is doing wrong, which can guide further work aimed at improving the model. For instance, we built from this hypothesis to create additional training examples that emphasize syntax. Adding these examples to the training set substantially decreased the model's reliance on heuristics that ignore word order (Min, McCoy, Das, Pitler, & Linzen 2020). Many other groups have also used HANS to motivate approaches for counteracting spurious heuristics, illustrating the impact that a carefully-designed evaluation can have.

Another area for which I have performed hypothesis-driven evaluation is natural language generation (NLG). Current language models (e.g., GPT-3) can generate coherent, grammatical passages of text. However, it is unclear how they achieve this success: do they have true generative abilities, or—as critics claim—are they simply copying text from their training set? In McCoy, Smolensky, Linzen, Gao, & Celikyilmaz (2021), we analyze whether NLG models are overly reliant on copying. Here the conclusion is more positive than with HANS: on a variety of linguistic levels, models show an impressive degree of novelty. For instance, the model GPT-2 generated the sentence *The **Sarrats** were lucky to have her as part of their lives*, which includes a novel plural word (*Sarrats*) accompanied by the proper syntactic consequences of this word's plurality: a plural verb, *were*, and a plural coreferential pronoun, *their*. Such examples show that some neural networks have a non-trivial amount of generative competence.

ii. Future work: Automating hypothesis search for evaluation: Going forward, one direction that I am excited about is automating the development of hypotheses about heuristics that models might be adopting. This could be done by starting with a manually-designed set of primitive features that are likely to be ingredients of such heuristics (e.g., example length), and then creating a model that combines these primitive features into hypothesized heuristics and generates evaluation examples to test for the use of those heuristics.

iii. Analyzing vector representations of symbolic structure: It has long been assumed that processing language requires representations of symbolic structure. Behavioral evaluations—such as our evaluations of novelty discussed in Section 1.i—give clear evidence that neural networks can process language well; yet their representations are vectors of continuous values, which look very different from the symbolic structures used in linguistic theory. How do neural networks encode linguistic structure within vector space?

Drawing on mathematical methods from cognitive science, we have developed a technique for analyzing models' vector representations as implicit encodings of symbol structures (McCoy, Linzen, Dunbar, & Smolensky 2018). The resulting analyses are much more interpretable than the original vectors. For instance, we showed that models trained to copy sequences encode position counting from left to right (*first, second...*); but models trained to reverse sequences encode position counting from right to left (*last, second-to-last...*). Analysis of more complex models has revealed more convoluted representational schemes, such as a scheme that includes the position *second-to-last word in the first of two clauses joined by a conjunction*.

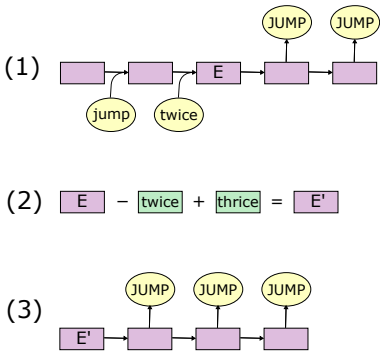


Figure 2: (1): Model being analyzed, which encodes the input (*jump twice*) as vector E then outputs *JUMP JUMP*. (2) Subtracting our analysis's predicted vector for *twice* and adding our analysis's predicted vector for *thrice* turns E into a new vector, E'. (3) Behavioral result of replacing E with E'.

Our analysis provides a closed-form equation for approximating a model's internal vector representations. This equation gives such a close approximation that we can use it to make targeted representational interventions to modify a model's behavior (Figure 2; Soulos, McCoy, Linzen, & Smolensky 2020), verifying that the representational structure we have revealed is causally linked to model behavior.

iv. Future work: Editing representations to remove socially-harmful biases: I intend to extend our representation-editing approach to large-scale models in order to remove undesired information (e.g., information about gender or ethnicity) that leads models to perpetuate or even amplify the societal prejudices in their training data.

2 Understanding linguistic inductive biases in humans

I study people's inductive biases using the paradigm of artificial language learning: teach people a specially-designed language and then test how they generalize it. This work has produced the first demonstration that people robustly extrapolate the recursive syntactic pattern of center embedding beyond the sentence sizes they have seen (McCoy, Culbertson, Smolensky, & Legendre 2021). For an ongoing second experiment, we have enriched a Bayesian model of language acquisition (Perfors, Tenenbaum, & Regier 2011) to test whether the bias we have observed specifically favors grammars generating unboundedly large syntactic structures, or more generally just favors simpler grammars. My aim in such experiments is to characterize people's inductive biases precisely enough to build those biases into computational models.

I also use neural networks as cognitive models to study what types of networks show the most human-like learning behavior. In an ongoing project, we are analyzing neural network models trained on the CHILDES corpus, which contains utterances made by parents to their children. Using this corpus allows us to bring our models into closer contact with cognitive questions, compared to existing models which are trained on corpora that are not representative of what children acquire language from (e.g., all of Wikipedia).

3 Improving models' inductive biases

The investigations of humans' biases discussed above give insight into biases that could be useful for models to have as well, but how can we give these biases to a model?

i. Meta-learning from synthetic languages: In McCoy, Grant, Smolensky, Griffiths, & Linzen (2020), we developed an approach for using *meta-learning* to give targeted inductive biases to a model. We first instantiate our desired biases as a distribution over synthetic languages. The model then meta-learns from languages sampled from this distribution to acquire our target biases; in meta-learning, exposure to many languages teaches a model about the commonalities across languages, enabling it to learn new languages more readily. This approach enabled our model to learn linguistic mappings from only 200 examples, vs. 20,000 examples without meta-learning. Meta-learning also enabled robust generalization, yielding 88% accuracy on out-of-distribution tests, vs. 6% without meta-learning.

ii. Future work: Exploration of which inductive biases give human-like learning behavior: This approach is a flexible way to give controlled biases to a model. I now plan to use it to instantiate different linguistic inductive biases and see which yield the most human-like learning, to test hypotheses about language acquisition and to provide candidate biases to add to NLP models to improve their data efficiency.

iii. Architectures with built-in linguistic structure: I have studied how we can give models a hierarchical inductive bias—a bias for generalizing based on hierarchical syntactic structure rather than linear order—which is a bias long argued to play a role in human language acquisition. Using synthetic datasets, we found that we could robustly impart this bias through use of a structured model whose computations are guided by syntax trees, but other approaches (multi-task learning and syntactic annotation of input data) had only minor effects (McCoy, Frank, & Linzen 2020). In later work, we showed that using tree-structured architectures also improves syntactic knowledge for models trained on natural text (Lepori, Linzen, & McCoy 2020). These results point toward structured architectures as a viable path for improving models' inductive biases.

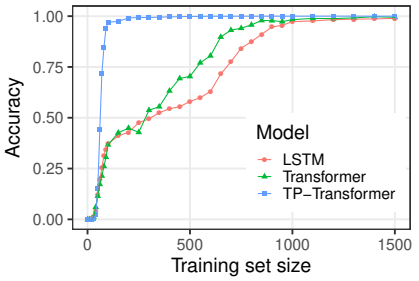


Figure 3: The TP-Transformer (argued for in our position piece) learns a symbolic task much more rapidly than standard NLP architectures.

In a position piece (Smolensky, McCoy, Fernandez, Goldrick, & Gao, in press), we argue for incorporating another type of structure into models—namely, a structure that disentangles contentful information (e.g., the identity of a word) from positional information (e.g., where the word appears). Collaborators at Microsoft have implemented the type of architecture that we argue for and have shown its strength on a variety of tasks. My own experiments have shown the rapid symbolic generalization that this architecture enables (Figure 3).

iv. Future work: Combining data-based and architecture-based methods: Adding structure to neural architectures can impart abstract inductive biases, such as a bias for hierarchical structure. Data-based approaches, such as our meta-learning approach, are better suited for imparting concrete biases that can be easily instantiated in data, such as biases regarding specific syntactic patterns. By combining the two approaches, I aim to adjust models' inductive biases at differing levels of abstraction.

4 Conclusion

I combine cognitive science and machine learning to analyze models of language and then use insights from those analyses to improve how models learn and generalize. Throughout these different threads of research, I focus on models' inductive biases and internal representations of linguistic structure. My long-term goal is to deepen our understanding of these two factors, allowing us to replicate the fast, robust learning capabilities that still set humans apart from even our most impressive computational models.

Tom McCoy: Teaching statement

1 Teaching experience

I have been a lab instructor for three courses at Johns Hopkins: *Intro to Computational Cognitive Science* (undergraduate class, taught by Tal Linzen), in which I led monthly lab sessions and created demos and assignments; *Computational Psycholinguistics* (joint undergraduate/graduate class, taught by Tal Linzen), for which I led biweekly lab sessions; and *World of Language* (undergraduate class, taught by Géraldine Legendre), in which I led 2 weekly sessions involving instruction and linguistic fieldwork. I have also been a teaching assistant for two joint undergraduate/graduate classes: *Syntax I* (taught by Géraldine Legendre) and *Foundations of Cognitive Science* (taught by Paul Smolensky).

2 Teaching philosophy

Multidisciplinarity: Like my research, my teaching is deeply multidisciplinary. Students develop a better understanding of computer science if they understand how it draws on and connects to other fields. For instance, in teaching about NLP, I emphasize several perspectives on language:

- *Linguistics:* How is language structured, and what demands does this structure place on our models?
- *Cognitive Science:* How do people learn and process language? How can this inform NLP?
- *Machine Learning:* Which approaches have been prevalent in the past, and which are prevalent today? What are the strengths and weaknesses of each perspective?

Skills over facts: The specific theories taught in a course may not be relevant to students after the course finishes. For instance, AI moves so quickly that today's state-of-the-art models will likely be outmoded before students graduate. Thus, my main goal when teaching is to impart the skills that remain relevant even when disciplines or students' interests change. These skills include the ability to analyze the target task to understand what computational challenges it poses; the ability to design an approach that can address these challenges; and the ability to analyze a proposed approach to determine what its strengths and weaknesses are. Of course, imparting these skills requires teaching some specific methods in detail; but the teaching of these methods is a step toward the goal, not the goal itself.

Hands-on experience: Many students learn best by doing. Thus, I emphasize homework assignments that give practical experience (e.g., implementing computational models) as well as course projects that provide a stepping stone to conducting research. As a lab instructor for *Intro to Computational Cognitive Science*, I developed interactive online demos illustrating course concepts such as Bayesian inference (Figure 1), funded by a grant awarded to me and the course instructor, Tal Linzen.[1] For *Computational Psycholinguistics*, I worked with the course instructor (Tal Linzen) to guide students through the process of conducting course projects; one student published his project in a conference. In *World of Language*, I taught students how to do linguistic fieldwork by developing theory-driven hypotheses about the language we were studying and testing these hypotheses by asking for linguistic judgments from a native speaker.

Figure 1: Part of a demo illustrating Bayesian word learning (Xu & Tenenbaum 2007). After a few examples, the Bayesian learner figures out that *naysayer* means "horse" (rather than being more specific—"horse number 4"—or more general—"mammal").

[1] https://cogscidemos.github.io/

3 Courses at Cornell

I am prepared to teach general and introductory computer science courses such as *Introduction to Computing Using Python*, *Object-Oriented Programming and Data Structures*, and *Discrete Structures*. For these courses, I will focus on grounding computational concepts in scientific questions or practical tasks; for instance, as a lab instructor for *Intro to Computational Cognitive Science*, I used the linguistic phenomenon of stress assignment to introduce basic Python programming.[2] I am also prepared to teach more in-depth courses in machine learning and natural language processing, such as *Natural Language Processing*, *Computational Linguistics*, and *Introduction to Machine Learning*.

Beyond these pre-existing courses, I would be interested in creating new courses in *Linguistics for NLP* or *Computational Models of Language Learning*, and in leading graduate seminars in which we discuss current literature on a chosen topic, such as analysis of NLP models.

Regardless of which courses I teach, I will promote an inclusive learning environment by encouraging participation in office hours or an online Piazza forum, for students who are more comfortable in these venues than in the large-group, in-person setting of the classroom. I will also provide flexibility with deadlines, to accommodate students who are undergoing extenuating circumstances in their lives outside of the classroom.

4 Mentoring

At Johns Hopkins University, I have mentored the research of four students: two Master's students whom I co-mentored along with their faculty advisors, and two undergraduates for whom I was the main research supervisor. Both Master's students and one of the undergraduates have since graduated; all three had first-author publications result from our work together, and all three have continued studying computational linguistics in academia or industry. The fourth student is still an undergraduate, and I expect that his project will result in a strong cognitive science publication. In all four cases, I have been very impressed with the work of these students, and all four of their projects have helped shape my own research interests.

In mentoring students, I focus on getting a research project started quickly, because nothing builds research skills like doing research—perhaps most importantly because thinking about one project in depth allows students to engage with scientific questions and to begin developing research agendas of their own. At the same time, I also emphasize discussion of foundational works relating to the student's interests, to instill the importance of connecting each research project to big-picture questions and debates. Beyond research, I emphasize the importance of mental health, encouraging mentees to take weekends off and get enough sleep, and to choose new meeting times or project deadlines if an existing deadline is proving stressful. Through these practices, I hope to help my advisees develop sustainable research practices that can carry them through a successful career.

[2] `https://colab.research.google.com/drive/1ghPQaTEdO9UH4s3gGD5OXmkYNvIwm2Zi`

Tom McCoy: Statement of contribution to diversity, equity, and inclusion

My commitment to diversity, equity, and inclusion is grounded in my experiences of research, teaching, and outreach. These experiences have shown me both the importance of diversity in enabling our field to do its best work, and also the systemic biases that continue to hamper diversity and inclusion in the field. Below I describe my views and experiences in promoting diversity.

Scholarship

Diversity and inclusion in the field: For the past two years, I have been actively involved in the Representation and Diversity Committee in the Johns Hopkins Department of Cognitive Science. In this committee, I have served two main roles. First, I am one of three co-organizers of our department's application mentoring program, in which prospective graduate students who belong to underrepresented groups are paired with current graduate students who provide feedback on the process of applying to graduate school. The goal of this program is to address the fact that some students have access to more application resources than others, such as having eminent faculty members as their undergraduate advisors who can guide them through applying. Through our program, we aim to provide this same sort of guidance even to students who do not have those advantages. Since this program began, several of the prospective students who signed up for it have been accepted as graduate students in our department.

The second initiative I have helped to run is motivated by the fact that the group of undergraduates who do research in our department is not as diverse as the group of students who take courses in our department. To work toward addressing this disparity, I and one other graduate student wrote a statement that faculty members in our department added to the syllabi for their courses. This statement describes where to look for research opportunities and also encourages students to reach out to faculty members and graduate students to see if they could support a research assistant. By including this statement, we aim to raise awareness of the paths toward research that currently only some types of students pursue. Along similar themes of raising awareness, this statement also briefly describes what graduate school is like and describes how to seek more information about applying to graduate school. For example, we mention the fact that being a PhD student is a paid job; a lack of awareness of this fact could prevent otherwise qualified students from applying, since they might assume that a PhD would incur further student debt.

A third way in which I promote an equitable workplace is more indirect, but no less important, which is that I actively take on departmental service roles. Such service roles include being a departmental representative for the graduate student organization, helping to organize the recruitment visit for prospective students, or mentoring more junior graduate students in our department's program for matching junior and senior graduate students. Though most of these service roles are not directly focused on diversity and inclusion, I still view them as critical to these goals because, throughout academia, departmental service roles disproportionately fall on the shoulders of female researchers and minority researchers. Thus, by doing my part to carry out service roles, I aim to reduce the burden on these groups.

Going forward, in building my own lab, I plan to continue such outreach initiatives in order to recruit a diverse group of students. In order to promote not just recruitment but also retention, I will continue to place top priority on supporting my students' mental health, as I have done with the students whom I have mentored so far. Steps I take in this direction include regularly reminding students to take breaks and get enough sleep (which sound like obvious points, but which new students often need to be reminded about), and having biweekly check-ins where I ask students about whether they are finding their projects stressful or unsatisfying, and if so figuring out ways to address these issues.

Promoting fairness through my research: Most modern Artificial Intelligence is driven by neural networks, from self-driving cars to machine translation systems. Despite their strengths, neural networks have some major weaknesses with societally deleterious effects. First, they replicate and even amplify societal prejudices in their training data, resulting in problems such as text-generation systems spewing hate speech (Gehman et al. 2020) and facial analysis systems getting an accuracy over 99% for light-skinned men yet only 75% for darker-skinned women (Buolamwini & Gebru 2018). Such issues could be mitigated by revising the training data; but current models are so data-hungry that training them typically requires enormous datasets that cannot feasibly be audited (Bender et al. 2021). Both of these problems—incorrect generalization and data hunger—derive from the inadequate learning biases of current models. One major focus of my research is to develop approaches for controlling and improving the learning biases of neural networks, which could then help to create models that do not require such massive datasets and which generalize in less prejudiced ways. One particular project that I have planned in this direction is to use an approach I have developed for altering the information inside neural network vector representations (McCoy, Linzen, Dunbar, and Smolensky 2018; Soulos, McCoy, Linzen, and Smolensky 2020) to edit out information that is not relevant for the target task but that could lead to prejudiced behavior, such as gender information.

Instruction

As a teaching assistant for the course Foundations of Cognitive Science, I worked with the course instructor (Paul Smolensky) to increase the diversity of the authors represented in the course's reading list. In addition, we used this as an opportunity to discuss with our students the field's (historical and ongoing) biases that have led to such little diversity in the authors whose work is considered foundational. As an instructor, I plan to continue this emphasis on presenting diverse perspectives in the reading lists that I create.

On a more practical level, as detailed in my teaching statement, I plan to take several steps to make my courses a welcoming environment for students from a variety of backgrounds. These steps include providing a generous allowance of late days for homework assignments (to give students room to handle any difficulties that may arise in their personal lives), and providing several different types of venues for course interaction so that students can participate in whichever type of venue they feel most comfortable in (namely, these venues would be the large-group lecture, small-group or one-on-one meetings in office hours, and online discussion in a Piazza forum).

Outreach

Since 2013, I have been closely involved in organizing the North American Computational Linguistics Open competition (NACLO), a contest that introduces high school students to computational linguistics so that they can start studying it as soon as they start college (instead of finding out about it later in their college career, at which point it might be too late to specialize in this area). 1,000 to 1,500 students participate in NACLO each year; last year, 42% of participants were female, a high proportion for a computational initiative. Though NACLO does not formally keep track of former participants, there are many anecdotal cases of participants going on to study linguistics or natural language processing; indeed, I participated in NACLO as a high school student, and it was what first introduced me to the field. Going forward, I plan to continue my involvement with NACLO, both by using my research as a source for problems for the national contest (as I have done with previous projects), and by visiting local schools to teach about linguistics using NACLO puzzles (as I have done in Baltimore and New Haven).

Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference

R. Thomas McCoy,[1] Ellie Pavlick,[2] & Tal Linzen[1]
[1]Department of Cognitive Science, Johns Hopkins University
[2]Department of Computer Science, Brown University
tom.mccoy@jhu.edu, ellie_pavlick@brown.edu, tal.linzen@jhu.edu

Abstract

A machine learning system can score well on a given test set by relying on heuristics that are effective for frequent example types but break down in more challenging cases. We study this issue within natural language inference (NLI), the task of determining whether one sentence entails another. We hypothesize that statistical NLI models may adopt three fallible syntactic heuristics: the lexical overlap heuristic, the subsequence heuristic, and the constituent heuristic. To determine whether models have adopted these heuristics, we introduce a controlled evaluation set called HANS (Heuristic Analysis for NLI Systems), which contains many examples where the heuristics fail. We find that models trained on MNLI, including BERT, a state-of-the-art model, perform very poorly on HANS, suggesting that they have indeed adopted these heuristics. We conclude that there is substantial room for improvement in NLI systems, and that the HANS dataset can motivate and measure progress in this area.

1 Introduction

Neural networks excel at learning the statistical patterns in a training set and applying them to test cases drawn from the same distribution as the training examples. This strength can also be a weakness: statistical learners such as standard neural network architectures are prone to adopting shallow heuristics that succeed for the majority of training examples, instead of learning the underlying generalizations that they are intended to capture. If such heuristics often yield correct outputs, the loss function provides little incentive for the model to learn to generalize to more challenging cases as a human performing the task would.

This issue has been documented across domains in artificial intelligence. In computer vision, for example, neural networks trained to recognize objects are misled by contextual heuristics: a network that is able to recognize monkeys in a typical context with high accuracy may nevertheless label a monkey holding a guitar as a human, since in the training set guitars tend to co-occur with humans but not monkeys (Wang et al., 2018). Similar heuristics arise in visual question answering systems (Agrawal et al., 2016).

The current paper addresses this issue in the domain of natural language inference (NLI), the task of determining whether a **premise** sentence entails (i.e., implies the truth of) a **hypothesis** sentence (Condoravdi et al., 2003; Dagan et al., 2006; Bowman et al., 2015). As in other domains, neural NLI models have been shown to learn shallow heuristics, in this case based on the presence of specific words (Naik et al., 2018; Sanchez et al., 2018). For example, a model might assign a label of *contradiction* to any input containing the word *not*, since *not* often appears in the examples of contradiction in standard NLI training sets.

The focus of our work is on heuristics that are based on superficial **syntactic** properties. Consider the following sentence pair, which has the target label *entailment*:

(1) *Premise:* The judge was paid by the actor.
Hypothesis: The actor paid the judge.

An NLI system that labels this example correctly might do so not by reasoning about the meanings of these sentences, but rather by assuming that the premise entails any hypothesis whose words all appear in the premise (Dasgupta et al., 2018; Naik et al., 2018). Crucially, if the model is using this heuristic, it will predict *entailment* for (2) as well, even though that label is incorrect in this case:

(2) *Premise:* The actor was paid by the judge.
Hypothesis: The actor paid the judge.

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3428–3448
Florence, Italy, July 28 - August 2, 2019.

Heuristic	Definition	Example
Lexical overlap	Assume that a premise entails all hypotheses constructed from words in the premise	**The doctor** was **paid** by **the actor**. $\xrightarrow[\text{WRONG}]{}$ The doctor paid the actor.
Subsequence	Assume that a premise entails all of its contiguous subsequences.	The doctor near **the actor danced**. $\xrightarrow[\text{WRONG}]{}$ The actor danced.
Constituent	Assume that a premise entails all complete subtrees in its parse tree.	If **the artist slept**, the actor ran. $\xrightarrow[\text{WRONG}]{}$ The artist slept.

