RISSANEN DATA ANALYSIS: EXAMINING DATASET CHARACTERISTICS VIA DESCRIPTION LENGTH

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a method to determine if a certain capability helps to achieve an accurate model of given data. We view labels as being generated from the inputs by a program composed of subroutines with different capabilities, and we posit that a subroutine is useful if and only if the minimal program that invokes it is shorter than the one that does not. Since minimum program length is uncomputable, we instead estimate the labels' minimum description length (MDL) as a proxy, giving us a theoretically-grounded method for analyzing dataset characteristics. We call the method Rissanen Data Analysis (RDA) after the father of MDL, and we showcase its applicability on a wide variety of settings in NLP, ranging from evaluating the utility of generating subquestions before answering a question, to analyzing the value of rationales and explanations, to investigating the importance of different parts of speech, and uncovering dataset gender bias.

1 INTRODUCTION & RELATED WORK

In many practical learning scenarios, it is useful to know what capabilities would help to achieve a good model of the data. According to Occam's Razor, a good model is one that provides a simple explanation for the data (Blumer et al., 1987), which means that the capability to perform a task is helpful when it enables us to find simpler explanations of the data. Kolmogorov complexity (Kolmogorov, 1968) formalizes the notion of simplicity as the length of the shortest program required to generate the labels of the data given the inputs. In this work, we estimate the Kolmogorov complexity of the data by approximately computing the data's Minimum Description Length (MDL; Rissanen, 1978), and we examine how the data complexity changes as we add or remove different features from the input. We name our method Rissanen Data Analysis (RDA) after the father of MDL, and we use it to examine several open questions about popular datasets.

We view a capability as a function $f(x)$ that transforms x in some way (e.g., adding a feature), and we say f is helpful if invoking it leads to a shorter minimum program for mapping x to the corresponding label in a dataset. Finding a short program is equivalent to finding a compressed version of the labels given the inputs, since the program can be run to generate the labels. Thus, we can measure the shortest program's length by estimating the labels' maximally compressed length, or Minimum Description Length (MDL; Rissanen, 1978; Grünwald, 2004). While prior work in machine learning uses MDL for model optimization (Hinton & van Camp, 1993), selection (Yogatama et al., 2019), and probing (Voita & Titov, 2020; Whitney et al., 2020; Lovering et al., 2021), we use MDL for a very different end: to understand the data itself ("dataset probing").

RDA addresses empirical and theoretical inadequacies of prior data analysis methods. For example, two common approaches are to evaluate the performance of a model when the inputs are modified or ablated (1) at training and test time or (2) at test time only. Training time input modification has been used to evaluate the usefulness of the capability to decompose a question into subquestions (Min et al., 2019b; Perez et al., 2020), to access the image for image-based question-answering (Antol et al., 2015; Zhang et al., 2016a), and to view the premise when detecting if it entails a hypothesis (Gururangan et al., 2018; Poliak et al., 2018; Tsuchiya, 2018). However, these works evaluate performance only on held-out dev examples, a fraction of the total examples in the dataset, which also are often drawn from a different distribution (e.g., in terms of quality). To understand what datasets teach our models, we must examine the entire dataset. Test time ablation has been used to evaluate the capability to view word order (Pham et al., 2020; Sinha et al., 2020; Gupta et al., 2021) or words of different

types (Sugawara et al., 2020), or to perform multi-hop reasoning (Jiang & Bansal, 2019). However, it is hard to rule out factors that may explain poor performance (e.g., distribution shift) or good performance (e.g., other ways to solve a problem). Here, we examine an intrinsic property of the dataset, MDL, and we provide a theoretical argument justifying why it is the correct measure to use.

We use RDA to provide insights on a variety of datasets. In §3.1, we examine a benchmark for answering questions where prior work has claimed that decomposing questions into subquestions is helpful (Min et al., 2019b; Perez et al., 2020) and called such claims into question (Min et al., 2019a; Jiang & Bansal, 2019; Chen & Durrett, 2019). RDA shows that subquestions are helpful and exposes how evaluation methods in prior work may have caused the value of subquestions to be underestimated. In §3.2, we evaluate if explanations are useful for recognizing textual entailment. Written explanations and decision-relevant keywords ("rationales") are both helpful, and rationales are more helpful than explanations. We examine a variety of popular NLP tasks (Appendix §A), evaluating the extent to which they require relying on word order, different types of words, and gender bias. Overall, our results show that RDA can answer a broad variety of questions about datasets.

2 Rissanen Data Analysis

How can we determine whether or not a certain capability $f(x)$ is helpful for building a good model of the data? To answer this question, we view a dataset with inputs $x_{1:N}$ and labels $y_{1:N}$ as generated by a program that maps $x_n \rightarrow y_n$. Let the length of the shortest such program P be $\mathcal{L}(y_{1:N}|x_{1:N})$, the data's Kolmogorov complexity. We view a capability as a function f that maps x_n to a possibly helpful output $f(x_n)$, with $\mathcal{L}(y_{1:N}|x_{1:N}, f)$ being the length of the shortest label-generating program when access to f is given. We say that f is helpful exactly when $\mathcal{L}(y_{1:N}|x_{1:N}, f) < \mathcal{L}(y_{1:N}|x_{1:N})$.

2.1 Minimum Description Length

The above inequality requires us to find the shortest program P, which is uncomputable in general. However, because P is a program that generates $y_{1:N}$ given $x_{1:N}$, we can instead consider any compressed version of $y_{1:N}$, along with an accompanying decompression algorithm that produces $y_{1:N}$ given $x_{1:N}$ and the compressed $y_{1:N}$. To find $\mathcal{L}$, then, we find the length of the maximally compressed $y_{1:N}$, or Minimum Description Length (MDL; Rissanen, 1978). While MDL is not computable, just like Kolmogorov complexity, many methods have been proposed to estimate MDL by restricting the set of allowed compression algorithms (see Grünwald, 2004, for an overview). Here, we use online coding (Rissanen, 1984; Dawid, 1984), which is effective for estimating MDL when used with deep learning (Blier & Ollivier, 2018; Yogatama et al., 2019; Voita & Titov, 2020).