Table 1: The heuristics targeted by the HANS dataset, along with examples of incorrect entailment predictions that these heuristics would lead to.

We introduce a new evaluation set called HANS (Heuristic Analysis for NLI Systems), designed to diagnose the use of such fallible structural heuristics.[1] We target three heuristics, defined in Table 1. While these heuristics often yield correct labels, they are not valid inference strategies because they fail on many examples. We design our dataset around such examples, so that models that employ these heuristics are guaranteed to fail on particular subsets of the dataset, rather than simply show lower overall accuracy.

We evaluate four popular NLI models, including BERT, a state-of-the-art model (Devlin et al., 2019), on the HANS dataset. All models performed substantially below chance on this dataset, barely exceeding 0% accuracy in most cases. We conclude that their behavior is consistent with the hypothesis that they have adopted these heuristics.

Contributions: This paper has three main contributions. First, we introduce the HANS dataset, an NLI evaluation set that tests specific hypotheses about invalid heuristics that NLI models are likely to learn. Second, we use this dataset to illuminate interpretable shortcomings in state-of-the-art models trained on MNLI (Williams et al., 2018b); these shortcoming may arise from inappropriate model inductive biases, from insufficient signal provided by training datasets, or both. Third, we show that these shortcomings can be made less severe by augmenting a model's training set with the types of examples present in HANS. These results indicate that there is substantial room for improvement for current NLI models and datasets, and that HANS can serve as a tool for motivating and measuring progress in this area.

[1]GitHub repository with data and code: https://github.com/tommccoy1/hans

2 Syntactic Heuristics

We focus on three heuristics: the lexical overlap heuristic, the subsequence heuristic, and the constituent heuristic, all defined in Table 1. These heuristics form a hierarchy: the constituent heuristic is a special case of the subsequence heuristic, which in turn is a special case of the lexical overlap heuristic. Table 2 in the next page gives examples where each heuristic succeeds and fails.

There are two reasons why we expect these heuristics to be adopted by a statistical learner trained on standard NLI training datasets such as SNLI (Bowman et al., 2015) or MNLI (Williams et al., 2018b). First, the MNLI training set contains far more examples that support the heuristics than examples that contradict them:[2]

Heuristic	Supporting Cases	Contradicting Cases
Lexical overlap	2,158	261
Subsequence	1,274	72
Constituent	1,004	58

Even the 261 contradicting cases in MNLI may not provide strong evidence against the heuristics. For example, 133 of these cases contain negation in the premise but not the hypothesis, as in (3). Instead of using these cases to overrule the lexical overlap heuristic, a model might account for them by learning to assume that the label is *contradiction* whenever there is negation in the premise but not the hypothesis (McCoy and Linzen, 2019):

(3) a. I **don't** care. $\nrightarrow$ I care.
b. This is **not** a contradiction. $\nrightarrow$ This is a contradiction.

[2]In this table, the lexical overlap counts include the subsequence counts, which include the constituent counts.

Heuristic	Premise	Hypothesis	Label
Lexical overlap heuristic	The banker near the judge saw the actor.	The banker saw the actor.	E
	The lawyer was advised by the actor.	The actor advised the lawyer.	E
	The doctors visited the lawyer.	The lawyer visited the doctors.	N
	The judge by the actor stopped the banker.	The banker stopped the actor.	N
Subsequence heuristic	The artist and the student called the judge.	The student called the judge.	E
	Angry tourists helped the lawyer.	Tourists helped the lawyer.	E
	The judges heard the actors resigned.	The judges heard the actors.	N
	The senator near the lawyer danced.	The lawyer danced.	N
Constituent heuristic	Before the actor slept, the senator ran.	The actor slept.	E
	The lawyer knew that the judges shouted.	The judges shouted.	E
	If the actor slept, the judge saw the artist.	The actor slept.	N
	The lawyers resigned, or the artist slept.	The artist slept.	N

Table 2: Examples of sentences used to test the three heuristics. The *label* column shows the correct label for the sentence pair; *E* stands for *entailment* and *N* stands for *non-entailment*. A model relying on the heuristics would label all examples as *entailment* (incorrectly for those marked as N).

There are some examples in MNLI that contradict the heuristics in ways that are not easily explained away by other heuristics; see Appendix A for examples. However, such cases are likely too rare to discourage a model from learning these heuristics. MNLI contains data from multiple genres, so we conjecture that the scarcity of contradicting examples is not just a property of one genre, but rather a general property of NLI data generated in the crowdsourcing approach used for MNLI. We thus hypothesize that any crowdsourced NLI dataset would make our syntactic heuristics attractive to statistical learners without strong linguistic priors.

The second reason we might expect current NLI models to adopt these heuristics is that their input representations may make them susceptible to these heuristics. The lexical overlap heuristic disregards the order of the words in the sentence and considers only their identity, so it is likely to be adopted by bag-of-words NLI models (e.g., Parikh et al. 2016). The subsequence heuristic considers linearly adjacent chunks of words, so one might expect it to be adopted by standard RNNs, which process sentences in linear order. Finally, the constituent heuristic appeals to components of the parse tree, so one might expect to see it adopted by tree-based NLI models (Bowman et al., 2016).

3 Dataset Construction

For each heuristic, we generated five templates for examples that support the heuristic and five templates for examples that contradict it. Below is one template for the subsequence heuristic; see Appendix B for a full list of templates.

(4) The N_1 P the N_2 V. $\nrightarrow$ The N_2 V.
The lawyer by the actor ran. $\nrightarrow$ *The actor ran.*

We generated 1,000 examples from each template, for a total of 10,000 examples per heuristic. Some heuristics are special cases of others, but we made sure that the examples for one heuristic did not also fall under a more narrowly defined heuristic. That is, for lexical overlap cases, the hypothesis was not a subsequence or constituent of the premise; for subsequence cases, the hypothesis was not a constituent of the premise.

3.1 Dataset Controls

Plausibility: One advantage of generating examples from templates—instead of, e.g., modifying naturally-occurring examples—is that we can ensure the plausibility of all generated sentences. For example, we do not generate cases such as *The student read the book* $\nrightarrow$ *The book read the student*, which could ostensibly be solved using a hypothesis-plausibility heuristic. To achieve this, we drew our core vocabulary from Ettinger et al. (2018), where every noun was a plausible subject of every verb or a plausible object of every transitive verb. Some templates required expanding this core vocabulary; in those cases, we manually curated the additions to ensure plausibility.

Selectional criteria: Some of our example types depend on the availability of lexically-specific verb frames. For example, (5) requires awareness of the fact that *believed* can take a clause (*the lawyer saw the officer*) as its complement:

(5) The doctor believed the lawyer saw the officer. $\nrightarrow$ The doctor believed the lawyer.

It is arguably unfair to expect a model to understand this example if it had only ever encountered *believe* with a noun phrase object (e.g., *I believed* ***the man***). To control for this issue, we only chose verbs that appeared at least 50 times in the MNLI training set in all relevant frames.

4 Experimental Setup

Since HANS is designed to probe for structural heuristics, we selected three models that exemplify popular strategies for representing the input sentence: DA, a bag-of-words model; ESIM, which uses a sequential structure; and SPINN, which uses a syntactic parse tree. In addition to these three models, we included BERT, a state-of-the-art model for MNLI. The following paragraphs provide more details on these models.

DA: The Decomposable Attention model (DA; Parikh et al., 2016) uses a form of attention to align words in the premise and hypothesis and to make predictions based on the aggregation of this alignment. It uses no word order information and can thus be viewed as a bag-of-words model.

ESIM: The Enhanced Sequential Inference Model (ESIM; Chen et al., 2017) uses a modified bidirectional LSTM to encode sentences. We use the variant with a sequential encoder, rather than the tree-based Hybrid Inference Model (HIM).

SPINN: The Stack-augmented Parser-Interpreter Neural Network (SPINN; Bowman et al., 2016) is tree-based: it encodes sentences by combining phrases based on a syntactic parse. We use the SPINN-PI-NT variant, which takes a parse tree as an input (rather than learning to parse). For MNLI, we used the parses provided in the MNLI release; for HANS, we used parse templates that we created based on parses from the Stanford PCFG Parser 3.5.2 (Klein and Manning, 2003), the same parser used to parse MNLI. Based on manual inspection, this parser generally provided correct parses for HANS examples.

BERT: The Bidirectional Encoder Representations from Transformers model (BERT; Devlin et al., 2019) is a Transformer model that uses attention, rather than recurrence, to process sentences. We use the `bert-base-uncased` pretrained model and fine-tune it on MNLI.

Implementation and evaluation: For DA and ESIM, we used the implementations from AllenNLP (Gardner et al., 2017). For SPINN[3] and BERT,[4] we used code from the GitHub repositories for the papers introducing those models.

We trained all models on MNLI. MNLI uses three labels (*entailment*, *contradiction*, and *neutral*). We chose to annotate HANS with two labels only (*entailment* and *non-entailment*) because the distinction between *contradiction* and *neutral* was often unclear for our cases.[5] For evaluating a model on HANS, we took the highest-scoring label out of *entailment*, *contradiction*, and *neutral*; we then translated *contradiction* or *neutral* labels to *non-entailment*. An alternate approach would have been to add the *contradiction* and *neutral* scores to determine a score for *non-entailment*; we found little difference between these approaches, since the models almost always assigned more than 50% of the label probability to a single label.[6]

5 Results

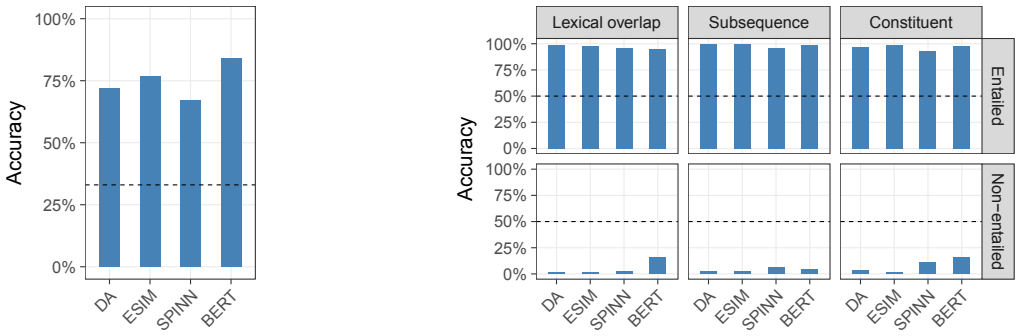
All models achieved high scores on the MNLI test set (Figure 1a), replicating the accuracies found in past work (DA: Gururangan et al. 2018; ESIM: Williams et al. 2018b; SPINN: Williams et al. 2018a; BERT: Devlin et al. 2019). On the HANS dataset, all models almost always assigned the correct label in the cases where the label is *entailment*, i.e., where the correct answer is in line with the hypothesized heuristics. However, they all performed poorly—with accuracies less than 10% in most cases, when chance is 50%—on the cases where the heuristics make incorrect predictions

[3] `https://github.com/stanfordnlp/spinn`; we used the NYU fork at `https://github.com/nyu-mll/spinn`.

[4] `https://github.com/google-research/bert`

[5] For example, with *The actor was helped by the judge* $\nrightarrow$ *The actor helped the judge*, it is possible that the actor did help the judge, pointing to a label of *neutral*; yet the premise does pragmatically imply that the actor did not help the judge, meaning that this pair could also fit the non-strict definition of *contradiction* used in NLI annotation.

[6] We also tried training the models on MNLI with *neutral* and *contradiction* collapsed into *non-entailment*; this gave similar results as collapsing after training (Appendix D) .



(a) (b)

Figure 1: (a) Accuracy on the MNLI test set. (b) Accuracies on six sub-components of the HANS evaluation set; each sub-component is defined by its correct label and the heuristic it addresses. The dashed lines indicate chance performance. All models behaved as we would expect them to if they had adopted the heuristics targeted by HANS. That is, they nearly always predicted *entailment* for the examples in HANS, leading to near-perfect accuracy when the true label is *entailment*, and near-zero accuracy when the true label is *non-entailment*.

(Figure 1b). Thus, despite their high scores on the MNLI test set, all four models behaved in a way consistent with the use of the heuristics targeted in HANS, and not with the correct rules of inference.

Comparison of models: Both DA and ESIM had near-zero performance across all three heuristics. These models might therefore make no distinction between the three heuristics, but instead treat them all as the same phenomenon, i.e. lexical overlap. Indeed, for DA, this must be the case, as this model does not have access to word order; ESIM does in theory have access to word order information but does not appear to use it here.

SPINN had the best performance on the subsequence cases. This might be due to the tree-based nature of its input: since the subsequences targeted in these cases were explicitly chosen not to be constituents, they do not form cohesive units in SPINN's input in the way they do for sequential models. SPINN also outperformed DA and ESIM on the constituent cases, suggesting that SPINN's tree-based representations moderately helped it learn how specific constituents contribute to the overall sentence. Finally, SPINN did worse than the other models on constituent cases where the correct answer is *entailment*. This moderately greater balance between accuracy on entailment and non-entailment cases further indicates that SPINN is less likely than the other models to assume that constituents of the premise are entailed; this harms its performance in cases where that assumption happens to lead to the correct answer.

BERT did slightly worse than SPINN on the subsequence cases, but performed noticeably less poorly than all other models at both the constituent and lexical overlap cases (though it was still far below chance). Its performance particularly stood out for the lexical overlap cases, suggesting that some of BERT's success at MNLI may be due to a greater tendency to incorporate word order information compared to other models.

Analysis of particular example types: In the cases where a model's performance on a heuristic was perceptibly above zero, accuracy was not evenly spread across subcases (for case-by-case results, see Appendix C). For example, within the lexical overlap cases, BERT achieved 39% accuracy on conjunction (e.g., *The actor and the doctor saw the artist* $\nrightarrow$ *The actor saw the doctor*) but 0% accuracy on subject/object swap (*The judge called the lawyer* $\nrightarrow$ *The lawyer called the judge*). Within the constituent heuristic cases, BERT achieved 49% accuracy at determining that a clause embedded under *if* and other conditional words is not entailed (*If the doctor resigned, the lawyer danced* $\nrightarrow$ *The doctor resigned*), but 0% accuracy at identifying that the clause outside of the conditional clause is also not entailed (*If the doctor resigned, the lawyer danced* $\nrightarrow$ *The lawyer danced*).

6 Discussion

Independence of heuristics: Though each heuristic is most closely related to one class of model (e.g., the constituent heuristic is related to tree-based models), all models failed on cases illustrating all three heuristics. This finding is unsurprising since these heuristics are closely related

to each other, meaning that an NLI model may adopt all of them, even the ones not specifically targeting that class of model. For example, the subsequence and constituent heuristics are special cases of the lexical overlap heuristic, so all models can fail on cases illustrating all heuristics, because all models have access to individual words.

Though the heuristics form a hierarchy—the constituent heuristic is a subcase of the subsequence heuristic, which is a subcase of the lexical overlap heuristic—this hierarchy does not necessarily predict the performance of our models. For example, BERT performed worse on the subsequence heuristic than on the constituent heuristic, even though the constituent heuristic is a special case of the subsequence heuristic. Such behavior has two possible causes. First, it could be due to the specific cases we chose for each heuristic: the cases chosen for the subsequence heuristic may be inherently more challenging than the cases chosen for the constituent heuristic, even though the constituent heuristic as a whole is a subset of the subsequence one. Alternately, it is possible for a model to adopt a more general heuristic (e.g., the subsequence heuristic) but to make an exception for some special cases (e.g., the cases to which the constituent heuristic could apply).

Do the heuristics arise from the architecture or the training set? The behavior of a trained model depends on both the training set and the model's architecture. The models' poor results on HANS could therefore arise from architectural limitations, from insufficient signal in the MNLI training set, or from both.

The fact that SPINN did markedly better at the constituent and subsequence cases than ESIM and DA, even though the three models were trained on the same dataset, suggests that MNLI does contain some signal that can counteract the appeal of the syntactic heuristics tested by HANS. SPINN's structural inductive biases allow it to leverage this signal, but the other models' biases do not.

Other sources of evidence suggest that the models' failure is due in large part to insufficient signal from the MNLI training set, rather than the models' representational capacities alone. The BERT model we used (`bert-base-uncased`) was found by Goldberg (2019) to achieve strong results in syntactic tasks such as subject-verb agreement prediction, a task that minimally requires a distinction between the subject and direct object of a sentence (Linzen et al., 2016; Gulordava et al., 2018; Marvin and Linzen, 2018). Despite this evidence that BERT has access to relevant syntactic information, its accuracy was 0% on the subject-object swap cases (e.g., *The doctor saw the lawyer* $\nrightarrow$ *The lawyer saw the doctor*). We believe it is unlikely that our fine-tuning step on MNLI, a much smaller corpus than the corpus BERT was trained on, substantially changed the model's representational capabilities. Even though the model most likely had access to information about subjects and objects, then, MNLI did not make it clear how that information applies to inference. Supporting this conclusion, McCoy et al. (2019) found little evidence of compositional structure in the InferSent model, which was trained on SNLI, even though the same model type (an RNN) did learn clear compositional structure when trained on tasks that underscored the need for such structure. These results further suggest that the models' poor compositional behavior arises more because of the training set than because of model architecture.

Finally, our BERT-based model differed from the other models in that it was pretrained on a massive amount of data on a masking task and a next-sentence classification task, followed by fine-tuning on MNLI, while the other models were only trained on MNLI; we therefore cannot rule out the possibility that BERT's comparative success at HANS was due to the greater amount of data it has encountered rather than any architectural features.

Is the dataset too difficult? To assess the difficulty of our dataset, we obtained human judgments on a subset of HANS from 95 participants on Amazon Mechanical Turk as well as 3 expert annotators (linguists who were unfamiliar with HANS: 2 graduate students and 1 postdoctoral researcher). The average accuracy was 76% for Mechanical Turk participants and 97% for expert annotators; further details are in Appendix F.

Our Mechanical Turk results contrast with those of Nangia and Bowman (2019), who report an accuracy of 92% in the same population on examples from MNLI; this indicates that HANS is indeed more challenging for humans than MNLI is. The difficulty of some of our examples is in line with past psycholinguistic work in which humans have been shown to incorrectly answer comprehension questions for some of our subsequence subcases. For example, in an experiment in which participants read the sentence *As Jerry played the violin*

gathered dust in the attic, some participants answered *yes* to the question *Did Jerry play the violin?* (Christianson et al., 2001).

Crucially, although Mechanical Turk annotators found HANS to be harder overall than MNLI, their accuracy was similar whether the correct answer was *entailment* (75% accuracy) or *non-entailment* (77% accuracy). The contrast between the balance in the human errors across labels and the stark imbalance in the models' errors (Figure 1b) indicates that human errors are unlikely to be driven by the heuristics targeted in the current work.

7 Augmenting the training data with HANS-like examples

The failure of the models we tested raises the question of what it would take to do well on HANS. One possibility is that a different type of model would perform better. For example, a model based on hand-coded rules might handle HANS well. However, since most models we tested are in theory capable of handling HANS's examples but failed to do so when trained on MNLI, it is likely that performance could also be improved by training the same architectures on a dataset in which these heuristics are less successful.

To test that hypothesis, we retrained each model on the MNLI training set augmented with a dataset structured exactly like HANS (i.e. using the same thirty subcases) but containing no specific examples that appeared in HANS. Our additions comprised 30,000 examples, roughly 8% of the size of the original MNLI training set (392,702 examples). In general, the models trained on the augmented MNLI performed very well on HANS (Figure 2); the one exception was that the DA model performed poorly on subcases for which a bag-of-words representation was inadequate.[7] This experiment is only an initial exploration and leaves open many questions about the conditions under which a model will successfully avoid a heuristic; for example, how many contradicting examples are required? At the same time, these results do suggest that, to prevent a model from learning a heuristic, one viable approach is to use a training set that does not support this heuristic.

[7]The effect on MNLI test set performance was less clear; the augmentation with HANS-like examples improved MNLI test set performance for BERT (84.4% vs. 84.1%) and ESIM (77.6% vs 77.3%) but hurt performance for DA (66.0% vs. 72.4%) and SPINN (63.9% vs. 67.0%).

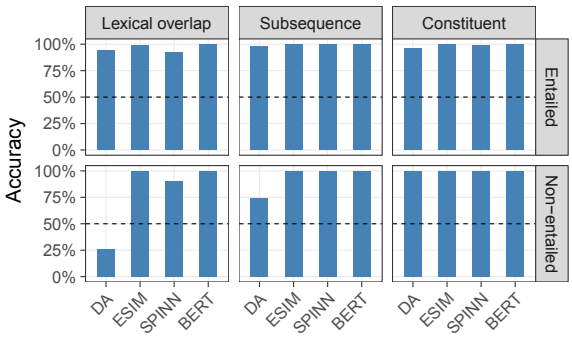


Figure 2: HANS accuracies for models trained on MNLI plus examples of all 30 categories in HANS.

Transfer across HANS subcases: The positive results of the HANS-like augmentation experiment are compatible with the possibility that the models simply memorized the templates that made up HANS's thirty subcases. To address this, we retrained our models on MNLI augmented with *subsets* of the HANS cases (withholding some cases; see Appendix E for details), then tested the models on the withheld cases.

The results of one of the transfer experiments, using BERT, are shown in Table 3. There were some successful cases of transfer; e.g., BERT performed well on the withheld categories with sentence-initial adverbs, regardless of whether the correct label was *non-entailment* or *entailment*. Such successes suggest that BERT is able to learn from some specific subcases that it should rule out the broader heuristics; in this case, the non-withheld cases plausibly informed BERT not to indiscriminately follow the constituent heuristic, encouraging it to instead base its judgments on the specific adverbs in question (e.g., *certainly* vs. *probably*). However, the models did not always transfer successfully; e.g., BERT had 0% accuracy on entailed passive examples when such examples were withheld, likely because the training set still included many non-entailed passive examples, meaning that BERT may have learned to assume that all sentences with passive premises are cases of non-entailment. Thus, though the models do seem to be able to rule out the broadest versions of the heuristics and transfer that knowledge to some new cases, they may still fall back to the heuristics for other cases. For further results involving withheld categories, see Appendix E.

Transfer to an external dataset: Finally, we tested models on the `comp_same_short` and

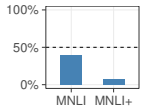
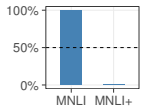
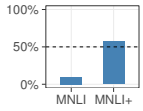
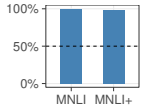
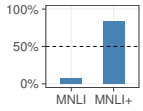
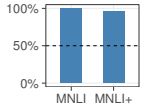
Withheld category	Results
Lexical overlap: Conjunctions (↛) *The doctor saw the author and the tourist.* ↛ *The author saw the tourist.*	
Lexical overlap: Passives (→) *The authors were helped by the actor.* → *The actor helped the authors.*	
Subsequence: NP/Z (↛) *Before the actor moved the doctor arrived.* ↛ *The actor moved the doctor.*	
Subsequence: PP on object (→) *The authors saw the judges by the doctor.* → *The authors saw the judges.*	
Constituent: Adverbs (↛) *Probably the artists helped the authors.* ↛ *The artists helped the authors.*	
Constituent: Adverbs (→) *Certainly the lawyers shouted.* → *The lawyers shouted.*	

Table 3: Accuracies for BERT fine-tuned on basic MNLI and on MNLI+, which is MNLI augmented with most HANS categories except withholding the categories in this table. The two lexical overlap cases shown here are adversarial in that MNLI+ contains cases superficially similar to them but with opposite labels (namely, the *Conjunctions (→)* and *Passives (↛)* cases from Table 4 in the Appendix). The remaining cases in this table are not adversarial in this way.

`comp_same_long` datasets from Dasgupta et al. (2018), which consist of lexical overlap cases:

(6) the famous and arrogant cat is not more nasty than the dog with glasses in a white dress. ↛ the dog with glasses in a white dress is not more nasty than the famous and arrogant cat.

This dataset differs from HANS in at least three important ways: it is based on a phenomenon not present in HANS (namely, comparatives); it uses a different vocabulary from HANS; and many of its sentences are semantically implausible.

We used this dataset to test both BERT fine-tuned on MNLI, and BERT fine-tuned on MNLI augmented with HANS-like examples. The augmentation improved performance modestly for the long examples and dramatically for the short examples, suggesting that training with HANS-like examples has benefits that extend beyond HANS.[8]

[8]We hypothesize that HANS helps more with short examples because most HANS sentences are short.

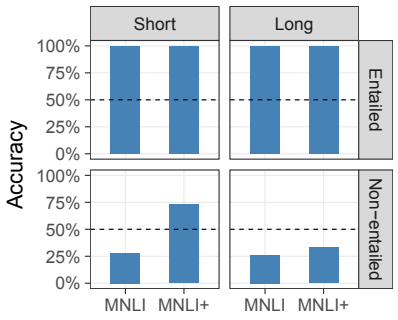


Figure 3: Results on the lexical overlap cases from Dasgupta et al. (2018) for BERT fine-tuned on MNLI or on MNLI augmented with HANS-like examples.

8 Related Work

8.1 Analyzing trained models

This project relates to an extensive body of research on exposing and understanding weaknesses in models' learned behavior and representations. In the NLI literature, Poliak et al. (2018b) and Gururangan et al. (2018) show that, due to biases in NLI datasets, it is possible to achieve far better than chance accuracy on those datasets by only looking at the hypothesis. Other recent works address possible ways in which NLI models might use fallible heuristics, focusing on semantic phenomena, such as lexical inferences (Glockner et al., 2018) or quantifiers (Geiger et al., 2018), or biases based on specific words (Sanchez et al., 2018). Our work focuses instead on *structural* phenomena, following the proof-of-concept work done by Dasgupta et al. (2018). Our focus on using NLI to address how models capture structure follows some older work about using NLI for the evaluation of parsers (Rimell and Clark, 2010; Mehdad et al., 2010).

NLI has been used to investigate many other types of linguistic information besides syntactic structure (Poliak et al., 2018a; White et al., 2017). Outside NLI, multiple projects have used classification tasks to understand what linguistic and/or structural information is present in vector encodings of sentences (e.g., Adi et al., 2017; Ettinger et al., 2018; Conneau et al., 2018). We instead choose the behavioral approach of using task performance on critical cases. Unlike the classification approach, this approach is agnostic to model structure; our dataset could be used to evaluate a symbolic NLI system just as easily as a neural one, whereas typical classification approaches only work for models with vector representations.

8.2 Structural heuristics

Similar to our lexical overlap heuristic, Dasgupta et al. (2018), Nie et al. (2018), and Kim et al. (2018) also tested NLI models on specific phenomena where word order matters; we use a larger set of phenomena to study a more general notion of lexical overlap that is less dependent on the properties of a single phenomenon, such as passives. Naik et al. (2018) also find evidence that NLI models use a lexical overlap heuristic, but our approach is substantially different from theirs.[9]

This work builds on our pilot study in McCoy and Linzen (2019), which studied one of the subcases of the subsequence heuristic. Several of our subsequence subcases are inspired by psycholinguistics research (Bever, 1970; Frazier and Rayner, 1982; Tabor et al., 2004); these works have aims similar to ours but are concerned with the representations used by humans rather than neural networks.

Finally, all of our constituent heuristic subcases depend on the implicational behavior of specific words. Several past works (Pavlick and Callison-Burch, 2016; Rudinger et al., 2018; White et al., 2018; White and Rawlins, 2018) have studied such behavior for verbs (e.g., *He knows it is raining* entails *It is raining*, while *He believes it is raining* does not). We extend that approach by including other types of words with specific implicational behavior, namely conjunctions (*and, or*), prepositions that take clausal arguments (*if, because*), and adverbs (*definitely, supposedly*). MacCartney and Manning (2009) also discuss the implicational behavior of these various types of words within NLI.

8.3 Generalization

Our work suggests that test sets drawn from the same distribution as the training set may be inadequate for assessing whether a model has learned to perform the intended task. Instead, it is also necessary to evaluate on a generalization set that departs from the training distribution. McCoy et al. (2018) found a similar result for the task of question formation; different architectures that all succeeded on the test set failed on the generalization set in different ways, showing that the test set alone was not sufficient to determine what the models had learned. This effect can arise not just from different architectures but also from different initializations of the same architecture (Weber et al., 2018).

[9] Naik et al. (2018) diagnose the lexical overlap heuristic by appending *and true is true* to existing MNLI hypotheses, which decreases lexical overlap but does not change the sentence pair's label. We instead generate new sentence pairs for which the words in the hypothesis all appear in the premise.

9 Conclusions

Statistical learners such as neural networks closely track the statistical regularities in their training sets. This process makes them vulnerable to adopting heuristics that are valid for frequent cases but fail on less frequent ones. We have investigated three such heuristics that we hypothesize NLI models are likely to learn. To evaluate whether NLI models do behave consistently with these heuristics, we have introduced the HANS dataset, on which models using these heuristics are guaranteed to fail. We find that four existing NLI models perform very poorly on HANS, suggesting that their high accuracies on NLI test sets may be due to the exploitation of invalid heuristics rather than deeper understanding of language. However, these models performed significantly better on both HANS and on a separate structure-dependent dataset when their training data was augmented with HANS-like examples. Overall, our results indicate that, despite the impressive accuracies of state-of-the-art models on standard evaluations, there is still much progress to be made and that targeted, challenging datasets, such as HANS, are important for determining whether models are learning what they are intended to learn.

Acknowledgments

We are grateful to Adam Poliak, Benjamin Van Durme, Samuel Bowman, the members of the JSALT General-Purpose Sentence Representation Learning team, and the members of the Johns Hopkins Computation and Psycholinguistics Lab for helpful comments, and to Brian Leonard for assistance with the Mechanical Turk experiment. Any errors remain our own.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1746891 and the 2018 Jelinek Summer Workshop on Speech and Language Technology (JSALT). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the JSALT workshop.

References

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2017. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. In *International Conference on Learning Representations*.

Aishwarya Agrawal, Dhruv Batra, and Devi Parikh. 2016. Analyzing the behavior of visual question answering models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1955–1960. Association for Computational Linguistics.

Thomas G. Bever. 1970. The cognitive basis for linguistic structures.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1466–1477. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668. Association for Computational Linguistics.

Kiel Christianson, Andrew Hollingworth, John F Halliwell, and Fernanda Ferreira. 2001. Thematic roles assigned along the garden path linger. *Cognitive Psychology*, 42(4):368–407.

Cleo Condoravdi, Dick Crouch, Valeria de Paiva, Reinhard Stolle, and Daniel G. Bobrow. 2003. Entailment, intensionality and text understanding. In *Proceedings of the HLT-NAACL 2003 Workshop on Text Meaning*.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment*, MLCW'05, pages 177–190, Berlin, Heidelberg. Springer-Verlag.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J. Gershman, and Noah D. Goodman. 2018. Evaluating compositionality in sentence embeddings. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 1596–1601, Madison, WI.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. 2018. Assessing composition in sentence vector representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1790–1801. Association for Computational Linguistics.

Lyn Frazier and Keith Rayner. 1982. Making and correcting errors during sentence comprehension: Eye movements in the analysis of structurally ambiguous sentences. *Cognitive Psychology*, 14(2):178–210.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A Deep Semantic Natural Language Processing Platform. In *Proceedings of the Workshop for NLP Open Source Software (NLP-OSS)*.

Atticus Geiger, Ignacio Cases, Lauri Karttunen, and Christopher Potts. 2018. Stress-testing neural models of natural language inference with multiply-quantified sentences. *arXiv preprint arXiv:1810.13033*.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI Systems with Sentences that Require Simple Lexical Inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing BERT's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. 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), pages 1195–1205. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112. Association for Computational Linguistics.

Juho Kim, Christopher Malon, and Asim Kadav. 2018. Teaching syntax by adversarial distraction. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 79–84. Association for Computational Linguistics.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535.

Bill MacCartney and Christopher D Manning. 2009. *Natural language inference*. Ph.D. thesis, Stanford University.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202. Association for Computational Linguistics.

R. Thomas McCoy, Robert Frank, and Tal Linzen. 2018. Revisiting the poverty of the stimulus: Hierarchical generalization without a hierarchical bias in recurrent neural networks. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, pages 2093–2098, Madison, WI.

R. Thomas McCoy and Tal Linzen. 2019. Non-entailed subsequences as a challenge for natural language inference. In *Proceedings of the Society for Computation in Linguistics*, volume 2.

R. Thomas McCoy, Tal Linzen, Ewan Dunbar, and Paul Smolensky. 2019. RNNs implicitly implement tensor-product representations. In *International Conference on Learning Representations*.

Yashar Mehdad, Alessandro Moschitti, and Fabio Massimo Zanzotto. 2010. Syntactic/semantic structures for textual entailment recognition. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1020–1028. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353. Association for Computational Linguistics.

Nikita Nangia and Samuel R. Bowman. 2019. Human vs. muppet: A conservative estimate of human performance on the GLUE benchmark.

Yixin Nie, Yicheng Wang, and Mohit Bansal. 2018. Analyzing compositionality-sensitivity of NLI models. *arXiv preprint arXiv:1811.07033*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255. Association for Computational Linguistics.

Ellie Pavlick and Chris Callison-Burch. 2016. Tense manages to predict implicative behavior in verbs. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2225–2229. Association for Computational Linguistics.

Adam Poliak, Aparajita Haldar, Rachel Rudinger, J. Edward Hu, Ellie Pavlick, Aaron Steven White, and Benjamin Van Durme. 2018a. Collecting diverse natural language inference problems for sentence representation evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 67–81. Association for Computational Linguistics.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. 2018b. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191. Association for Computational Linguistics.

Laura Rimell and Stephen Clark. 2010. Cambridge: Parser evaluation using textual entailment by grammatical relation comparison. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 268–271. Association for Computational Linguistics.

Rachel Rudinger, Aaron Steven White, and Benjamin Van Durme. 2018. Neural models of factuality. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 731–744. Association for Computational Linguistics.

Ivan Sanchez, Jeff Mitchell, and Sebastian Riedel. 2018. Behavior analysis of NLI models: Uncovering the influence of three factors on robustness. In *Proceedings of the 2018 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 1975–1985. Association for Computational Linguistics.

Whitney Tabor, Bruno Galantucci, and Daniel Richardson. 2004. Effects of merely local syntactic coherence on sentence processing. *Journal of Memory and Language*, 50(4):355–370.

Jianyu Wang, Zhishuai Zhang, Cihang Xie, Yuyin Zhou, Vittal Premachandran, Jun Zhu, Lingxi Xie, and Alan Yuille. 2018. Visual concepts and compositional voting. *Annals of Mathematical Sciences and Applications*, 3(1):151–188.

Noah Weber, Leena Shekhar, and Niranjan Balasubramanian. 2018. 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*, pages 24–27. Association for Computational Linguistics.

Aaron Steven White, Pushpendre Rastogi, Kevin Duh, and Benjamin Van Durme. 2017. Inference is everything: Recasting semantic resources into a unified evaluation framework. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 996–1005. Asian Federation of Natural Language Processing.

Aaron Steven White and Kyle Rawlins. 2018. The role of veridicality and factivity in clause selection. In *Proceedings of the 48th Annual Meeting of the North East Linguistic Society*.

Aaron Steven White, Rachel Rudinger, Kyle Rawlins, and Benjamin Van Durme. 2018. Lexicosyntactic inference in neural models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4717–4724. Association for Computational Linguistics.

Adina Williams, Andrew Drozdov, and Samuel R. Bowman. 2018a. Do latent tree learning models identify meaningful structure in sentences? *Transactions of the Association of Computational Linguistics*, 6:253–267.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018b. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

A MNLI examples that contradict the HANS heuristics

The sentences in (7) show examples from the MNLI training set that contradict the lexical overlap, subsequence, and constituent heuristics. The full set of all 261 contradicting examples in the MNLI training set may be viewed at https://github.com/tommccoy1/hans/blob/master/mnli_contradicting_examples.

(7) a. A subcategory of accuracy is consistency. $\nrightarrow$ Accuracy is a subcategory of consistency.
b. At the same time, top Enron executives were free to exercise their stock options, and some did. $\nrightarrow$ Top Enron executives were free to exercise.
c. She was chagrined at The Nation's recent publication of a column by conservative education activist Ron Unz arguing that liberal education reform has been an unmitigated failure. $\nrightarrow$ Liberal education reform has been an unmitigated failure.

B Templates

Tables 4, 5, and 6 contain the templates for the lexical overlap heuristic, the subsequence heuristic, and the constituent heuristic, respectively.

In some cases, a given template has multiple versions, such as one version where a noun phrase modifier attaches to the subject and another where the modifier attaches to the object. For clarity, we have only listed one version of each template here. The full list of templates can be viewed in the code on GitHub.[10]

C Fine-grained results

Table 7 shows the results by subcase for models trained on MNLI for the subcases where the correct answer is *entailment*. Table 8 shows the results by subcase for these models for the subcases where the correct answer is *non-entailment*.

D Results for models trained on MNLI with *neutral* and *contradiction* merged

Table 9 shows the results on HANS for models trained on MNLI with the labels *neutral* and *contradiction* merged in the training set into the single label *non-entailment*. The results are similar to the results obtained by merging the labels after training, with the models generally outputting *entailment* for all HANS examples, whether that was the correct answer or not.

[10] https://github.com/tommccoy1/hans

Subcase	Template	Example
Entailment: Untangling relative clauses	The N_1 who the N_2 V_1 V_2 the N_3 $\rightarrow$ The N_2 V_1 the N_1.	The athlete who the judges admired called the manager. $\rightarrow$ The judges admired the athlete.
Entailment: Sentences with PPs	The N_1 P the N_2 V the N_3 $\rightarrow$ The N_1 V the N_3	The tourists by the actor recommended the authors. $\rightarrow$ The tourists recommended the authors.
Entailment: Sentences with relative clauses	The N_1 that V_2 V_1 the N_2 $\rightarrow$ The N_1 V_1 the N_2	The actors that danced saw the author. $\rightarrow$ The actors saw the author.
Entailment: Conjunctions	The N_1 V the N_2 and the N_3 $\rightarrow$ The N_1 V the N_3	The secretaries encouraged the scientists and the actors. $\rightarrow$ The secretaries encouraged the actors.
Entailment: Passives	The N_1 were V by the N_2 $\rightarrow$ The N_1 V the N_2	The authors were supported by the tourists. $\rightarrow$ The tourists supported the authors.
Non-entailment: Subject-object swap	The N_1 V the N_2. $\nrightarrow$ The N_2 V the N_1.	The senators mentioned the artist. $\nrightarrow$ The artist mentioned the senators.
Non-entailment: Sentences with PPs	The N_1 P the N_2 V the N_3 $\nrightarrow$ The N_3 V the N_2	The judge behind the manager saw the doctors. $\nrightarrow$ The doctors saw the manager.
Non-entailment: Sentences with relative clauses	The N_1 V_1 the N_2 who the N_3 V_2 $\nrightarrow$ The N_2 V_1 the N_3	The actors advised the manager who the tourists saw. $\nrightarrow$ The manager advised the tourists.
Non-entailment: Conjunctions	The N_1 V the N_2 and the N_3 $\nrightarrow$ The N_2 V the N_3	The doctors advised the presidents and the tourists. $\nrightarrow$ The presidents advised the tourists.
Non-entailment: Passives	The N_1 were V by the N_2 $\nrightarrow$ The N_1 V the N_2	The senators were recommended by the managers. $\nrightarrow$ The senators recommended the managers.

Table 4: Templates for the lexical overlap heuristic

Subcase	Template	Example
Entailment: Conjunctions	The N_1 and the N_2 V the N_3 $\rightarrow$ The N_2 V the N_3	The actor and the professor mentioned the lawyer. $\rightarrow$ The professor mentioned the lawyer.
Entailment: Adjectives	Adj N_1 V the N_2 $\rightarrow$ N_1 V the N_2	Happy professors mentioned the lawyer. $\rightarrow$ Professors mentioned the lawyer.
Entailment: Understood argument	The N_1 V the N_2 $\rightarrow$ The N_1 V	The author read the book. $\rightarrow$ The author read.
Entailment: Relative clause on object	The N_1 V_1 the N_2 that V_2 the N_3 $\rightarrow$ The N_1 V_1 the N_2	The artists avoided the senators that thanked the tourists. $\rightarrow$ The artists avoided the senators.
Entailment: PP on object	The N_1 V the N_2 P the N_3 $\rightarrow$ The N_1 V the N_2	The authors supported the judges in front of the doctor. $\rightarrow$ The authors supported the judges.
Non-entailment: NP/S	The N_1 V_1 the N_2 V_2 the N_3 $\nrightarrow$ The N_1 V_1 the N_2	The managers heard the secretary encouraged the author. $\nrightarrow$ The managers heard the secretary.
Non-entailment: PP on subject	The N_1 P the N_2 V $\nrightarrow$ The N_2 V	The managers near the scientist resigned. $\nrightarrow$ The scientist resigned.
Non-entailment: Relative clause on subject	The N_1 that V_1 the N_2 V_2 the N_3 $\nrightarrow$ The N_2 V_2 the N_3	The secretary that admired the senator saw the actor. $\nrightarrow$ The senator saw the actor.
Non-entailment: MV/RR	The N_1 V_1 P the N_2 V_2 $\nrightarrow$ The N_1 V_1 P the N_2	The senators paid in the office danced. $\nrightarrow$ The senators paid in the office.
Non-entailment: NP/Z	P the N_1 V_1 the N_2 V_2 the N_3 $\nrightarrow$ The N_1 V_1 the N_2	Before the actors presented the professors advised the manager. $\nrightarrow$ The actors presented the professors.

Table 5: Templates for the subsequence heuristic

Subcase	Template	Example
Entailment: Embedded under preposition	P the N_1 V_1, the N_2 V_2 the N_3 $\rightarrow$ The N_1 V_1	Because the banker ran, the doctors saw the professors. $\rightarrow$ The banker ran.
Entailment: Outside embedded clause	P the N_1 V_1 the N_2, the N_3 V_2 the N_4 $\rightarrow$ The N_3 V_2 the N_4	Although the secretaries recommended the managers, the judges supported the scientist. $\rightarrow$ The judges supported the scientist.
Entailment: Embedded under verb	The N_1 V_1 that the N_2 V_2 $\rightarrow$ The N_2 V_2	The president remembered that the actors performed. $\rightarrow$ The actors performed.
Entailment: Conjunction	The N_1 V_1, and the N_2 V_2 the N_3. $\rightarrow$ The N_2 V_2 the N_3	The lawyer danced, and the judge supported the doctors. $\rightarrow$ The judge supported the doctors.
Entailment: Adverbs	Adv the N V $\rightarrow$ The N V	Certainly the lawyers resigned. $\rightarrow$ The lawyers resigned.
Non-entailment: Embedded under preposition	P the N_1 V_1, the N_2 V_2 the N_2 $\nrightarrow$ The N_1 V_1	Unless the senators ran, the professors recommended the doctor. $\nrightarrow$ The senators ran.
Non-entailment: Outside embedded clause	P the N_1 V_1 the N_2, the N_3 V_2 the N_4 $\nrightarrow$ The N_3 V_2 the N_4	Unless the authors saw the students, the doctors helped the bankers. $\nrightarrow$ The doctors helped the bankers.
Non-entailment: Embedded under verb	The N_1 V_1 that the N_2 V_2 the N_3 $\nrightarrow$ The N_2 V_2 the N_3	The tourists said that the lawyer saw the banker. $\nrightarrow$ The lawyer saw the banker.
Non-entailment: Disjunction	The N_1 V_1, or the N_2 V_2 the N_3 $\nrightarrow$ The N_2 V_2 the N_3	The judges resigned, or the athletes mentioned the author. $\nrightarrow$ The athletes mentioned the author.
Non-entailment: Adverbs	Adv the N_1 V the N_2 $\nrightarrow$ The N_1 V the N_2	Probably the artists saw the authors. $\nrightarrow$ The artists saw the authors.

Table 6: Templates for the constituent heuristic

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Untangling relative clauses	0.97	0.95	0.88	0.98
	The athlete who the judges saw called the manager. → The judges saw the athlete.				
	Sentences with PPs	1.00	1.00	1.00	1.00
	The tourists by the actor called the authors. → The tourists called the authors.				
	Sentences with relative clauses	0.98	0.97	0.97	0.99
	The actors that danced encouraged the author. → The actors encouraged the author.				
	Conjunctions	1.00	1.00	1.00	0.77
	The secretaries saw the scientists and the actors. → The secretaries saw the actors.				
	Passives	1.00	1.00	0.95	1.00
	The authors were supported by the tourists. → The tourists supported the authors.				
Subsequence	Conjunctions	1.00	1.00	1.00	0.98
	The actor and the professor shouted. → The professor shouted.				
	Adjectives	1.00	1.00	1.00	1.00
	Happy professors mentioned the lawyer. → Professors mentioned the lawyer.				
	Understood argument	1.00	1.00	0.84	1.00
	The author read the book. → The author read.				
	Relative clause on object	0.98	0.99	0.95	0.99
	The artists avoided the actors that performed. → The artists avoided the actors.				
	PP on object	1.00	1.00	1.00	1.00
	The authors called the judges near the doctor. → The authors called the judges.				
Constituent	Embedded under preposition	0.99	0.99	0.85	1.00
	Because the banker ran, the doctors saw the professors. → The banker ran.				
	Outside embedded clause	0.94	1.00	0.95	1.00
	Although the secretaries slept, the judges danced. → The judges danced.				
	Embedded under verb	0.92	0.94	0.99	0.99
	The president remembered that the actors performed. → The actors performed.				
	Conjunction	0.99	1.00	0.89	1.00
	The lawyer danced, and the judge supported the doctors. → The lawyer danced.				
	Adverbs	1.00	1.00	0.98	1.00
	Certainly the lawyers advised the manager. → The lawyers advised the manager.				