2.2 Online Coding

To examine how much $y_{1:N}$ can be compressed, we look at the minimum number of bits (minimal codelength) needed by a sender Alice to transmit $y_{1:N}$ to a receiver Bob, when both share $x_{1:N}$. Without loss of generality, we assume y_n is an element from a finite set. In online coding, Alice first sends Bob the learning algorithm $\mathcal{A}$, including the model architecture, trainable parameters θ, optimization procedure, hyperparameter selection method, initialization scheme, random seed, and pseudo-random number generator. Alice and Bob each initialize a model p_{θ_1} using the random seed and pseudo-random number generator, such that both models are identical.

Next, Alice sends each label y_n one by one. Shannon (1948) showed that there exists a minimum code to send y_n with $-\log_2 p_{\theta_n}(y_n|x_n)$ bits when Alice and Bob share p_{θ_n} and x_n. After Alice sends y_n, Alice and Bob use $\mathcal{A}$ to train a better model $p_{\theta_{n+1}}(y|x)$ on $(x_{1:n}, y_{1:n})$ to get shorter codes for future labels. The codelength for $y_{1:N}$ is then:

$$\mathcal{L}_p(y_{1:N}|x_{1:N}) = \sum_{n=1}^{N} -\log_2 p_{\theta_n}(y|x). \tag{1}$$

Overall, Alice's message consists of $\mathcal{A}$ plus the label encoding ($\mathcal{L}_p(y_{1:N}|x_{1:N})$ bits). When Alice and Bob share f, Alice's message consists of $\mathcal{A}$ plus $\mathcal{L}_p(y_{1:N}|x_{1:N}, f)$ bits to encode the labels with a model $p_\theta(y|x, f)$. f is helpful when it shortens the message: $\mathcal{L}_p(y_{1:N}|x_{1:N}, f) < \mathcal{L}_p(y_{1:N}|x_{1:N})$.

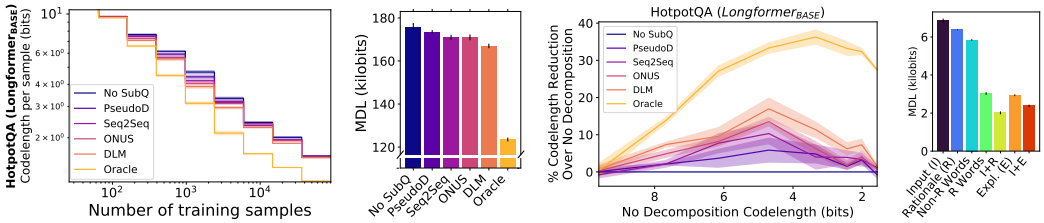

Figure 1: **First**: Codelengths for HOTPOTQA with the answers to sub-questions from various methods. **Second**: MDL for decomposition methods, which consistently help to compress the answers. **Third**: On HOTPOTQA, the reduction in codelength over the no-decomposition baseline from using sub-answers from various decomposition methods. **Fourth**: On e-SNLI, adding rationales and explanations greatly reduces MDL. (Mean and std. error across 5 runs for all plots.)

2.3 PRACTICAL IMPLEMENTATION WITH BLOCK-WISE CODING

The online code in Eq. 1 is expensive to compute. It has a computational complexity that is quadratic in N (assuming linear time learning), which is prohibitive for large N and compute-intensive $\mathcal{A}$. Following Blier & Ollivier (2018), we upper bound online codelength by having Alice and Bob only train the model upon having sent $0 = t_0 < t_1 < \cdots < t_S = N$ labels. Alice thus sends all labels in a "block" $y_{t_s+1:t_{s+1}}$ at once using $p_{\theta_{t_s}}$, giving codelength:

$$\bar{\mathcal{L}}_p(y_{1:N}|x_{1:N}) = \sum_{s=0}^{S-1} \sum_{n=t_s+1}^{t_{s+1}} -\log_2 p_{\theta_{t_s}}(y_n|x_n) \tag{2}$$

Since θ_{t_0} has no training data, Alice sends Bob the first block with a uniform prior. We use Eq. 2 in practice to evaluate when $f(x)$ reduces program length (implementation details in Appendix §C).

3 EXAMINING DATASET CHARACTERISTICS

3.1 IS IT HELPFUL TO ANSWER SUBQUESTIONS?

Yang et al. (2018) proposed HOTPOTQA as a dataset that benefits from decomposing questions into subquestions, but recent work has questioned such benefit (Min et al., 2019a; Jiang & Bansal, 2019; Chen & Durrett, 2019) while there is also evidence that decomposition helps (Min et al., 2019b; Perez et al., 2020). We use RDA to evaluate if subquestions and their answers ("subanswers") are useful.

Experimental Setup HOTPOTQA consists of crowdsourced questions (*"Are Coldplay and Pierre Bouvier from the same country?"*) whose answers are intended to rely on information from two Wikipedia paragraphs. The input consists of these two "supporting" paragraphs, 8 "distractor" paragraphs, and the question. Answers are either `yes`, `no`, or a text span in an input paragraph. As our model, we use LONGFORMER$_{\text{BASE}}$ (Beltagy et al., 2020), a transformer (Vaswani et al., 2017) modified to handle long inputs as in HOTPOTQA (details in Appendix §D.3). We consider a subanswer to be a paragraph containing question-relevant information, as Perez et al. (2020) claimed that subquestions help by using a QA model to find question