Table 7: Results for the subcases where the correct label is *entailment*.

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Subject-object swap	0.00	0.00	0.03	0.00
	The senators mentioned the artist. ↛ The artist mentioned the senators.				
	Sentences with PPs	0.00	0.00	0.01	0.25
	The judge behind the manager saw the doctors. ↛ The doctors saw the manager.				
	Sentences with relative clauses	0.04	0.04	0.06	0.18
	The actors called the banker who the tourists saw. ↛ The banker called the tourists.				
	Conjunctions	0.00	0.00	0.01	0.39
	The doctors saw the presidents and the tourists. ↛ The presidents saw the tourists.				
	Passives	0.00	0.00	0.00	0.00
	The senators were helped by the managers. ↛ The senators helped the managers.				
Subsequence	NP/S	0.04	0.02	0.09	0.02
	The managers heard the secretary resigned. ↛ The managers heard the secretary.				
	PP on subject	0.00	0.00	0.00	0.06
	The managers near the scientist shouted. ↛ The scientist shouted.				
	Relative clause on subject	0.03	0.04	0.05	0.01
	The secretary that admired the senator saw the actor. ↛ The senator saw the actor.				
	MV/RR	0.04	0.03	0.03	0.00
	The senators paid in the office danced. ↛ The senators paid in the office.				
	NP/Z	0.02	0.01	0.11	0.10
	Before the actors presented the doctors arrived. ↛ The actors presented the doctors.				
Constituent	Embedded under preposition	0.14	0.02	0.29	0.50
	Unless the senators ran, the professors recommended the doctor. ↛ The senators ran.				
	Outside embedded clause	0.01	0.00	0.02	0.00
	Unless the authors saw the students, the doctors resigned. ↛ The doctors resigned.				
	Embedded under verb	0.00	0.00	0.01	0.22
	The tourists said that the lawyer saw the banker. ↛ The lawyer saw the banker.				
	Disjunction	0.01	0.03	0.20	0.01
	The judges resigned, or the athletes saw the author. ↛ The athletes saw the author.				
	Adverbs	0.00	0.00	0.00	0.08
	Probably the artists saw the authors. ↛ The artists saw the authors.				

Table 8: Results for the subcases where the correct label is *non-entailment*.

		Correct: *Entailment*			Correct: *Non-entailment*		
Model	Model class	Lexical	Subseq.	Const.	Lexical	Subseq.	Const.
DA	Bag-of-words	1.00	1.00	0.98	0.00	0.00	0.03
ESIM	RNN	0.99	1.00	1.00	0.00	0.01	0.00
SPINN	TreeRNN	0.94	0.96	0.93	0.06	0.14	0.11
BERT	Transformer	0.98	1.00	0.99	0.04	0.02	0.20

Table 9: Results for models trained on MNLI with *neutral* and *contradiction* merged into a single label, *non-entailment*.

E Results with augmented training with some subcases withheld

For each model, we ran five experiments, each one having 6 of the 30 subcases withheld. Each trained model was then evaluated on the categories that had been withheld from it. The results of these experiments are in Tables 10, 11, 12, 13 and 14.

F Human experiments

To obtain human results, we used Amazon Mechanical Turk. We subdivided HANS into 114 different categories of examples, covering all possible variations of the template used to generate the example and the specific word around which the template was built. For example, for the constituent heuristic subcase of clauses embedded under verbs (e.g. *The doctor believed the lawyer danced* $\nrightarrow$ *The lawyer danced*), each possible verb under which the clause could be embedded (e.g. *believed*, *thought*, or *assumed*) counted as a different category.

For each of these 114 categories, we chose 20 examples from HANS and obtained judgments from 5 human participants for each of those 20 examples. Each participant provided judgments for 57 examples plus 10 controls (67 stimuli total) and was paid $2.00. The controls consisted of 5 examples where the premise and hypothesis were the same (e.g. *The doctor saw the lawyer* $\rightarrow$ *The doctor saw the lawyer*) and 5 examples of simple negation (e.g. *The doctor saw the lawyer* $\nrightarrow$ *The doctor did not see the lawyer*). For analyzing the data, we discarded any participants who answered any of these controls incorrectly; this led to 95 participants being retained and 105 being rejected (participants were still paid regardless of whether they were retained or filtered out). On average, each participant spent 6.5 seconds per example; the participants we retained spent 8.9 seconds per example, while the participants we discarded spent 4.2 seconds per example. The total amount of time from a participant accepting the experiment to completing the experiment averaged 17.6 minutes. This included 9.1 minutes answering the prompts (6.4 minutes for discarded participants and 12.1 minutes for retained participants) and roughly one minute spent between prompts (1 second after each prompt). The remaining time was spent reading the consent form, reading the instructions, or waiting to start (Mechanical Turk participants often wait several minutes between accepting an experiment and beginning the experiment).

The expert annotators were three native English speakers who had a background in linguistics but who had not heard about this project before providing judgments. Two of them were graduate students and one was a postdoctoral researcher. Each expert annotator labeled 124 examples (one example from each of the 114 categories, plus 10 controls).

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Subject-object swap *The senators mentioned the artist.* $\nrightarrow$ *The artist mentioned the senators.*	0.01	1.00	1.00	1.00
Lexical overlap	Untangling relative clauses *The athlete who the judges saw called the manager.* $\rightarrow$ *The judges saw the athlete.*	0.34	0.23	0.23	0.20
Subsequence	NP/S *The managers heard the secretary resigned.* $\nrightarrow$ *The managers heard the secretary.*	0.27	0.00	0.00	0.10
Subsequence	Conjunctions *The actor and the professor shouted.* $\rightarrow$ *The professor shouted.*	0.49	0.38	0.38	0.38
Constituent	Embedded under preposition *Unless the senators ran, the professors recommended the doctor.* $\nrightarrow$ *The senators ran.*	0.51	0.51	0.51	1.00
Constituent	Embedded under preposition *Because the banker ran, the doctors saw the professors.* $\rightarrow$ *The banker ran.*	1.00	0.06	1.00	0.03

Table 10: Accuracies for models trained on MNLI augmented with most HANS example categories except withholding the categories in this table (experiment 1/5 for the withheld category investigation).

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Sentences with PPs *The judge behind the manager saw the doctors.* $\nrightarrow$ *The doctors saw the manager.*	0.00	0.96	0.71	0.97
Lexical overlap	Sentences with PPs *The tourists by the actor called the authors.* $\rightarrow$ *The tourists called the authors.*	1.00	1.00	0.94	1.00
Subsequence	PP on subject *The managers near the scientist shouted.* $\nrightarrow$ *The scientist shouted.*	0.00	0.07	0.57	0.39
Subsequence	Adjectives *Happy professors mentioned the lawyer.* $\rightarrow$ *Professors mentioned the lawyer.*	0.71	0.99	0.64	1.00
Constituent	Outside embedded clause *Unless the authors saw the students, the doctors resigned.* $\nrightarrow$ *The doctors resigned.*	0.78	1.00	1.00	0.17
Constituent	Outside embedded clause *Although the secretaries slept, the judges danced.* $\rightarrow$ *The judges danced.*	0.78	0.78	0.78	0.97

Table 11: Accuracies for models trained on MNLI augmented with most HANS example categories except withholding the categories in this table (experiment 2/5 for the withheld category investigation).

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Sentences with relative clauses	0.00	0.04	0.02	0.84
	The actors called the banker who the tourists saw. $\nrightarrow$ *The banker called the tourists.*				
Lexical overlap	Sentences with relative clauses	1.00	0.97	1.00	1.00
	The actors that danced encouraged the author. $\rightarrow$ *The actors encouraged the author.*				
Subsequence	Relative clause on subject	0.00	0.04	0.00	0.93
	The secretary that admired the senator saw the actor. $\nrightarrow$ *The senator saw the actor.*				
Subsequence	Understood argument	0.28	1.00	0.81	0.94
	The author read the book. $\rightarrow$ *The author read.*				
Constituent	Embedded under verb	0.00	0.00	0.05	0.98
	The tourists said that the lawyer saw the banker. $\nrightarrow$ *The lawyer saw the banker.*				
Constituent	Embedded under verb	1.00	0.94	0.98	0.43
	The president remembered that the actors performed. $\rightarrow$ *The actors performed.*				

Table 12: Accuracies for models trained on MNLI augmented with most HANS example categories except withholding the categories in this table (experiment 3/5 for the withheld category investigation).

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Passives	0.00	0.00	0.00	0.00
	The senators were helped by the managers. $\nrightarrow$ *The senators helped the managers.*				
Lexical overlap	Conjunctions	0.05	0.51	0.52	1.00
	The secretaries saw the scientists and the actors. $\rightarrow$ *The secretaries saw the actors.*				
Subsequence	MV/RR	0.76	0.44	0.32	0.07
	The senators paid in the office danced. $\nrightarrow$ *The senators paid in the office.*				
Subsequence	Relative clause on object	0.72	1.00	0.99	0.99
	The artists avoided the actors that performed. $\rightarrow$ *The artists avoided the actors.*				
Constituent	Disjunction	0.11	0.29	0.51	0.44
	The judges resigned, or the athletes saw the author. $\nrightarrow$ *The athletes saw the author.*				
Constituent	Conjunction	0.99	1.00	0.74	1.00
	The lawyer danced, and the judge supported the doctors. $\rightarrow$ *The lawyer danced.*				

Table 13: Accuracies for models trained on MNLI augmented with most HANS example categories except withholding the categories in this table (experiment 4/5 for the withheld category investigation).

Heuristic	Subcase	DA	ESIM	SPINN	BERT
Lexical overlap	Conjunctions	0.00	0.44	0.00	0.08
	The doctors saw the presidents and the tourists. $\nrightarrow$ *The presidents saw the tourists.*				
Lexical overlap	Passives	0.00	0.00	0.00	0.00
	The authors were supported by the tourists. $\rightarrow$ *The tourists supported the authors.*				
Subsequence	NP/Z	0.00	0.10	0.18	0.57
	Before the actors presented the doctors arrived. $\nrightarrow$ *The actors presented the doctors.*				
Subsequence	PP on object	0.04	0.76	0.04	0.98
	The authors called the judges near the doctor. $\rightarrow$ *The authors called the judges.*				
Constituent	Adverbs	0.76	0.33	0.20	0.84
	Probably the artists saw the authors. $\nrightarrow$ *The artists saw the authors.*				
Constituent	Adverbs	0.66	1.00	0.59	0.96
	Certainly the lawyers advised the manager. $\rightarrow$ *The lawyers advised the manager.*				

Table 14: Accuracies for models trained on MNLI augmented with most HANS example categories except withholding the categories in this table (experiment 5/5 for the withheld category investigation).

RNNs IMPLICITLY IMPLEMENT TENSOR-PRODUCT REPRESENTATIONS

R. Thomas McCoy,[1] Tal Linzen,[1] Ewan Dunbar,[2] & Paul Smolensky[3,1]
[1]Department of Cognitive Science, Johns Hopkins University
[2]Laboratoire de Linguistique Formelle, CNRS - Université Paris Diderot - Sorbonne Paris Cité
[3]Microsoft Research AI, Redmond, WA USA
tom.mccoy@jhu.edu, tal.linzen@jhu.edu,
ewan.dunbar@univ-paris-diderot.fr, smolensky@jhu.edu

ABSTRACT

Recurrent neural networks (RNNs) can learn continuous vector representations of symbolic structures such as sequences and sentences; these representations often exhibit linear regularities (analogies). Such regularities motivate our hypothesis that RNNs that show such regularities implicitly compile symbolic structures into tensor product representations (TPRs; Smolensky, 1990), which additively combine tensor products of vectors representing roles (e.g., sequence positions) and vectors representing fillers (e.g., particular words). To test this hypothesis, we introduce Tensor Product Decomposition Networks (TPDNs), which use TPRs to approximate existing vector representations. We demonstrate using synthetic data that TPDNs can successfully approximate linear and tree-based RNN autoencoder representations, suggesting that these representations exhibit interpretable compositional structure; we explore the settings that lead RNNs to induce such structure-sensitive representations. By contrast, further TPDN experiments show that the representations of four models trained to encode naturally-occurring sentences can be largely approximated with a bag of words, with only marginal improvements from more sophisticated structures. We conclude that TPDNs provide a powerful method for interpreting vector representations, and that standard RNNs can induce compositional sequence representations that are remarkably well approximated by TPRs; at the same time, existing training tasks for sentence representation learning may not be sufficient for inducing robust structural representations.

1 INTRODUCTION

Compositional symbolic representations are widely held to be necessary for intelligence (Newell, 1980; Fodor & Pylyshyn, 1988), particularly in the domain of language (Montague, 1974). However, neural networks have shown great success in natural language processing despite using continuous vector representations rather than explicit symbolic structures. How can these continuous representations yield such success in a domain traditionally believed to require symbol manipulation?

One possible answer is that neural network representations implicitly encode compositional structure. This hypothesis is supported by the spatial relationships between such vector representations, which have been argued to display geometric regularities that parallel plausible symbolic structures of the elements being represented (Mikolov et al. 2013; see Figure 1).

Analogical relationships such as those in Figure 1 are special cases of linearity properties shared by several methods developed in the 1990s for designing compositional vector embeddings of symbolic structures. The most general of these is **tensor product representations** (TPRs; Smolensky 1990). Symbolic structures are first decomposed into **filler-role bindings**; for example, to represent the sequence $[5, 2, 4]$, the filler 5 may be bound to the role of *first element*, the filler 2 may be bound to the role of *second element*, and so on. Each filler f_i and — crucially — each **role** r_i has a vector embedding; these two vectors are combined using their tensor product $f_i \otimes r_i$, and these tensor products are summed to produce the representation of the sequence: $\sum f_i \otimes r_i$. This linear combination can predict the linear relations between sequence representations illustrated in Figure 1.

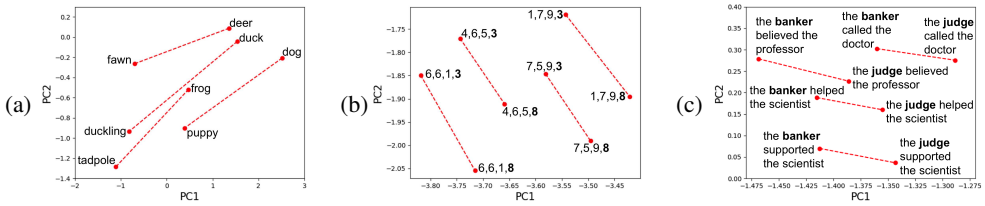


Figure 1: Plots of the first two principal components of (a) word embeddings (Pennington et al., 2014), (b) digit-sequence embeddings learned by an autoencoder (Section 2), and (c) sentences (InferSent: Conneau et al. 2017). All demonstrate systematicity in the learned vector spaces.

In this article, we test the hypothesis that vector representations of sequences can be approximated as a sum of filler-role bindings, as in TPRs. We introduce the Tensor Product Decomposition Network (TPDN) which takes a set of continuous vector representations to be analyzed and learns filler and role embeddings that best predict those vectors, given a particular hypothesis for the relevant set of roles (e.g., sequence indexes or structural positions in a parse tree).

To derive structure-sensitive representations, in Section 2 we look at a task driven by structure, not content: autoencoding of sequences of meaningless symbols, denoted by digits. The focus here is on sequential structure, although we also devise a version of the task that uses tree structure. For the representations learned by these autoencoders, TPDNs find excellent approximations that are TPRs.

In Section 3, we turn to sentence-embedding models from the contemporary literature. It is an open question how structure-sensitive these representations are; to the degree that they are structure-sensitive, our hypothesis is that they can be approximated by TPRs. Here, TPDNs find less accurate approximations, but they also show that a TPR equivalent to a bag-of-words already provides a reasonable approximation; these results suggest that these sentence representations are not robustly structure-sensitive. We therefore return to synthetic data in Section 4, exploring which architectures and training tasks are likely to lead RNNs to induce structure-sensitive representations.

To summarize the contributions of this work, TPDNs provide a powerful method for interpreting vector representations, shedding light on hard-to-understand neural architectures. We show that standard RNNs can induce compositional representations that are remarkably well approximated by TPRs and that the nature of these representations depends, in intrepretable ways, on the architecture and training task. Combined with our finding that standard sentence encoders do not seem to learn robust representations of structure, these findings suggest that more structured architectures or more structure-dependent training tasks could improve the compositional capabilities of existing models.

1.1 The Tensor Product Decomposition Network

The Tensor Product Decomposition Network (TPDN), depicted in Figure 2c, learns a TPR that best approximates an existing set of vector encodings. While TPDNs can be applied to any structured space, including embeddings of images or words, this work focuses on applying TPDNs to sequences. The model is given a hypothesized role scheme and the dimensionalities of the filler and role embeddings. The elements of each sequence are assumed to be the fillers in that sequence's representation; for example, if the hypothesized roles are indexes counting from the end of the sequence, then the hypothesized filler-role pairs for $[5, 2, 4]$ would be (4:*last*, 2:*second-to-last*, 5:*third-to-last*).

The model then learns embeddings for these fillers and roles that minimize the distance between the TPRs generated from these embeddings and the existing encodings of the sequences. Before the comparison is performed, the tensor product (which is a matrix) is flattened into a vector, and a linear transformation M is applied (see Appendix B for an ablation study showing that this transformation, which was not a part of the original TPR proposal, is necessary). The overall function computed by the architecture is thus $M(\textit{flatten}(\sum_i r_i \otimes f_i))$.

PyTorch code for the TPDN model is available on GitHub,[1] along with an interactive demo.[2]

[1] https://github.com/tommccoy1/tpdn

[2] https://tommccoy1.github.io/tpdn/tpr_demo.html

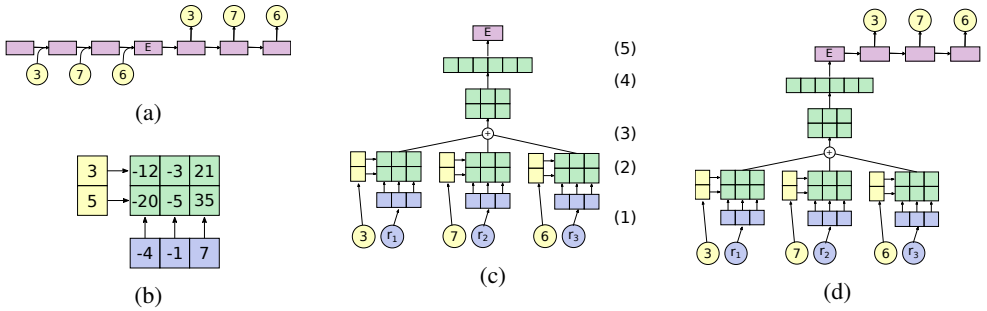


Figure 2: **(a)** A unidirectional sequence-to-sequence autoencoder. **(b)** The tensor product operation. **(c)** A TPDN trained to approximate the encoding E from the autoencoder: (1) The fillers and roles are embedded. (2) The fillers and roles are bound together using the tensor product. (3) The tensor products are summed. (4) The sum is flattened into a vector by concatenating the rows. (5) A linear transformation is applied to get the final encoding. **(d)** The architecture for evaluation: using the original autoencoder's decoder with the trained TPDN as the encoder.

2 Approximating RNN autoencoder representations

To establish the effectiveness of the TPDN at uncovering the structural representations used by RNNs, we first apply the TPDN to sequence-to-sequence networks trained on an autoencoding objective: they are expected to encode a sequence of digits and then decode that encoding to reproduce the same sequence (Figure 2a). In addition to testing the TPDN, this experiment also addresses a scientific question: do different architectures (specifically, unidirectional, bidirectional, and tree-based sequence-to-sequence models) induce different representations?

2.1 Experimental setup

Digit sequences: The sequences consisted of the digits from 0 to 9. We randomly generated 50,000 unique sequences with lengths ranging from 1 to 6 inclusive and averaging 5.2; these sequences were divided into 40,000 training sequences, 5,000 development sequences, and 5,000 test sequences.

Architectures: For all sequence-to-sequence networks, we used gated recurrent units (GRUs, Cho et al. (2014)) as the recurrent units. We considered three encoder-decoder architectures: unidirectional, bidirectional, and tree-based.[3] The unidirectional encoders and decoders follow the setup of Sutskever et al. (2014): the encoder is fed the input elements one at a time, left to right, updating its hidden state after each element. The decoder then produces the output sequence using the final hidden state of the encoder as its input. The bidirectional encoder combines left-to-right and right-to-left unidirectional encoders (Schuster & Paliwal, 1997); for symmetry, we also create a bidirectional decoder, which has both a left-to-right and a right-to-left unidirectional decoder whose hidden states are concatenated to form bidirectional hidden states from which output predictions are made. Our final topology is tree-based RNNs (Pollack, 1990; Socher et al., 2010), specifically the Tree-GRU encoder of Chen et al. (2017) and the tree decoder of Chen et al. (2018). These architectures require a tree structure as part of their input; we generated a tree for each sequence using a deterministic algorithm that groups digits based on their values (see Appendix C). To control for initialization effects, we trained five instances of each architecture with different random initializations.

Role schemes: We consider 6 possible methods that networks might use to represent the roles of specific digits within a sequence; see Figure 3a for examples of these role schemes.

1. **Left-to-right:** Each digit's role is its index in the sequence, counting from left to right.
2. **Right-to-left:** Each digit's role is its index in the sequence, counting from right to left.
3. **Bidirectional:** Each digit's role is an ordered pair containing its left-to-right index and its right-to-left index (compare human representations of spelling, Fischer-Baum et al. 2010).
4. **Wickelroles:** Each digit's role is the digit before it and the digit after it (Wickelgren, 1969).

[3]For this experiment, the encoder and decoder always matched in type.

5. **Tree positions:** Each digit's role is its position in a tree, such as RRL (left child of right child of right child of root). The tree structures are given by the algorithm in Appendix C.
6. **Bag-of-words:** All digits have the same role. We call this a *bag-of-words* because it represents which digits ("words") are present and in what quantities, but ignores their positions.

Hypothesis: We hypothesize that RNN autoencoders will learn to use role representations that parallel their architectures: left-to-right roles for a unidirectional network, bidirectional roles for a bidirectional network, and tree-position roles for a tree-based network.

Evaluation: We evaluate how well a given sequence-to-sequence network can be approximated by a TPR with a particular role scheme as follows. First, we train a TPDN with the role scheme in question (Section 1.1). Then, we take the original encoder/decoder network and substitute the fitted TPDN for its encoder (Figure 2d). We do not conduct any additional training upon this hybrid network; the decoder retains exactly the weights it learned in association with the original encoder, while the TPDN retains exactly the weights it learned for approximating the original encoder (including the weights on the final linear layer). We then compute the accuracy of the resulting hybrid network; we call this metric the **substitution accuracy**. High substitution accuracy indicates that the TPDN has approximated the encoder well enough for the decoder to handle the resulting vectors.

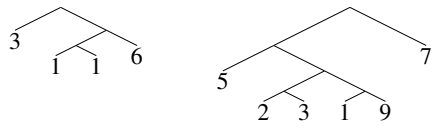
2.2 Results

Performance of seq2seq networks: The unidirectional and tree-based architectures both performed the training task nearly perfectly, with accuracies of 0.999 and 0.989 (averaged across five runs). Accuracy was lower (0.834) for the bidirectional architecture; this might mean that the hidden size of 60 becomes too small when divided into two 30-dimensional halves, one half for each direction.

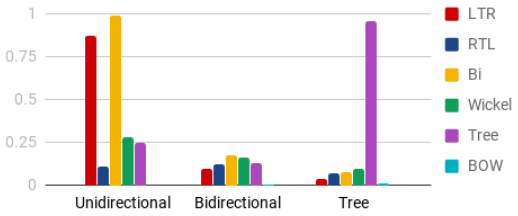
Quality of TPDN approximation: For each of the six role schemes, we fitted a TPDN to the vectors generated by the trained encoder, and evaluated it using substitution accuracy (Section 2.1). The results, in Figure 3c, show that different architectures do use different representations to solve the task. The tree-based autoencoder can be well-approximated using tree-position roles but not using any of the other role schemes. By contrast, the unidirectional architecture is approximated very closely (with a substitution accuracy of over 0.99 averaged across five runs) by bidirectional roles. Left-to-right roles are also fairly successful (accuracy = 0.87), and right-to-left roles are decidedly unsuccessful (accuracy = 0.11). This asymmetry suggests that the unidirectional network

	3	1	1	6	5	2	3	1	9	7
Left-to-right	0	1	2	3	0	1	2	3	4	5
Right-to-left	3	2	1	0	5	4	3	2	1	0
Bidirectional	(0, 3)	(1, 2)	(2, 1)	(3, 0)	(0, 5)	(1, 4)	(2, 3)	(3, 2)	(4, 1)	(5, 0)
Wickelroles	#_1	3_1	1_6	1_#	#_2	5_3	2_1	3_9	1_7	9_#
Tree	L	RLL	RLR	RR	LL	LRLL	LRLR	LRRL	LRRR	R
Bag of words	r_0	r_0	r_0	r_0	r_0	r_0	r_0	r_0	r_0	r_0

(a)



(b)



(c)

Figure 3: (a) The filler-role bindings assigned by the six role schemes to two sequences, 3116 and 523197. Roles not shown are assigned the null filler. (b) The trees used to assign tree roles to these sequences. (c) Substitution accuracy for three architectures at the autoencoding task with six role schemes. Each bar represents an average across five random initializations.

uses *mildly bidirectional* roles: while it is best approximated by bidirectional roles, it strongly favors one direction over the other. Though the model uses bidirectional roles, then, roles with the same left-to-right position (e.g. (2,3), (2,4), and (2,5)) can be collapsed without much loss of accuracy.

Finally, the bidirectional architecture is not approximated well by any of the role schemes we investigated. It may be implementing a role scheme we did not consider, or a structure-encoding scheme other than TPR. Alternately, it might simply not have adopted any robust method for representing sequence structure; this could explain why its accuracy on the training task was relatively low (0.83).

3 ENCODINGS OF NATURALLY-OCCURRING SENTENCES

Will the TPDN's success with digit-sequence autoencoders extend to models trained on naturally occurring data? We explore this question using sentence representations from four models: InferSent (Conneau et al., 2017), a BiLSTM trained on the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015); Skip-thought (Kiros et al., 2015), an LSTM trained to predict the sentence before or after a given sentence; the Stanford sentiment model (SST) (Socher et al., 2013), a tree-based recursive neural tensor network trained to predict movie review sentiment; and SPINN (Bowman et al., 2016), a tree-based RNN trained on SNLI. More model details are in Appendix E.

3.1 TPDN APPROXIMATION

We now fit TPDNs to these four sentence encoding models. We experiment with all of the role schemes used in Section 2 except for Wickelroles; for sentence representations, the vocabulary size $|V|$ is so large that the Wickelrole scheme, which requires $|V|^2$ distinct roles, becomes intractable.

Preliminary experiments showed that the TPDN performed poorly when learning the filler embeddings from scratch, so we used pretrained word embeddings; for each model, we use the word embeddings used by that model. We fine-tuned the embeddings with a linear transformation on top of the word embedding layer (though the embeddings themselves remain fixed). Thus, what the model has to learn are: the role embeddings, the linear transformation to apply to the fixed filler embeddings, and the final linear transformation applied to the sum of the filler/role bindings.

We train TPDNs on the sentence embeddings that each model generates for all SNLI premise sentences (Bowman et al., 2015). For other training details see Appendix E. Table 1a shows the mean squared errors (MSEs) for various role schemes. In general, the MSEs show only small differences between role schemes, except that tree-position roles do noticeably outperform other role schemes for SST. Notably, bag-of-words roles perform nearly as well as the other role schemes, in stark contrast to the poor performance of bag-of-words roles in Section 2. MSE is useful for comparing models but is less useful for assessing absolute performance since the exact value of this error is not very interpretable. In the next section, we use downstream tasks for a more interpretable evaluation.

3.2 PERFORMANCE ON DOWNSTREAM TASKS

Tasks: We assess how the tensor product approximations compare to the models they approximate at four tasks that are widely accepted for evaluating sentence embeddings: (1) Stanford Sentiment Treebank (SST), rating the sentiment of movie reviews (Socher et al., 2013); (2) Microsoft Research

(a)

	LTR	RTL	Bi	Tree	BOW
InferSent	0.17	0.18	0.17	**0.16**	0.19
Skip-thought	0.45	0.46	0.47	**0.42**	0.45
SST	0.24	0.26	0.26	**0.17**	0.27
SPINN	0.22	0.23	0.21	**0.18**	0.25

(b)

	LTR	RTL	Bi	Tree	BOW
InferSent	0.35	0.34	**0.29**	0.35	0.40
Skip-thought	0.34	0.37	**0.24**	0.34	0.51
SST	0.27	0.32	**0.25**	0.26	0.34
SPINN	0.49	0.53	**0.44**	0.49	0.56

Table 1: (a) MSEs of TPDN approximations of sentence encodings (normalized by dividing by the MSE from training the TPDN on random vectors, to allow comparisons across models). (b) Performance of the sentence encoding models on our role-diagnostic analogies. Numbers indicate Euclidean distances (normalized by dividing by the average distance between vectors in the analogy set). Each column contains the average over all analogies diagnostic of the role heading that column.

(a)

Model	LTR	RTL	Bi	Tree	BOW
InferSent	**0.79**	**0.79**	0.78	0.78	0.77
Skip-thought	0.53	0.52	0.46	0.50	**0.58**
SST	**0.83**	0.82	0.82	0.82	0.81
SPINN	0.73	0.75	0.75	**0.76**	0.74

(b)

Model	LTR	RTL	Bi	Tree	BOW
InferSent	**0.77**	0.74	**0.77**	**0.77**	0.71
Skip-thought	**0.37**	**0.37**	0.36	0.36	**0.37**
SST	0.48	0.51	0.49	**0.67**	0.49
SPINN	0.72	0.72	0.73	**0.76**	0.58

Table 2: The proportion of test examples on which a classifier trained on sentence encodings gave the same predictions for the original encodings and for their TPDN approximations. (a) shows the average of these proportions across SST, MRPC, and STS-B, while (b) shows only SNLI. (For including STS-B in (a), we linearly shift its values to be in the same range as the other tasks' results).

Paraphrase Corpus (MRPC), classifying whether two sentences paraphrase each other (Dolan et al., 2004); (3) Semantic Textual Similarity Benchmark (STS-B), labeling how similar two sentences are (Cer et al., 2017); and (4) Stanford Natural Language Inference (SNLI), determining if one sentence entails a second sentence, contradicts the second sentence, or neither (Bowman et al., 2015).

Evaluation: We use SentEval (Conneau & Kiela, 2018) to train a classifier for each task on the original encodings produced by the sentence encoding model. We freeze this classifier and use it to classify the vectors generated by the TPDN. We then measure what proportion of the classifier's predictions for the approximation match its predictions for the original sentence encodings.[4]

Results: For all tasks besides SNLI, we found no marked difference between bag-of-words roles and other role schemes (Table 2a). For SNLI, we did see instances where other role schemes outperformed bag-of-words (Table 2b). Within the SNLI results, both tree-based models (SST and SPINN) are best approximated with tree-based roles. InferSent is better approximated with structural roles than with bag-of-words roles, but all structural role schemes perform similarly. Finally, Skip-thought cannot be approximated well with any role scheme we considered. It is unclear why Skip-thought has lower results than the other models. Overall, even for SNLI, bag-of-words roles provide a fairly good approximation, with structured roles yielding rather modest improvements.

Based on these results, we hypothesize that these models' representations can be characterized as a bag-of-words representation plus some incomplete structural information that is not always encoded. This explanation is consistent with the fact that bag-of-words roles yield a strong but imperfect approximation for the sentence embedding models. However, this is simply a conjecture; it is possible that these models do use a robust, systematic structural representation that either involves a role scheme we did not test or that cannot be characterized as a tensor product representation at all.

3.3 ANALOGIES

We now complement the TPDN tests with sentence analogies. By comparing pairs of minimally different sentences, analogies might illuminate representational details that are difficult to discern in individual sentences. We construct sentence-based analogies that should hold only under certain role schemes, such as the following analogy (expressed as an equation as in Mikolov et al. 2013):

$$\textbf{I see now} - \textbf{I see} = \textbf{you know now} - \textbf{you know} \tag{1}$$

A left-to-right role scheme makes (1) equivalent to (2) (f:r denotes the binding of filler f to role r):

$$\text{(I:0 + see:1 + now:2) – (I:0 + see:1) = (you:0 + know:1 + now:2) – (you:0 + know:1)} \tag{2}$$

In (2), both sides reduce to now:2, so (1) holds for representations using left-to-right roles. However, if (2) instead used right-to-left roles, it would not reduce in any clean way, so (1) would not hold. We construct a dataset of such role-diagnostic analogies, where each analogy should only hold for certain role schemes. For example, (1) works for left-to-right roles or bag-of-words roles, but not the other role schemes. The analogies use a vocabulary based on Ettinger et al. (2018) to ensure plausibility of the constructed sentences. For each analogy, we create 4 equations, one isolating

[4]We also train SentEval classifiers on top of the TPDN instead of the original model; see Appendix F for these results. In general, for all models besides Skip-thought, the TPDN approximations perform nearly as well as the original models, and in some cases the approximations even outperform the originals.

each of the four terms (e.g. **I see = I see now – you know now + you know**). We then compute the Euclidean distance between the two sides of each equation using each model's encodings.

The results are in Table 1b. InferSent, Skip-thought, and SPINN all show results most consistent with bidirectional roles, while SST shows results most consistent with tree-based or bidirectional roles. The bag-of-words column shows poor performance by all models, indicating that in controlled enough settings these models can be shown to have some more structured behavior even though evaluation on examples from applied tasks does not clearly bring out that structure. These analogies thus provide independent evidence for our conclusions from the TPDN analysis: these models have a weak notion of structure, but that structure is largely drowned out by the non-structure-sensitive, bag-of-words aspects of their representations. However, the other possible explanations mentioned above—namely, the possibilities that the models use alternate role schemes that we did not test or that they use some structural encoding other than tensor product representation—still remain.

4 When do RNNs learn compositional representations?

The previous section suggested that all sentence models surveyed did not robustly encode structure and could even be approximated fairly well with a bag of words. Motivated by this finding, we now investigate how aspects of training can encourage or discourage compositionality in learned representations. To increase interpretability, we return to the setting (from Section 2) of operating over digit sequences. We investigate two aspects of training: the architecture and the training task.

Teasing apart the contribution of the encoder and decoder: In Section 2, we investigated autoencoders whose encoder and decoder had the same topology (unidirectional, bidirectional, or tree-based). To test how each of the two components contributes to the learned representation, we now expand the investigation to include networks where the encoder and decoder differ. We crossed all three encoder types with all three decoder types (nine architectures in total). The results are in Table 7 in Appendix D. The decoder largely dictates what roles are learned: models with unidirectional decoders prefer mildly bidirectional roles, models with bidirectional decoders fail to be well-approximated by any role scheme, and models with tree-based decoders are best approximated by tree-based roles. However, the encoder still has some effect: in the tree/uni and tree/bi models, the tree-position roles perform better than they do for the other models with the same decoders. Though work on novel architectures often focuses on the encoder, this finding suggests that focusing on the decoder may be more fruitful for getting neural networks to learn specific types of representations.

The contribution of the training task: We next explore how the training task affects the representations that are learned. We test four tasks, illustrated in Table 3a: **autoencoding** (returning the input sequence unchanged), **reversal** (reversing the input), **sorting** (returning the input digits in ascending order), and **interleaving** (alternating digits from the left and right edges of the input).

Table 3b gives the substitution accuracy for a TPDN trained to approximate a unidirectional encoder that was trained with a unidirectional decoder on each task. Training task noticeably influences the learned representations. First, though the model has learned mildly bidirectional roles favoring the left-to-right direction for autoencoding, for reversal the right-to-left direction is far preferred over left-to-right. For interleaving, the model is approximated best with strongly bidirectional roles: that is, bidirectional roles work nearly perfectly, while neither unidirectional scheme works well. Finally, for sorting, bag-of-words roles work nearly as well as all other schemes, suggesting that the model

(a)

Input	3,4,0	4,3,6,5,1,3
Autoencode	3,4,0	4,3,6,5,1,3
Reverse	0,4,3	3,1,5,6,3,4
Sort	0,3,4	1,3,3,4,5,6
Interleave	3,0,4	4,3,3,1,6,5

(b)

	LTR	RTL	Bi	Wickel	Tree	BOW
Autoencode	0.87	0.11	**0.99**	0.28	0.25	0.00
Reverse	0.06	0.99	**1.00**	0.18	0.20	0.00
Sort	0.90	0.90	**0.92**	0.88	0.89	0.89
Interleave	0.27	0.18	**0.99**	0.63	0.36	0.00

Table 3: (a) Tasks used to test for the effect of task on learned roles (Section 4). (b) Accuracy of the TPDN applied to models trained on these tasks with a unidirectional encoder and decoder. All numbers are averages across five random initializations.

has learned to discard most structural information since sorting does not depend on structure. These experiments suggest that RNNs only learn compositional representations when the task requires them. This result might explain why the sentence embedding models do not seem to robustly encode structure: perhaps the training tasks for these models do not heavily rely on sentence structure (e.g. Parikh et al. (2016) achieved high accuracy on SNLI using a model that ignores word order), such that the models learn to ignore structural information, as was the case with models trained on sorting.

5 Related work

There are several approaches for interpreting neural network representations. One approach is to infer the information encoded in the representations from the system's behavior on examples targeting specific representational components, such as semantics (Pavlick, 2017; Dasgupta et al., 2018; Poliak et al., 2018) or syntax (Linzen et al., 2016). Another approach is based on probing tasks, which assess what information can be easily decoded from a vector representation (Shi et al. 2016; Belinkov et al. 2017; Kádár et al. 2017; Ettinger et al. 2018; compare work in cognitive neuroscience, e.g. Norman et al. 2006). Our method is wider-reaching than the probing task approach, or the Mikolov et al. (2013) analogy approach: instead of decoding a single feature, we attempt to exhaustively decompose the vector space into a linear combination of filler-role bindings.

The TPDN's successful decomposition of sequence representations in our experiments shows that RNNs can sometimes be approximated with no nonlinearities or recurrence. This finding is related to the conclusions of Levy et al. (2018), who argued that LSTMs dynamically compute weighted sums of their inputs; TPRs replace the weights of the sum with the role vectors. Levy et al. (2018) also showed that recurrence is largely unnecessary for practical applications. Vaswani et al. (2017) report very good performance for a sequence model without recurrence; importantly, they find it necessary to incorporate sequence position embeddings, which are similar to the left-to-right roles discussed in Section 2. Methods for interpreting neural networks using more interpretable architectures have been proposed before based on rules and automata (Omlin & Giles, 1996; Weiss et al., 2018).

Our decomposition of vector representations into independent fillers and roles is related to work on separating latent variables using singular value decomposition and other factorizations (Tenenbaum & Freeman, 2000; Anandkumar et al., 2014). For example, in face recognition, eigenfaces (Sirovich & Kirby, 1987; Turk & Pentland, 1991) and TensorFaces (Vasilescu & Terzopoulos, 2002; 2005) use such techniques to disentangle facial features, camera angle, and lighting.

Finally, there is a large body of work on incorporating explicit symbolic representations into neural networks (for a recent review, see Battaglia et al. 2018); indeed, tree-shaped RNNs are an example of this approach. While our work is orthogonal to this line of work, we note that TPRs and other filler-role representations can profitably be used as an explicit component of neural models (Koniusz et al., 2017; Palangi et al., 2018; Huang et al., 2018; Tang et al., 2018; Schlag & Schmidhuber, 2018).

6 Conclusion

What kind of internal representations could allow simple sequence-to-sequence models to perform the remarkable feats they do, including tasks previously thought to require compositional, symbolic representations (e.g., translation)? Our experiments show that, in heavily structure-sensitive tasks, sequence-to-sequence models learn representations that are extremely well approximated by tensor-product representations (TPRs), distributed embeddings of symbol structures that enable powerful symbolic computation to be performed with neural operations (Smolensky, 2012). We demonstrated this by approximating learned representations via TPRs using the proposed tensor-product decomposition network (TPDN). Variations in architecture and task were shown to induce different types and degrees of structure-sensitivity in representations, with the decoder playing a greater role than the encoder in determining the structure of the learned representation. TPDNs applied to mainstream sentence-embedding models reveal that unstructured bag-of-words models provide a respectable approximation; nonetheless, this experiment also provides evidence for a moderate degree of structure-sensitivity. The presence of structure-sensitivity is corroborated by targeted analogy tests motivated by the linearity of TPRs. A limitation of the current TPDN architecture is that it requires a hypothesis about the representations to be selected in advance. A fruitful future research direction would be to automatically explore hypotheses about the nature of the TPR encoded by a network.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1746891 and NSF INSPIRE grant BCS-1344269. This work was also supported by ERC grant ERC-2011-AdG-295810 (BOOTPHON), and ANR grants ANR-10-LABX-0087 (IEC) and ANR-10-IDEX-0001-02 (PSL*), ANR-17-CE28-0009 (GEOMPHON), ANR-11-IDEX-0005 (USPC), and ANR-10-LABX-0083 (EFL). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or the other supporting agencies.

For helpful comments, we are grateful to Colin Wilson, John Hale, Marten van Schijndel, Jan Hůla, the members of the Johns Hopkins Gradient Symbolic Computation research group, and the members of the Deep Learning Group at Microsoft Research, Redmond. Any errors remain our own.

REFERENCES

Animashree Anandkumar, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *The Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 861–872, Vancouver, Canada, 2017. Association for Computational Linguistics. URL http://aclweb.org/anthology/P17-1080.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. URL http://aclweb.org/anthology/D15-1075.

Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1466–1477. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/P16-1139.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 1–14. Association for Computational Linguistics, 2017. doi: 10.18653/v1/S17-2001. URL http://www.aclweb.org/anthology/S17-2001.

Huadong Chen, Shujian Huang, David Chiang, and Jiajun Chen. Improved neural machine translation with a syntax-aware encoder and decoder. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1936–1945. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1177. URL http://www.aclweb.org/anthology/P17-1177.

Xinyun Chen, Chang Liu, and Dawn Song. Tree-to-tree neural networks for program translation. *arXiv preprint arXiv:1802.03691*, 2018.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. In *International Conference on Language Resources and Evaluation*, 2018.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 670–680. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/D17-1070.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J. Gershman, and Noah D. Goodman. Evaluating compositionality in sentence embeddings. In *Proceedings of the 40th Annual Conference of the Cognitive Science Society*, 2018. URL https://arxiv.org/abs/1802.04302.

Bill Dolan, Chris Quirk, and Chris Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, 2004. URL http://www.aclweb.org/anthology/C04-1051.

Allyson Ettinger, Ahmed Elgohary, Colin Phillips, and Philip Resnik. Assessing composition in sentence vector representations. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1790–1801. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/C18-1152.

Simon Fischer-Baum, Michael McCloskey, and Brenda Rapp. Representation of letter position in spelling: Evidence from acquired dysgraphia. *Cognition*, 115(3):466–490, 2010.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

Qiuyuan Huang, Paul Smolensky, Xiaodong He, Li Deng, and Dapeng Wu. Tensor product generation networks for deep nlp modeling. In *Proceedings of NAACL*, 2018. URL https://arxiv.org/abs/1709.09118.

Ákos Kádár, Grzegorz Chrupała, and Afra Alishahi. Representation of linguistic form and function in recurrent neural networks. *Computational Linguistics*, 43(4):761–780, 2017. URL http://aclweb.org/anthology/J17-4003.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference for Learning Representations*, 2015.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *Advances in Neural Information Processing Systems*, pp. 3294–3302, 2015.

Piotr Koniusz, Fei Yan, Philippe-Henri Gosselin, and Krystian Mikolajczyk. Higher-order occurrence pooling for bags-of-words: Visual concept detection. *IEEE transactions on pattern analysis and machine intelligence*, 39(2):313–326, 2017.

Omer Levy, Kenton Lee, Nicholas FitzGerald, and Luke Zettlemoyer. Long short-term memory as a dynamically computed element-wise weighted sum. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 732–739. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/P18-2116.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of NAACL-HLT*, pp. 746–751, 2013.

Richard Montague. English as a formal language. In Richard Thomason (ed.), *Formal Philosophy. Selected papers by Richard Montague*, pp. 188–221. Yale University Press, 1974.

Allen Newell. Physical symbol systems. *Cognitive Science*, 4(2):135–183, 1980.

Kenneth A. Norman, Sean M. Polyn, Greg J. Detre, and James V. Haxby. Beyond mind-reading: multi-voxel pattern analysis of fMRI data. *Trends in Cognitive Sciences*, 10(9):424–430, 2006.

Christian W. Omlin and C. Lee Giles. Extraction of rules from discrete-time recurrent neural networks. *Neural Networks*, 9(1):41–52, 1996.

Hamid Palangi, Paul Smolensky, Xiaodong He, and Li Deng. Question-answering with grammatically-interpretable representations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17090.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2249–2255. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1244. URL http://aclweb.org/anthology/D16-1244.

Ellie Pavlick. *Compositional lexical semantics in natural language inference*. PhD thesis, University of Pennsylvania, 2017.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

Tony A. Plate. Holographic reduced representations. *IEEE Transactions on Neural networks*, 6(3): 623–641, 1995.

Adam Poliak, Yonatan Belinkov, James Glass, and Benjamin Van Durme. On the evaluation of semantic phenomena in neural machine translation using natural language inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pp. 513–523. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/N18-2082.

Jordan B. Pollack. Recursive distributed representations. *Artificial Intelligence*, 46(1-2):77–105, 1990.

Imanol Schlag and Jürgen Schmidhuber. Learning to reason with third order tensor products. In *Advances in Neural Information Processing Systems*, pp. 10003–10014, 2018.

Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

Xing Shi, Inkit Padhi, and Kevin Knight. Does string-based neural MT learn source syntax? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1526–1534. Association for Computational Linguistics, 2016. doi: 10.18653/v1/D16-1159. URL http://www.aclweb.org/anthology/D16-1159.

Lawrence Sirovich and Michael Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of the Optical Society of America A*, 4(3):519–524, 1987.

Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46(1-2):159–216, 1990.

Paul Smolensky. Symbolic functions from neural computation. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 370(1971):3543–3569, 2012.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, 2010.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1556–1566, Beijing, China, July 2015. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P15-1150`.

Shuai Tang, Paul Smolensky, and Virginia R de Sa. Learning distributed representations of symbolic structure using binding and unbinding operations. *NeurIPS Workshop https://openreview.net/forum?id=r1zvGR6jjm and arXiv preprint arXiv:1810.12456*, 2018.

Joshua B Tenenbaum and William T Freeman. Separating style and content with bilinear models. *Neural computation*, 12(6):1247–1283, 2000.

Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3 (1):71–86, 1991.

M. Alex O. Vasilescu and Demetri Terzopoulos. Multilinear image analysis for facial recognition. In *Proceedings of the 16th International Conference on Pattern Recognition, 2002.*, volume 2, pp. 511–514. IEEE, 2002.

M. Alex O. Vasilescu and Demetri Terzopoulos. Multilinear independent components analysis. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 547–553. IEEE, 2005.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. In *ICML*, pp. 5244–5253, 2018.

Wayne A. Wickelgren. Context-sensitive coding, associative memory, and serial order in (speech) behavior. *Psychological Review*, 76(1):1–15, 1969.

A LIST OF ACRONYMS AND ABBREVIATIONS

Bi	Bidirectional
BOW	Bag of words
LTR	Left to right
MRPC	Microsoft Research Paraphrase Corpus
RTL	Right to left
SNLI	Stanford Natural Language Inference corpus
SST	Stanford Sentiment Treebank
STS-B	Semantic Textual Similarity Benchmark
TPDN	Tensor product decomposition network
TPR	Tensor product representation
Uni	Unidirectional
Wickel	Wickelroles (see Section 2.1)

B ANALYSIS OF ARCHITECTURE COMPONENTS

Here we analyze how several aspects of the TPDN architecture contribute to our results. For all of the experiements described in this section, we used TPDNs to approximate a sequence-to-sequence network with a unidirectional encoder and unidirectional decoder that was trained to perform the reversal task (Section 4); we chose this network because it was strongly approximated by right-to-left roles, which are relatively simple (but still non-trivial).

B.1 IS THE FINAL LINEAR LAYER NECESSARY?

One area where our model diverges from traditional tensor product representations is in the presence of the final linear layer (step 5 in Figure 2c). This layer is necessary if one wishes to have freedom to choose the dimensionality of the filler and role embeddings; without it, the dimensionality of the representations that are being approximated must factor exactly into the product of the dimensionality of the filler embeddings and the dimensionality of the role embedding (see Figure 2c). It is natural to wonder whether the only contribution of this layer is in adjusting the dimensionality or whether it serves a broader function. Table 4 shows the results of approximating the reversal sequence-to-sequence network with and without this layer; it indicates that this layer is highly necessary for the successful decomposition of learned representations. (Tables follow all appendix text.)

B.2 VARYING THE DIMENSIONALITY OF THE FILLER AND ROLE EMBEDDINGS

Two of the parameters that must be provided to the TPDN are the dimensionality of the filler embeddings and the dimensionality of the role embeddings. We explore the effects of these parameters in Figure 4. For the role embeddings, substitution accuracy increases noticeably with each increase in dimensionality until the dimensionality hits 6, where accuracy plateaus. This behavior is likely due to the fact that the reversal seq2seq network is most likely to employ right-to-left roles, which involves 6 possible roles in this setting. A dimensionality of 6 is therefore the minimum embedding size needed to make the role vectors linearly independent; linear independence is an important property for the fidelity of a tensor product representation (Smolensky, 1990). The accuracy also generally increases as filler dimensionality increases, but there is a less clear point where it plateaus for the fillers than for the roles.

B.3 FILLER-ROLE BINDING OPERATION

The body of the paper focused on using the tensor product ($f_i \otimes r_i$, see Figure 2b) as the operation for binding fillers to roles. There are other conceivable binding operations. Here we test two alternatives, both of which can be viewed as special cases of the tensor product or as related to it: circular convolution, which is used in holographic reduced representations (Plate, 1995), and elementwise product ($f_i \odot r_i$). Both of these are restricted such that roles and fillers must have the same embedding dimension ($N_f = N_r$). We first try setting this dimension to 20, which is what was used as both the role and filler dimension in all tensor product experiments with digit sequences.

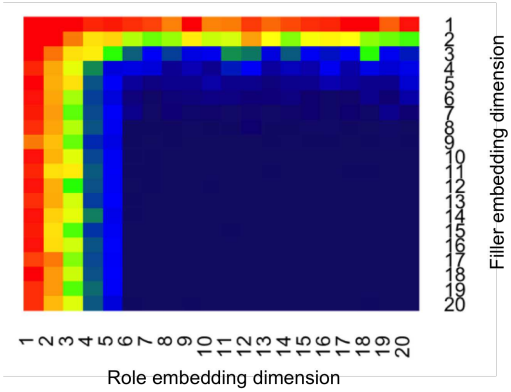


Figure 4: Heatmap of substitution accuracies with various filler and role embedding dimensions. Red indicates accuracy under 1%; dark blue indicates accuracy over 80%. The models whose substitution accuracies are displayed are all TPDNs trained to approximate a sequence-to-sequence model that was trained on the task of reversal.

We found that while these dimensions were effective for the tensor product binding operation, they were not effective for elementwise product and circular convolution (Table 5). When the dimension was increased to 60, however, the elementwise product performed roughly as well as as the tensor product; circular convolution now learned one of the two viable role schemes (right-to-left roles) but failed to learn the equally viable bidirectional role scheme. Thus, our preliminary experiments suggest that these other two binding operations do show promise, but seem to require larger embedding dimensions than tensor products do. At the same time, they still have fewer parameters overall compared to the tensor product because their final linear layers (of dimensionality N) are much smaller than those used with a tensor product (of dimensionality N^2).

C THE DIGIT PARSING ALGORITHM

When inputting digit sequences to our tree-based model, the model requires a predefined tree structure for the digit sequence. We use the following algorithm to generate this tree structure: at each timestep, combine the smallest element of the sequence (other than the last element) with its neighbor immediately to the right, and replace the pair with that neighbor. If there is a tie for the smallest digit, choose the leftmost tied digit.

For example, the following shows step-by-step how the tree for the sequence 523719 would be generated:

- 5 2 3 7 1 9
- 5 2 3 7 [1 9]
- 5 [2 3] 7 [1 9]
- 5 [[2 3] 7] [1 9]
- [5 [[2 3] 7] [1 9]]

D FULL RESULTS OF SEQUENCE-TO-SEQUENCE EXPERIMENTS

Section 4 summarized the results of our experiments which factorially varied the training task, the encoder and the decoder. Here we report the full results of these experiments in two tables: Table 6 shows the accuracies achieved by the sequence-to-sequence models at the various training tasks, and Table 7 shows the substitution accuracies of TPDNs applied to the trained sequence-to-sequence models for all architectures and tasks.

E MODEL AND TRAINING DETAILS

E.1 SEQUENCE-TO-SEQUENCE MODELS

As much as possible, we standardized parameters across all sequence-to-sequence models that we trained on the digit-sequence tasks.

For all decoders, when computing a new hidden state, the only input to the recurrent unit is the previous hidden state (or parent hidden state, for a tree-based decoder), without using any previous outputs as inputs to the hidden state update. This property is necessary for using a bidirectional decoder, since it would not be possible to generate the output both before and after each bidirectional decoder hidden state.

We also inform the decoder of when to stop decoding; that is, for sequential models, the decoder stops once its output is the length of the sequence, while for tree-based models we tell the model which positions in the tree are leaves. Stopping could alternately be determined by some action of the decoder (e.g., generating an end-of-sequence symbol); for simplicity we chose the strategy outlined above instead.

For all architectures, we used a digit embedding dimensionality of 10 (chosen arbitrarily) and a hidden layer size of 60 (this hidden layer size was chosen because 60 has many integer factors, making it amenable to the dimensionality analyses in Appendix B.2). For the bidirectional architectures, the forward and backward recurrent layers each had a hidden layer size of 30, so that their concatenated hidden layer size was 60. For bidirectional decoders, a linear layer condensed the 60-dimensional encoding into 30 dimensions before it was passed to the forward and backward decoders.

The networks were trained using the Adam optimizer (Kingma & Ba, 2015) with the standard initial learning rate of 0.001. We used negative log likelihood, computed over the softmax probability distributions for each output sequence element, as the loss function. Training proceeded with a batch size of 32, with loss on the held out development set computed after every 1,000 training examples. Training was halted when the loss on the heldout development set had not improved for any of the development loss checkpoints for a full epoch of training (i.e. 40,000 training examples). Once training completed, the parameters from the best-performing checkpoint were reloaded and used for evaluation of the network.

E.2 TPDNS TRAINED ON DIGIT MODELS

When applying TPDNs to the digit-based sequence-to-sequence models, we always used 20 as both the filler embedding dimension and the role embedding dimension. This decision was based on the experiments in Appendix B.2; we selected filler and role embedding dimensions that were safely above the cutoff needed to lead to successful decomposition.

The TPDNs were trained with the same training regimen as the sequence-to-sequence models, except that, instead of using negative log likelihood as the loss function, for the TPDNs we used mean squared error between the predicted vector representation and the actual vector representation from the original sequence-to-sequence network.

The TPDNs were given the sequences of fillers (i.e. the digits), the roles hypothesized to go with those fillers, the sequence embeddings produced by the RNN, and the dimensionalities of the filler embeddings, role embeddings, and final linear transformation. The parameters that were updated by training were the specific values for the filler embeddings, the role embeddings, and the final linear transformation.

E.3 SENTENCE EMBEDDING MODELS

For all four sentence encoding models, we used publicly available and freely downloadable pretrained versions found at the following links:

- InferSent: `https://github.com/facebookresearch/InferSent`
- Skip-thought: `https://github.com/ryankiros/skip-thoughts`
- SST: `https://nlp.stanford.edu/software/corenlp.shtml`

- SPINN: `https://github.com/stanfordnlp/spinn`

InferSent is a bidirectional LSTM with 4096-dimensional hidden states. For Skip-thought, we use the unidirectional variant, which is an LSTM with 2400-dimensional hidden states. The SST model is a recurrent neural tensor network (RNTN) with 25-dimensional hidden states. Finally, for SPINN, we use the SPINN-PI-NT version, which is equivalent to a tree-LSTM (Tai et al., 2015) with 300-dimensional hidden states.

E.4 TPDNs TRAINED ON SENTENCE MODELS

For training a TPDN to approximate the sentence encoding models, the filler embedding dimensions were dictated by the size of the pretrained word embeddings; these dimensions were 300 for InferSent and SPINN, 620 for Skip-thought, and 25 for SST. The linear transformation applied to the word embeddings did not change their size. For role embedding dimensionality we tested all role dimensions in $\{1, 5, 10, 20, 40, 60\}$. The best-performing dimension was chosen based on preliminary experiments and used for all subsequent experiments; we thereby chose role dimensionalities of 10 for InferSent and Skip-thought, 20 for SST, and 5 for SPINN. In general, role embedding dimensionalities of 5, 10, and 20 all performed noticeably better than 1, 40, and 60, but there was not much difference between 5, 10, and 20.

The training regimen for the TPDNs on sentence models was the same as for the TPDNs trained on digit sequences. The TPDNs were given the sequences of fillers (i.e. the words), the roles hypothesized to go with those fillers, the sequence embeddings produced by the RNN, the initial pretrained word embeddings, the dimensionalities of the linearly-transformed filler embeddings, the role embeddings, and the final linear transformation. The parameters that were updated by training were the specific values for the role embeddings, the linear transformation that was applied to the pretrained word embeddings, and the final linear transformation.

The sentences whose encodings we trained the TPDNs to approximate were the premise sentences from the SNLI corpus (Bowman et al., 2015). We also tried instead using the sentences in the WikiText-2 corpus (Merity et al., 2016) but found better performance with the SNLI sentences. This is plausibly because the shorter, simpler sentences in the SNLI corpus made it easier for the model to learn the role embeddings without distraction from the fillers.

F DOWNSTREAM TASK PERFORMANCE FOR TPDNs APPROXIMATING SENTENCE ENCODERS

For each TPDN trained to approximate a sentence encoder, we evaluate it on four downstream tasks: (i) Stanford Sentiment Treebank (SST), which is labeling the sentiment of movie reviews (Socher et al., 2013); this task is further subdivided into SST2 (labeling the reviews as *positive* or *negative*) and SST5 (labeling the reviews on a 5-point scale, where 1 means *very negative* and 5 means *very positive*). The metric we report for both tasks is accuracy. (ii) Microsoft Research Paraphrase Corpus (MRPC), which is labeling whether two sentences are paraphrases of each other (Dolan et al., 2004). For this task, we report both accuracy and F1. (iii) Semantic Textual Similarity Benchmark (STS-B), which is giving a pair of sentences a score on a scale from 0 to 5 indicating how similar the two sentences are (Cer et al., 2017). For this task, we report Pearson and Spearman correlation coefficients. (iv) Stanford Natural Language Inference (SNLI), which involves labeling a pair of sentences to indicate whether the first entails the second, contradicts the second, or neither (Bowman et al., 2015). For this task, we report accuracy as the evaluation metric.

F.1 SUBSTITUTION PERFORMANCE

The first results we report for the TPDN approximations of sentence encoders is similar to the substitution accuracy used for digit encoders. Here, we use SentEval (Conneau & Kiela, 2018) to train linear classifiers for all downstream tasks on the original sentence encoding model; then, we freeze the weights of these classifiers and use them to classify the test-set encodings generated by the TPDN approximation. We use the classifier parameters recommended by the SentEval authors: using a linear classifier (not a multi-layer perceptron) trained with the Adam algorithm (Kingma &

Ba, 2015) using a batch size of 64, a tenacity of 5, and an epoch size of 4. The results are shown in Table 8.

F.2 AGREEMENT BETWEEN THE TPDN AND THE ORIGINAL MODEL

Next, we analyze the same results from the previous section, but instead of reporting accuracies we report the extent to which the TPDN's predictions agree with the original model's predictions. For SST, MRPC, and SNLI, this agreement is defined as the proportion of their labels that are the same. For STS-B, the agreement is the Pearson correlation between the original model's outputs and the TPDN's outputs. The results are in Table 9.

F.3 TRAINING A CLASSIFIER ON THE TPDN

Finally, we consider treating the TPDNs as models in their own right and use SentEval to both train and test downstream task classifiers on the TPDNs. The results are in Table 10.

Filler dim.	Role dim.	Without linear layer	With linear layer
1	60	0.0002	0.003
2	30	0	0.042
3	20	0.001	0.82
4	15	0.0006	0.80
5	12	0	0.90
6	10	0	0.92
10	6	0.0002	0.99
12	5	0	0.67
15	4	0.0002	0.37
20	3	0	0.14
30	2	0.0002	0.02
60	1	0.001	0.0014

Table 4: Substitution accuracies with and without the final linear layer, for TPDNs using various combinations of filler and role embedding dimensionality. These TPDNs were approximating a seq2seq model trained to perform reversal.

	LTR	RTL	Bi	Wickel	Tree	BOW	Parameters
Tensor product (20 dim.)	0.054	0.993	0.996	0.175	0.188	0.002	24k
Tensor product (60 dim.)	0.046	0.988	0.996	0.138	0.172	0.001	217k
Circular convolution (20 dim.)	0.004	0.045	0.000	0.000	0.003	0.001	1.5k
Circular convolution (60 dim.)	0.048	0.964	0.066	0.001	0.013	0.001	4.6k
Elementwise product (20 dim.)	0.026	0.617	0.386	0.024	0.027	0.001	1.5k
Elementwise product (60 dim.)	0.051	0.992	0.993	0.120	0.173	0.001	4.6k

Table 5: Approximating a unidirectional seq2seq model trained to perform sequence reversal: substitution accuracies using different binding operations.

Encoder	Decoder	Autoencode	Reverse	Sort	Interleave
Uni	Uni	0.999	1.000	1.000	1.000
Uni	Bi	0.949	0.933	1.000	0.968
Uni	Tree	0.979	0.967	0.970	0.964
Bi	Uni	0.993	0.999	1.000	0.995
Bi	Bi	0.834	0.883	1.000	0.939
Bi	Tree	0.967	0.920	0.959	0.909
Tree	Uni	0.981	0.978	1.000	0.987
Tree	Bi	0.891	0.900	1.000	0.894
Tree	Tree	0.989	0.962	0.999	0.934

Table 6: Accuracies of the various sequence-to-sequence encoder/decoder combinations at the different training tasks. Each number in this table is an average across five random initializations. Uni = unidirectional; bi = bidirectional.

Task	Encoder	Decoder	LTR	RTL	Bi	Wickel	Tree	BOW
Autoencode	Uni	Uni	0.871	0.112	**0.992**	0.279	0.246	0.001
	Uni	Bi	0.275	0.273	**0.400**	0.400	0.238	0.005
	Uni	Tree	0.053	0.086	0.094	0.105	**0.881**	0.009
	Bi	Uni	0.748	0.136	**0.921**	0.209	0.207	0.002
	Bi	Bi	0.097	0.124	**0.179**	0.166	0.128	0.006
	Bi	Tree	0.051	0.081	0.088	0.095	**0.835**	0.007
	Tree	Uni	0.708	0.330	**0.865**	0.595	0.529	0.007
	Tree	Bi	0.359	0.375	0.491	**0.569**	0.376	0.009
	Tree	Tree	0.041	0.069	0.076	0.095	**0.958**	0.009
Reverse	Uni	Uni	0.062	0.986	**0.995**	0.177	0.204	0.002
	Uni	Bi	0.262	0.268	0.386	**0.406**	0.228	0.006
	Uni	Tree	0.103	0.112	0.177	0.169	**0.413**	0.002
	Bi	Uni	0.037	0.951	**0.965**	0.084	0.146	0.001
	Bi	Bi	0.121	0.140	**0.228**	0.170	0.140	0.005
	Bi	Tree	0.085	0.105	0.17	0.151	**0.385**	0.002
	Tree	Uni	0.178	0.755	**0.802**	0.424	0.564	0.007
	Tree	Bi	0.302	0.332	0.442	**0.549**	0.368	0.009
	Tree	Tree	0.083	0.096	0.147	0.152	**0.612**	0.004
Sort	Uni	Uni	0.895	0.895	**0.923**	0.878	0.890	0.892
	Uni	Bi	0.898	0.894	**0.923**	0.916	0.915	0.904
	Uni	Tree	0.218	0.212	0.207	0.193	**0.838**	0.275
	Bi	Uni	0.886	0.884	**0.917**	0.812	0.871	0.847
	Bi	Bi	0.921	0.925	**0.945**	0.835	0.927	0.934
	Bi	Tree	0.219	0.216	0.209	0.194	**0.816**	0.273
	Tree	Uni	0.997	0.998	0.997	**0.999**	0.999	0.998
	Tree	Bi	**1.000**	1.000	0.997	1.000	1.000	1.000
	Tree	Tree	0.201	0.199	0.179	0.181	**0.978**	0.249
Interleave	Uni	Uni	0.269	0.181	**0.992**	0.628	0.357	0.003
	Uni	Bi	0.177	0.095	**0.728**	0.463	0.255	0.005
	Uni	Tree	0.040	0.033	0.116	0.089	**0.373**	0.003
	Bi	Uni	0.186	0.126	**0.965**	0.438	0.232	0.001
	Bi	Bi	0.008	0.074	**0.600**	0.128	0.162	0.002
	Bi	Tree	0.031	0.025	0.069	0.057	**0.395**	0.004
	Tree	Uni	0.330	0.208	**0.908**	0.663	0.522	0.005
	Tree	Bi	0.191	0.151	**0.643**	0.518	0.391	0.006
	Tree	Tree	0.027	0.025	0.059	0.069	**0.606**	0.004

Table 7: Substitution accuracies for TPDNs applied to all combinations of encoder, decoder, training task, and hypothesized role scheme. Each number is an average across five random initializations. Uni = unidirectional; bi = bidirectional.

	LTR	RTL	Bi	Tree	BOW	Original
InferSent						
SST2	0.79	0.79	0.77	0.79	0.80	0.85
SST5	0.40	0.39	0.40	0.41	0.42	0.46
MRPC (accuracy)	0.70	0.71	0.72	0.70	0.72	0.73
MRPC (F1)	0.81	0.82	0.82	0.80	0.81	0.81
STS-B (Pearson)	0.69	0.70	0.71	0.70	0.69	0.78
STS-B (Spearman)	0.68	0.68	0.69	0.68	0.67	0.78
SNLI	0.71	0.69	0.72	0.71	0.66	0.84
Skip-thought						
SST2	0.58	0.51	0.50	0.50	0.61	0.81
SST5	0.29	0.27	0.21	0.25	0.31	0.43
MRPC (accuracy)	0.60	0.62	0.62	0.61	0.66	0.74
MRPC (F1)	0.73	0.75	0.76	0.75	0.79	0.82
STS-B (Pearson)	-0.01	-0.07	-0.10	-0.05	0.06	0.73
STS-B (Spearman)	0.23	-0.01	-0.05	0.00	0.08	0.72
SNLI	0.35	0.35	0.34	0.35	0.35	0.73
SST						
SST2	0.76	0.76	0.76	0.75	0.77	0.83
SST5	0.37	0.38	0.37	0.37	0.38	0.45
MRPC (accuracy)	0.67	0.67	0.67	0.66	0.66	0.66
MRPC (F1)	0.80	0.80	0.80	0.79	0.80	0.80
STS-B (Pearson)	0.24	0.21	0.22	0.19	0.24	0.29
STS-B (Spearman)	0.24	0.22	0.23	0.20	0.25	0.27
SNLI	0.40	0.41	0.41	0.41	0.40	0.42
SPINN						
SST2	0.73	0.73	0.73	0.74	0.74	0.76
SST5	0.36	0.36	0.35	0.37	0.37	0.39
MRPC (accuracy)	0.67	0.68	0.67	0.67	0.68	0.70
MRPC (F1)	0.75	0.78	0.76	0.76	0.76	0.79
STS-B (Pearson)	0.60	0.60	0.62	0.62	0.53	0.67
STS-B (Spearman)	0.58	0.59	0.59	0.59	0.57	0.65
SNLI	0.67	0.67	0.68	0.69	0.54	0.79

Table 8: Substitution results on performing the applied tasks for TPDNs trained to approximate the representations from each of the four downloaded models. For MRPC, we report accuracy and F1. For STS-B, we report Pearson correlation and Spearman correlation. All other metrics are accuracies.

	LTR	RTL	Bi	Tree	BOW	Original
Infersent						
SST2	0.85	0.84	0.82	0.83	0.84	1.00
SST5	0.60	0.59	0.58	0.59	0.61	1.00
MRPC	0.78	0.80	0.78	0.77	0.79	1.00
STS-B	0.87	0.86	0.87	0.86	0.84	1.00
SNLI	0.77	0.74	0.77	0.77	0.71	1.00
Skip-thought						
SST2	0.58	0.54	0.50	0.51	0.60	1.00
SST5	0.41	0.41	0.18	0.35	0.43	1.00
MRPC	0.69	0.71	0.74	0.72	0.79	1.00
STS-B	-0.10	-0.12	-0.16	-0.13	-0.04	1.00
SNLI	0.37	0.37	0.36	0.36	0.37	1.00
SST						
SST2	0.84	0.84	0.83	0.84	0.85	1.00
SST5	0.65	0.64	0.64	0.64	0.65	1.00
MRPC	0.99	0.99	0.99	0.98	0.99	1.00
STS-B	0.64	0.59	0.62	0.68	0.60	1.00
SNLI	0.48	0.51	0.49	0.67	0.49	1.00
SPINN						
SST2	0.77	0.77	0.77	0.79	0.79	1.00
SST5	0.61	0.62	0.63	0.63	0.62	1.00
MRPC	0.68	0.73	0.72	0.74	0.70	1.00
STS-B	0.72	0.73	0.74	0.76	0.70	1.00
SNLI	0.72	0.72	0.73	0.76	0.58	1.00

Table 9: The proportion of times that a classifier trained on a sentence encoding model gave the same downstream-task predictions based on the original sentence encoding model and based on a TPDN approximating that model, where the TPDN uses the role schemes indicated by the column header. For all tasks but STS-B, these numbers show the proportion of predictions that matched; chance performance is 0.5 for SST2 and MRPC, 0.2 for SST5, and 0.33 for SNLI. For STS-B, the metric shown is the Pearson correlation between the TPDN's similarity ratings and the original model's similarity ratings; chance performance here is 0.0.

	LTR	RTL	Bi	Tree	BOW	Original
Infersent						
SST2	0.82	0.82	0.81	0.81	0.83	0.85
SST5	0.44	0.44	0.44	0.44	0.43	0.46
MRPC (acc.)	0.71	0.73	0.72	0.70	0.73	0.73
MRPC (F1)	0.80	0.81	0.81	0.80	0.81	0.81
STS-B (Pearson)	0.71	0.71	0.71	0.71	0.71	0.78
STS-B (Spearman)	0.69	0.70	0.70	0.69	0.70	0.78
SNLI	0.77	0.76	0.77	0.77	0.75	0.84
Skip-thought						
SST2	0.59	0.56	0.50	0.52	0.61	0.81
SST5	0.30	0.30	0.25	0.28	0.32	0.43
MRPC (acc.)	0.67	0.66	0.66	0.66	0.67	0.74
MRPC (F1)	0.80	0.80	0.80	0.80	0.79	0.82
STS-B (Pearson)	0.19	0.17	0.13	0.13	0.23	0.73
STS-B (Spearman)	0.17	0.16	0.08	0.10	0.20	0.72
SNLI	0.47	0.46	0.46	0.44	0.46	0.73
SST						
SST2	0.78	0.78	0.77	0.78	0.79	0.83
SST5	0.40	0.40	0.40	0.39	0.41	0.45
MRPC (acc.)	0.67	0.67	0.66	0.66	0.67	0.66
MRPC (F1)	0.80	0.80	0.80	0.80	0.80	0.88
STS-B (Pearson)	0.28	0.23	0.23	0.20	0.28	0.29
STS-B (Spearman)	0.27	0.23	0.23	0.20	0.27	0.27
SNLI	0.45	0.44	0.45	0.43	0.45	0.42
SPINN						
SST2	0.79	0.78	0.77	0.77	0.80	0.76
SST5	0.42	0.42	0.40	0.42	0.42	0.39
MRPC (acc.)	0.72	0.71	0.71	0.68	0.72	0.70
MRPC (F1)	0.81	0.80	0.80	0.79	0.80	0.79
STS-B (Pearson)	0.68	0.67	0.67	0.67	0.66	0.67
STS-B (Spearman)	0.67	0.66	0.66	0.65	0.65	0.65
SNLI	0.72	0.71	0.72	0.73	0.71	0.79

Table 10: Downstream task performance for classifiers trained and tested on the TPDNs that were trained to approximate each of the four applied models. The rightmost column indicates the performance of the original model (without the TPDN approximation).

NYU

Tal Linzen
Assistant Professor

Department of Linguistics and Center for Data Science
60 5th Avenue
New York, NY 10011
linzen@nyu.edu

Dear search committee:

I am writing to express my strongest possible support for Richard Thomas (Tom) McCoy's application to the faculty position in your department.[1] I can say without reservations that Tom is the **single strongest Ph.D. student anywhere in the world working on cognitively motivated computational linguistics**. Tom is a rare example of an early-career researcher who combines intellectual maturity and independence, extremely strong modeling and experimentation skills, and an ability to think deeply about fundamental scientific and theoretical issues. His work brings ideas and methods from cognitive science and linguistics to bear on the challenges of delineating and addressing the limitations of current deep learning approaches to AI, especially in the areas of sample efficiency and robust compositional generalization. As I hope to illustrate below, his work is already amply recognized in the field; I have no doubt that his contributions will only deepen in scope and impact.

Tom's goal in his dissertation is to come up with a complete mechanical understanding of the compositional generalization abilities of neural networks. Tom started working on this project in the first year of his Ph.D. In a paper we published in ICLR 2019, a top-tier machine learning conference, Tom showed that recurrent neural networks (RNNs) that are trained as sequence autoencoders evolve surprisingly compositional internal representations. The representation of the sequence can be decomposed into a sum of the representations of its elements; for example, the sequence (2, 7) is represented as the sum of vectors that indicate "the digit 2 in first position" and "the digit 7 in second position". Moreover, these filler/role binding vectors can be further decomposed: the vector for "2 in first position" can be predicted from vectors whose semantics is "the digit 2" and "first position". In a follow-up paper, which was presented at the BlackboxNLP 2020 workshop, Tom and another student extended this technique to provide a complete, interpretable characterization of the algorithm implemented by a network trained to solve the SCAN compositionality benchmark, where simple English sentences need to be translated into "action sequences" (e.g., *jump twice* to JUMP JUMP). By contrast, when trained on natural language, the networks' representations were much less compositional. This line of work is a rare triumph of "interpretability", the area that attempts, often unsuccessfully, to shed light on how "black box" neural networks accomplish their behavior.

In another line of work, Tom has worked with Robert Frank (Yale) and myself on a project that studies how various architectural factors affect syntactic generalization by neural networks. The goal of this project is to understand what architectural features will lead those networks to generalize like humans from small amounts of data. We focus on syntactic transformations, such as forming a question from a declarative sentence; these transformations are informed by classic debates in linguistics that fall under the banner of "the poverty of the stimulus", but are also representative of the challenges that arise in applied sequence transduction problems such as machine translation. The conclusion of Tom's experiments, published in 2020 in TACL, was that the only way to impart a reliable human-like inductive bias to a neural network is by using explicit syntactic trees as part of the architecture of the network (Tree RNNs). Other architectures, including the Ordered Neurons architecture, which has been advertised as imparting a syntactic inductive bias, did not lead to the expected generalization pattern. This project showcases Tom's ability to synthesize the results of a dizzying number of experiments into a concise statement that addresses theoretical questions in cognitive science and artificial intelligence.

[1] I am currently an Assistant Professor of Linguistics and Data Science at NYU, after serving three years as an Assistant Professor of Cognitive Science at Johns Hopkins University (where Tom is enrolled). I retain a non-salaried position at Johns Hopkins University and co-advise Tom's dissertation, jointly with Paul Smolensky.

Tom is also concerned with linguistically sophisticated evaluation of NLP models. In a project published at ACL 2019 (a top-ranked NLP conference), he constructed a challenge set, called HANS, that assesses the syntactic generalization abilities of neural networks trained, roughly, to determine whether one sentence logically follows from another (this task, commonly seen as a benchmark for language understanding, is referred to as Natural Language Inference, or NLI). Tom used the HANS data set to show that an NLI system based on the widely popular BERT model makes errors in embarrassingly simple cases – instead of performing the task based on the structure of the two sentences, it relies on heuristics such as word overlap between the sentences. This heuristic does sometimes work, but often it doesn't; for example, BERT concludes from the sentence *the judge chastised the lawyer* that it is the case that *the lawyer chastised the judge*. This suggests that standard datasets for measuring progress in NLI provide an incomplete picture of the task, and that considerable progress will be required before we can construct robust natural language understanding models. Tom completed this project as with relatively light supervision from me – even as a second-year Ph.D. student he had my complete trust in his scientific intuitions, experimental design abilities and technical skills.

It has been a pleasure to observe the influence that HANS has had on NLP practitioners, with 386 citations so far, 179 of them in the first nine months of 2021 alone. Tom and I have recently started planning a retrospective article reviewing all of the methods that have been proposed in the last three years to address the issues highlighted by HANS, and I can confirm that the list of approaches motivated by HANS is really quite long. I don't mean to suggest that Tom's paper was the only one to come out in 2018 or 2019 that highlighted issues with neural networks' overreliance on heuristics; there were quite a few of those. But HANS differs from the rest of these papers in clarity of presentation, in how compellingly it demonstrates the severity of the issues, and, perhaps most importantly, in its linguistic sophistication; where Tom identified constructions, such as subject/object inversion, the dative alternation, or garden path sentences, where word or subsequence overlap can lead the model astray, other studies have measured sensitivity to word overlap by appending tautological clauses to one of the sentences in the pair (e.g., *the judge chastised the lawyer and true is true*). Grated, such strategies do in fact reduce word overlap, but make for a less compelling demonstration of the issue.

A last project I will mention – a more theoretical project that I am personally particularly excited about – is his work on "Universal Grammar through meta-learning". This project, a collaboration with Tom Griffith's group at Princeton, is motivated by a deep question: how can we endow neural network architectures for language processing with the inductive biases that will enable them to generalize as quickly and as robustly as humans? The technique that Tom proposes to apply to address this question is meta-learning: imparting inductive bias to a learner by providing it with exposure to strategically chosen tasks that are similar to the target task. This method is much more flexible than architecturally-defined inductive bias (such as Tree RNNs): it is usually difficult to construct architectures that implement a particular bias, but easier to come up with cases that demonstrate the desired generalization pattern. A proof-of-concept demonstration of this technique, applied to the classic phonological problem of learning syllable structure constraints, was published in the Proceedings of the Cognitive Science Society in 2020. Next, Tom plans to apply this technique to syntax, implementing typological biases such as the head directionality bias, where the order of the head and the complement are often consistent within a language. This project is a great example of Tom's unique strength: mastering cutting-edge machine learning techniques and, rather than advancing those techniques (which thousands of people who submit to ICLR are able to do), putting them into practice in a linguistically motivated way.

In all these projects, Tom was able to run an enormous number of experiments in a short period of time, and showed impressive independence: where some early Ph.D. students will run an experiment and wait until their weekly meeting with their advisor to interpret it, Tom runs a long series of follow-up experiments to rule out alternative hypotheses, explore other hyperparameter values and perform ablation studies, all

between one weekly meeting and the next. Mysteriously, he does not seem to suffer from the ebbs and flows in productivity that plague the rest of us.

Tom's research instincts are robust enough, and his ability to collaborate in research teams developed enough, that I have entrusted him, even as a relatively junior Ph.D. student, with supervising the research projects of a number of undergraduate and Master's students. He single-handedly supervised a recent senior thesis written by a star JHU undergraduate, which has resulted in two separate papers (in ACL 2020 and COLING 2020), both of which with Tom as the senior author, and with minimal or no involvement from me. It wouldn't be an exaggeration to say that beginning in his third or fourth year as a Ph.D. student Tom already played a larger role in my group than a typical post-doc would. I treat him as a colleague, rather than a student.

Tom served as a teaching assistant (TA) for two of my classes, Introduction to Computational Cognitive Science and Computational Psycholinguistics. He was a highly reliable and effective TA. He taught the labs he created for the class in an exceptionally clear way, and gave excellent guest lectures (for example, explaining the Earley parsing algorithm much better than I would have, with very detailed slides). As part of a grant that we received from the Johns Hopkins Center of Educational Resources, he developed online demos for teaching computational cognitive science to undergraduate students with limited computational background; I have found those demos to be very useful teaching tools since then.

Alongside his research and teaching activities, Tom is deeply committed to community outreach. In fact, he serves as the outreach officer for the North American Computational Linguistics Olympiad, an annual competition whose goal is to expose high school students to linguistics in general and computational linguistics in particular. As part of his role as outreach officer, Tom has presented the Olympiad in Baltimore schools, most of which are serve a low-income minority population. He has constructed questions for the Olympiad that are based on our joint research on linguistic generalization and inductive biases – a nice synergy between research and outreach activities.

In summary, Tom is a bright, driven, unusually productive and socially conscious AI researcher. He has already risen to the level of a superstar in computational linguistics, and given his brilliance, dedication, depth of thinking and collaboration abilities, I have no doubt that his success will only increase. He is bursting with ideas, and is absolutely ready to start his own research group. And, quite simply, he is a delight to be around and work with: always cheerful, generous and reliable. He has my absolutely highest possible recommendation for the junior faculty position in your department.

Sincerely,

Tal Linzen
Assistant Professor of Linguistics and Data Science
New York University

JOHNS HOPKINS
UNIVERSITY

Paul Smolensky
Partner Researcher
psmo@microsoft.com
microsoft.com/research/people/psmo/

Microsoft Research
One Microsoft Way
Redmond, WA
98052-6399 USA

Department of Cognitive Science
Krieger-Eisenhower Professor Paul Smolensky
239A Krieger Hall / 3400 N. Charles Street
Baltimore, MD 21218-2685
smolensky@jhu.edu
cogsci.jhu.edu/directory/paul-smolensky/

Microsoft

November 28, 2021

Dear colleagues:

I'm writing to give my strongest possible support to Richard Thomas McCoy's candidacy for your advertised faculty position in computer science. Tom is already a star and will no doubt become a superstar in computational linguistics. I've worked closely with Tom as his co-Ph.D. supervisor (with Tal Linzen) since 2017, when he joined the doctoral program in the Johns Hopkins Department of Cognitive Science, where I now have a part-year position. In this letter, I will first discuss Tom's research, and then the set of general qualifications he brings to the position of faculty member.

To avoid endless repetition of "and colleagues" I'll assume this to be tacitly present throughout the discussion of "Tom's work". In all the projects discussed here, Tom played by far the greatest role in designing and carrying out the work. I will also use "nML" as an abbreviation for "neural models of language", meant in a general sense (not restricted to NLP "language models").

RESEARCH

Those of us interested in computation in language are fortunate to be living in an extraordinary time. We are witnessing a huge shift in our field, and the exciting question of which of many still-possible futures we will see some years from now is very much open. Tom views his work in this context, seeing the branching futures as turning, first, on the question:

(1) Given their great success, do neural models of language processing (nMLs) prove that language scientists have been wrong to believe (for many centuries) that true linguistic competence requires complex symbolic representations and rules?

It may indeed be true that language scientists have been profoundly wrong about this, but 2 alternative possible responses remain alive:

(2) Perhaps not, *version 1*: Although they perform well on automatic benchmarks, current nMLs do not in fact possess true linguistic knowledge or competence.

(3) Perhaps not, *version 2*: Current nMLs do possess true linguistic competence, and on the surface it seems that they lack symbolic representations and rules; but in fact they do have such structure hidden inside them, and it is this structure that explains their success.

Tom sees the next layer of branching as turning on the question:

(4) If current nMLs do not possess true linguistic competence, can they be strengthened to possess such competence?

There are several live possible responses here too:

(5) Perhaps yes, if they are given the kinds of inductive biases that enable children to reliably acquire such competence from relatively little experience. In nMLs, such biases could, for example, take the form of:

a. extra training examples designed to target linguistic knowledge that analysis has shown models tend to fail to learn;

b. capabilities for explicitly building hidden symbolic representations and rules;

c. training objectives rewarding construction of implicit hidden symbolic structure;

d. an initial set of connection weights that has been selected to bias search during learning towards weights that efficiently learn any of a pre-selected set of target languages.

Tom's work has in fact contributed significant progress towards the realization of each of these possible futures. The following 9 contributions provide a non-exhaustive sample (the numbering corresponds with the above analysis of possible computational-linguistic futures).

(1) ***nML text generation displays an impressive degree of novelty consistent with abstract linguistic competence.*** In his extensive study of text generation by GPT-2 ([3] in the references at the end of this letter), Tom has carefully documented many telling instances, and compiled convincing statistics, showing that those who would like to believe that GPT-2's success relies on regurgitating chunks of text memorized during training are simply wrong. For example, the model generates new words and uses them correctly with respect to morpho-syntactic criteria, and generates a plethora of novel syntactic structures.

(1′) ***Certain nMLs generalize ambiguous data hierarchically.*** As long conjectured by linguists, Tom's experimental work on human artificial-language learning [4] reveals a bias to impose recursive hierarchical structure on artificial sentences, allowing learners to generalize correctly to longer sentences than they were exposed to in training. The question of such a bias is a long-standing controversy addressed by many human artificial-language learning experiments, which Tom submitted to an extensive review. In an 11-page Appendix to [4], he summarized 8 previous experiments and, in a 6-page table, assessed these and many more for 10 specific flaws. His experiment is the only one to escape all these faults; it provides the first solid evidence for a hierarchical bias in human learners.

(1″) Tom's work [14] shows that some (but not other) standard recurrent nMLs (GRUs) actually tend to generalize to correct hierarchically-governed question formation and subject-verb agreement, despite having been trained only on examples that are equally consistent with linear, non-hierarchically-governed rules. His further work [5] showed that nMLs in which the connections between layers of neurons are arranged to form a tree structure robustly generalize in this hierarchical fashion.

(2) ***nMLs do lack important elements of syntactic competence.*** In a highly influential paper [12], Tom examined models that scored highly on a widely-accepted benchmark testing the ability to determine whether one sentence logically entails another (NMLI). He showed these models lacked a basic ingredient in syntactic competence, sensitivity to word order. They believed that "Kim saw Sandy" entailed "Sandy saw Kim", and achieved success on the benchmark by exploiting statistical quirks in the test to game it by using superficial heuristics that work on the large majority of test cases (such as "sentence 1 entails sentence 2 if they contain many of the same words"). Tom generated a new test set, HANS, which tests sensitivity to word order and other properties ignored by shallow statistical generalizations over the training data. Performance of the best model, BERT, dropped from 84% to 16% (where chance performance is 50%).

(2′) ***nMLs' generalization out of their training distribution is very non-robust.*** In fact, Tom then showed [8] that BERT's performance on HANS varied wildly depending on the random seed used in the simulation: from 0% to 66%. This is a particularly dramatic demonstration of how, when tested on examples that do not fall within the statistical distribution from which the training set is drawn, nMLs' performance is highly underdetermined; robust generalization would require considerably stronger inductive biases than standard nMLs possess.

(3) ***nMLs with standard architectures learn hidden symbolic representations: Tensor Product Representations (TPRs).*** For compositional mappings between symbol sequences — including several simple functions [13] and a more complex, well-studied function designed by others as a compositionality challenge for nMLs (SCAN) [7] — Tom showed that standard RNN (GRU) models learn intermediate representations of input sequences that are, up to a linear transformation, extremely well approximated by Tensor Product Representations, which explicitly embed the compositional organization of symbol structures in vector spaces. This provides a closed-form algebraic expression for the hidden state representing any input. This in turn makes it possible to compute the exact manipulation of the hidden state (altering the activation of all neurons by a precise amount) in order to control the model's output. Thus, the identified compositional TPR structure hidden in the hidden state is causally responsible for the models' behavior. (See https://rtmccoy.com/tpdn/tpr_demo.html for Tom's demo.)

(4) ***nMLs' abilities to generalize correctly outside their training distribution can indeed be strengthened by giving them appropriate biases.*** This is illustrated by work Tom has carried out deploying all 4 methods listed in (5).

(5a) ***nMLs' generalization becomes more syntactically sound when their training data is augmented by syntactic transformations.*** This method boosts BERT's performance on HANS from 28% to 73% [11].

(5b) ***nMLs' learning is strengthened by providing them operations for building hidden symbolic structure.*** When the Transformer architecture is provided with the operators required to explicitly build TPRs, introduced in (3), their capacity for out-of-distribution compositional generalization increases markedly: for a simple symbol sequence-mapping task, the size of training set needed shrinks by an order of magnitude, and the rate at which models learn the task perfectly rises from 0% to over 70% [2].

(5c) ***nMLs' learning is boosted by rewarding representations that contain hidden symbolic structure.*** The methods described in (3) make it possible to define an objective function for learning which evaluates the degree to which an internal representation has a particular hidden compositional structure. This enables an inductive bias that greatly boosts out-of-training-distribution generalization on linguistic mappings demanding correct syntactic structure [1]. The bias towards such structured representations is only present during training; during testing, the trained model has no such information.

(5d) ***nMLs can be given an initial state that positions them to efficiently learn classes of languages governed by soft universal constraints.*** A prominent view in linguistics is that the initial state of human language learners encodes knowledge about possible human languages that biases them to quickly home in on such languages from relatively sparse data. Ultimate success of the nML framework would seem to disprove this view. Yet Tom has shown [9] that in one such parade case — learning how the target language builds the syllable structure for a phoneme string — meta-learning can find an initial set of weights for a standard nML architecture which enables the architecture to rapidly learn any language in a set of languages sampled during meta-learning. This initial state provides

rapid learning for a set of languages not seen during meta-learning that respect the cross-linguistic generalizations arising from a set of soft constraints on syllable structure posited by Optimality Theory. This work opens an entire new field of research on meta-learning of universal language typologies. (See his demo: https://rtmccoy.com/meta-learning-linguistic-biases.html.)

QUALIFICATIONS FOR A FACULTY POSITION

Interdisciplinarity. It should be clear from even this brief summary of several of Tom's research projects that he puts his broad and deep knowledge of linguistics and cognitive science to good use. He is among the few researchers who combine such knowledge with great skill in developing nMLs. The rate at which he produces new ideas to test, and experimental work to test them, is astonishing. This in fact goes beyond nML techniques. In current work pursuing project (1′), he quickly mastered Bayesian modeling of language learning to the point of being able to design and implement the first such model for a new grammar class (which he defined within a probabilistic version of the PATR-II class). His artificial language learning study of recursive center embedding (1′) relies crucially on agreement relations spanning the central embedding site. He judged the classic Bayesian model of Perfors et al. 2020 insufficient for analyzing this experiment because the hypothesis space it assumes, simple CFGs, requires much complexity to handle such agreement; his PATR-II class, in contrast, enables a compact treatment of agreement, so that a Bayesian prior favoring simpler grammars is less biased against non-recursive grammars. (In addition to the search within the hypothesis space required of any Bayesian learner, this work also demands a search over alternative hypothesis spaces.)

Tom also makes creative use of mathematics. He has good command of statistical analysis of experimental data. And although I have worked on TPRs for years, I have learned a number of important things about them from Tom, things that genuinely surprised me. Entirely on his own, he conducted insightful analyses of how recurrent neural networks (RNNs) can compute TPRs, how TPRs can be usefully generalized from vector to matrix embeddings of structural roles, and how a great diversity of seemingly unrelated NN representational schemes are cases of linearly-transformed TPRs [1].

Tom has led several large interdisciplinary research groups, combining theoretical and experimental linguists with academic and industrial-lab NLP researchers. I have found him to be an outstanding collaborator.

Productivity. Tom works very smart, but also very hard. He has published 18 reviewed papers, most in major conferences (4 at *ACL*, 3 at *CogSci*, 2 each at *ICLR* and *BlackboxNLP*, and 1 at each of *COLING*, *EMNLP*, the *Society for Computation in Linguistics* (*SCiL*), *LREC*, and *the TAG workshop*, as well as a best paper at **SEM* 2019 and a journal article in *TACL* 2020). There is also a review and perspective paper to appear in *AI Magazine* [2], and several other papers very close to completion and submission ([1] and [3], among others).

Tom also works very fast, as evident from the number of papers he has already produced. A sample anecdote: Tom presented a paper (on others' work) to a recent meeting of my research lab, and it happened that 24 hours prior to this meeting he mentioned he'd not yet started to read the paper, which was a deep, creative analysis showing how to characterize Transformer neural networks as implementing interpretable programs in a language the authors invented. Tom's presentation was brilliant: 37 excellent slides presenting this difficult paper very clearly. He then went on to 6 further slides presenting two new programs he had written in this new language, to perform English past-tense inflection and subject-auxiliary inversion. He had gotten an interpreter of the language running and showed sample runs of his programs. All prepared in one day.

Clarity. Tom is highly analytical by nature, and his work always reflects extensive thought about the big-picture questions at stake, the precise hypothesis to test, the manifold decisions to be made in designing particular studies and implementing experiments, both with human and machine subjects (see the discussion in (1') above for one example); his testing of models is truly exhaustive, often in ways I hadn't even imagined. This analytical rigor makes his presentations of his work — in writing, speaking, and on-line demos — extremely clear. Even as a second-year graduate student, he gave an invited lecture at Microsoft Research that was praised for its clarity and importance.

Funding. Tom received an NSF Graduate Fellowship as well as an instructional-software development grant (with Tal Linzen); he has, since college, continuously and tirelessly sought external funding, only some of which appears on his CV. He received a Microsoft Research Internship and has in-depth experience working within an industrial research environment.

Teaching. Tom has valuable teaching experience, as documented in his CV. He was a wonderful TA in my course *Foundations of Cognitive Science*, despite it being taught remotely. This required grading over 30 written assignments, and he kept on top of that flood extremely well. He did an excellent job leading a class on one of my papers, with 1 hour's notice. Tom was also an outstanding TA in *Syntax I* and in an introductory course, *The World of Language*, where he led 2 weekly fieldwork sessions, teaching those beginning students the additional linguistic theory they needed along the way. While Tom was producing all his research papers, he taught during 5 semesters: each time a different course, many with heavy workloads like my *Foundations* class, several others requiring creation of instructional software.

Mentoring. Tom has been unusually active in supervising undergraduate and master's student research, resulting in 3 publications to date: [6], [7] and [10].

Service and diversity-promotion. Tom has contributed generously to the service needs of his community. He regularly makes presentations to research labs and provides thoughtful feedback to other presenters; he has initiated virtual celebrations and gifts for departing lab members. He has represented the department in the Graduate Student Association and actively participated in annual graduate-student recruitment efforts. Tom has been especially committed to efforts to promote diversity. To support the careers of first-generation college students, he co-organized a mentorship program for applicants to the graduate program. He also co-wrote a statement, added to syllabi in department courses, to raise awareness about how to pursue research in college and beyond, and to address the lack of diversity in course reading lists. Tom's outreach activities in Baltimore public schools, most of which are low-income and minority-serving, and long-time dedicated service to the North American Computational Linguistics Olympiad (NACLO), attest to his desire to bring a diverse set of newcomers into computational linguistics.

Puzzles. Tom's effervescent creativity feeds his pastime of puzzle-creation, including many NACLO problems derived from his own research, and 33 crossword puzzles published in the *New York Times*. He gave an invited lecture at the *National Museum of Language* entitled "Language squared: The linguistics of crosswords" [18].

Intellect. Tom is unsurpassed in intellectual talent by anyone I have known in my career. Even in college, this was attested by his award-studded career at Yale, graduating *summa cum laude* in Linguistics with a perfect grade-point average of 4.0. I find talking with him about foundational issues in our field so rewarding that I meet one-on-one with him every week, solely for this purpose; this meeting is the highlight of my week, and I often stretch it well beyond the scheduled hour. Tom was instrumental in making the forthcoming perspective piece in *AI Magazine* what it is; his extensive critique of my first draft sent me right back to the drawing board, and he wrote much of the final draft himself.

Personally, Tom is a delight. Despite his achievements and brilliance, he is completely down-to-earth and not the least bit cocky or arrogant. He will complete his Ph.D. by the end of Summer 2022. I will miss him terribly.

There can be little doubt that Richard Thomas McCoy will continue to be a brilliant star in computational linguistics throughout his career. He will greatly strengthen whichever university is fortunate enough to recruit him.

Sincerely,

Paul Smolensky

Krieger-Eisenhower Professor of Cognitive Science, Johns Hopkins University
Partner Researcher, Deep Learning Group, Microsoft Research, Redmond

References

[1] In prep. McCoy et al. DISCOVER: A framework for dissecting compositionality in vector representations.

[2] To appear. Smolensky, McCoy, Fernandez, Goldrick, Gao. Neurocompositional computing: From the central paradox of cognition to a new generation of AI systems. *AI Magazine*.

[3] 2021 McCoy, Smolensky, Linzen, Gao, Celikyilmaz. How much do language models copy from their training data? Evaluating linguistic novelty in text generation using RAVEN. https://arxiv.org/abs/2111.09509 (10 pages of main text + 39 pages of Appendices; journal submission pending)

[4] 2021 McCoy, Culbertson, Smolensky, Legendre. Infinite use of finite means? Evaluating the generalization of center embedding learned from an artificial grammar. *Cognitive Science Society*.

[5] 2020 McCoy, Frank, Linzen. Does syntax need to grow on trees? Sources of hierarchical inductive bias in sequence-to-sequence networks. *TACL*.

[6] 2020 Lepori, McCoy. Picking BERT's brain: Probing for linguistic dependencies in contextualized embeddings using representational similarity analysis. *COLING*.

[7] 2020 Soulos, McCoy, Linzen, Smolensky. Discovering the compositional structure of vector representations with Role Learning Networks. *Third BlackboxNLP Workshop*.

[8] 2020 McCoy, Min, Linzen. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. *Third BlackboxNLP Workshop*.

[9] 2020 McCoy, Grant, Smolensky, Griffiths, Linzen. Universal linguistic inductive biases via meta-learning. *Cognitive Science Society*.

[10] 2020 Lepori, Linzen, McCoy. Representations of syntax [MASK] useful: Effects of constituency and dependency structure in recursive LSTMs. *ACL*.

[11] 2020 Min, McCoy, Das, Pitler, Linzen. Syntactic data augmentation increases robustness to inference heuristics. *ACL*.

[12] 2019 McCoy, Pavlick, Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. *ACL*.

[13] 2019 McCoy, Linzen, Dunbar, Smolensky. RNNs implicitly implement tensor-product representations. *ICLR*.

[14] 2018 McCoy, Frank, Linzen. Revisiting the poverty of the stimulus: Hierarchical generalization without a hierarchical bias in recurrent neural networks. *Cognitive Science Society*.

[15] Invited lecture at the *National Museum of Language*: "Language squared: The linguistics of crosswords"; talk: https://www.youtube.com/watch?v=r5Lhir9eKyY, highlights: https://www.youtube.com/watch?v=UwbO8HMrOq4 (a related story about Tom in the *Washington Post:* https://www.washingtonpost.com/education/2019/12/27/puzzling-over-language-computational-linguistics-researcher-crafts-crossword-johns-hopkins/#comments-wrapper).

FACEBOOK

1 Hacker Way
Menlo Park, CA 94025
United States

November 27, 2021

Dear Member of the Hiring Committee,

I am thrilled to provide my most enthusiastic support for **R Thomas McCoy**, for a tenure track academic position at your institution. I am writing this letter as his Ph.D. student research intern mentor at Deep Learning and the NLP team at Microsoft Research and later as his collaborator[1]. Tom is among the very best Ph.D. students I have worked with who has a very unique background (Cognitive Science and Natural Language Processing) and amazing technical and teaching experience. I recommend hiring him in the strongest terms possible.

I have worked with Tom extensively and know his exceptional research capabilities first-hand. Tom has published several top-tier conference papers, and he is the first author for several papers. When I was searching for a Ph.D. intern for my research project on analyzing text generation systems, I was looking to work with someone who has a linguistic background as well as fluent in state of the art neural models. Tom fit into this position perfectly. Tom's career interests have always been squarely in academia, despite his well-rounded intellectual strengths that would bode well if he were to pursue a career in industry labs. Nevertheless, he has done phenomenal research work during his internship and after and formed amazing collaborators with everyone in the team. Tom is among the most intellectually independent and autonomous students I have worked with, overall requiring considerably less low-level help from me compared to many others. For one, if there are new technical concepts, methods, algorithms, or codes he needs to learn, he would just go off and learn them on his own, and soon be able to implement them and experiment with them with no apparent difficulties. While implementing some of the algorithms

[1] I am currently a research scientist at Facebook AI Research @MetaAI in Seattle and an Affiliate Associate Member at the University of Washington focusing on fundamental NLP problems. Formerly, I was Senior Principal Researcher at Microsoft Research (MSR) in Redmond, Washington. I earned my Ph.D. Degree in Information Science from University of Toronto, Canada, and later continued my Postdoc study at the Computer Science Department of the University of California, Berkeley. My research interests are mainly in deep learning and natural language, specifically on language generation with long-term coherence, language understanding, language grounding with vision, and building intelligent agents for human-computer interaction I am serving on the editorial boards of Transactions of the ACL (TACL) as area editor and Open Journal of Signal Processing (OJSP) as Associate Editor.

during his internship, he found and fixed a bug in Hugging Face's Transformer-XL (one of the transformer based Language Model). Huggingface's engineers have later accepted Tom's fix to their main repo. He has done numerous experiments running the latest language models including (GPT-2 and Transformer-XL). After his internship was over, we have continued our collaboration and he has completed all his experiments and even tried new ones to strengthen our hypothesis. He has written an extensive research work[2], and we are submitting it to a top NLP Journal). This work has been already recognized by several NLP groups and has been selected as the paper of the week[3]. While technically proficient and competent, Tom stands out especially for his ability to drive original research addressing open-ended research questions that require novel conceptual frameworks and/or nontrivial data preparation. While I often see students who are strong at the former or the latter, it is rather uncommon to see students who are equally talented at both.

Tom has worked with me and others in the team on understanding if and how neural language models learn to generate novel text. For this analysis, he not only needed to collect an extensive amount of data, but also build several baseline models using different types of the neural language models. He has no trouble doing either of them. At each week's team meeting, Tom always had a great deal of progress to report. He initiated his own original experiment designs based on diagnoses he developed himself to understand what problems needed addressing. He expanded the scope of the project to new datasets on his own initiative, and carried out additional interpretive analyses that he discovered independently. His presentations of new results were always lucid, with low-level details and high-level conclusions both presented clearly; these meetings were efficient and effective, as he came well prepared with useful discussion points. This is all particularly impressive for the reason that introducing new criteria to investigate language model outputs has not been investigated in the past and Tom had no trouble presenting new and useful angles. Even though Tom has not worked on long-form text generation research before, he came up to speed on different architectures of neural language models very quickly, reading a lot of relevant papers and running the baseline models, and variants, and the testing pipeline, all very efficiently. He was showing impacts of the new evaluation criteria that he introduced astonishingly quickly. His work is the first evidence showing that the large language models do indeed generate novel text.

Tom's great success in the project was remarkable for another reason. He got on board as an intern virtually, early in the pandemic, and while working remotely slowed down quite a few interns, Tom was capable of producing

[2] https://arxiv.org/pdf/2111.09509.pdf
[3] https://twitter.com/fbk_mt/status/1464186165532180482?s=20

impressive results nonetheless. Tom is a marvelous team player. He collaborated with other interns, advising them and giving new directions where necessary. During the paper writing process of another intern's project, which was on machine translation, Tom has taken a lot of responsibility and provided amazing new angles that have helped shape the paper into a strong one, which is later accepted for publication. His contribution to this other project has made a huge impact on the success and eventually ended up with a peer-reviewed publication.

The area that Tom proposes to continue to pursue during his academic work is one that I think is absolutely central to making progress in AI: robust generalization in models of language. Together with his great talent and amazing work ethic, I think this means that Tom is an all-round superb candidate. Thus, I provide my most enthusiastic recommendation for R Thomas McCoy for this position at your institution.

Please feel free to contact me at aslic@fb.com, should you have any questions.

Asli Celikyilmaz
Principal Research Scientist
Facebook AI Research
Seattle, WA, USA

MIT COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE LABORATORY

December 1, 2021

Jacob Andreas
The Stata Center, Building 32-386H
32 Vassar Street Cambridge, Massachusetts USA 02139-4307
jda@mit.edu
web.mit.edu/jda/www

To whom it may concern,

I'm writing to give Tom McCoy my **strong recommendation** for a faculty position. Tom is currently a PhD student at Johns Hopkins, and I'm the outside member of his doctoral committee.[1] While we've never collaborated directly, I've followed Tom's research closely for many years. He's doing some of the most exciting current work on *explaining* the successes and limitations of deep network models for natural language processing. His papers have in several instances influenced my own choice of projects; the questions he studies contain the material for a research program that's likely to long outlast the current generation of machine learning approaches in NLP. He compares favorably to a number of recent PhD graduates now in tenure-track faculty positions. You should hire him!

Tom's research: Natural language processing, like much of the rest of artificial intelligence, is currently in a state of crisis. Our best results in almost every task and problem domain are obtained by neural network models whose internal decision-making processes are opaque. Standard benchmark datasets and evaluation metrics show "human-level performance" from learned models for question answering, translation, and summarization, but anyone who's spent ten minutes interacting with these systems knows that they are easily fooled—failing in unpredictable and decidedly un-human-like ways. In short, we understand neither *how* current approaches to NLP work nor even *when* they work, and in what contexts they are likely to make incorrect predictions.

Tom's work approaches this failure of understanding as a scientific challenge, applying the experimental apparatus of cognitive science to better characterize both the behavior and inner workings of the current modeling toolkit. I expect that his other letters will describe his specific projects in more detail, but I want to highlight two that I think are particularly important. In *RNNs Implicitly Implement Tensor Product Representations*, he provides a remarkably fine-grained mechanistic account of the computations performed by recurrent neural networks (RNNs), relating behavior (the specific input–output mappings implemented by trained models) to implementation (the compositional structure of the RNN-internal representations), along the way introducing a general-purpose framework for analyzing neural representations that can be (and has been) applied to other model architectures and training objectives. In *Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference*, he identifies and explains a set of (incorrect)

[1] A bit about myself: I'm a faculty member at MIT in in the Electrical Engineering and Computer Science department and the Computer Science and Artificial Intelligence Laboratory. My research interests include a broad set of topics in natural language processing and machine learning, and my work has been recognized with academic and industry awards in both the computational linguistics and machine learning communities. Lately, my students have focused on two research themes: *generalization* in models for language processing (understanding what language learning algorithms can extract from limited datasets and why they make the mistakes they do) as well as the use of language as an *explanatory tool* and *supervisory signal* in other machine learning problems (especially computer vision and robotics).

inferential heuristics implemented by language understanding models, in this case relating behavior to training data rather than internal computation. As discussed below, both studies have contributed meaningfully to our understanding of the limitations of current neural models in NLP; what I think is more important is the broader style of empirical investigation that these papers represent, and which I think Tom deserves a lot of credit for popularizing.

Where he's had impact: Both of the papers above have been extremely influential. The analysis technique introduced by the paper on tensor product representations provided the core analytical technique underlying *What does BERT learn about the structure of language?* (Jawahar et al. 2019), which in turn helped gave rise to an entire new sub-field of NLP (now a reasonably large fraction of all NLP research) dedicated to studying linguistic representations implicit in trained models. Tom's work on inference heuristics is also important, widely cited, and has clearly played a role in shaping how new datasets (for natural language inference and other tasks) are designed and annotated.

I can also offer two personal perspectives on the impact of Tom's work. The first has to do with its implications for work on interpretable machine learning outside of NLP. In the last year or so, part of my research has focused on developing general-purpose tools for identifying the functions of individual neurons and small circuits in deep networks. When looking for applications of these techniques to NLP, we needed a task in which models' failure modes were well understood but the mechanism underlying them still unknown. Here Tom had paved the way for us—his work on heuristics in natural language inference provided a phenomenon to explain and a set of hypotheses about where the behavior originated, leaving only he question of how it was implemented. Our eventual set of experiments wouldn't have been possible to formulate, much less execute, without Tom's prior theory-building.

Second, last spring, I taught a graduate seminar on neuro-symbolic methods for natural language processing. The course featured a mix of both classic (80s and 90s) connectionist models and modern neural techniques. One of the big challenges in putting together the reading list was figuring out how to bridge the gap between these two periods of research, and especially laying out how insights from symbolic models of cognition might be integrated into neural approaches. Tom's research was invaluable for this. I assigned multiple of his papers as readings (both the tensor product paper mentioned above, and a paper on meta-learning of phonological rules) to help students map out the landscape of implicit and explicit approaches to symbol processing in deep models. This reflects his *depth* as a researcher—he's produced work that's ready to be taught as exemplary of the cutting edge of the field. But it also reflects his *breadth*—he showed up multiple times in a class designed to expose students to a diversity of approaches spanning multiple academic disciplines and time periods.

Why his research matters: The central role of neural network models for natural language processing is unlikely to disappear any time soon. But we're never going to theorize or engineer our way around the limitations of the toolkit as it exists today without a much deeper scientific (and especially empirical) understanding of our current modeling toolkit—its inductive biases, fundamental computational limitations, and relationship to human sentence processing. On each of these questions, Tom's research plan offers concrete next steps and a long-term plan. I expect that analytical work of the kind Tom excels at will be one of the most important drivers of progress in NLP in the coming years.

Where he ranks: I'd place Tom in a category with several prominent junior faculty who work at the boundaries between NLP and other fields, including Mohit Iyyer and Diyi Yang (UMass and Georgia Tech; both working in computational social science) and Jesse Thomason and Yonatan Bisk (USC and CMU; computer vision and robotics). I should emphasize that my model of him is formed almost entirely

on the basis of work that he publishes at NLP and machine learning venues (though as noted above I've also taught one of his CogSci papers); take this as some evidence that he's done a full PhD's worth of work in NLP *in addition* to his human experimental work.

In summary: Tom is great—doing important research that brings methodological clarity and experimental rigor into NLP, with a history and likely future of high impact. He's creative, technically skilled, and a great fit for your program; I urge you to give him your strong consideration.

Please do not hesitate to contact me if there are any other questions I can answer.

Warm regards,

Jacob Andreas
X Consortium Career Development Assistant Professor
Department of Electrical Engineering and Computer Science
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology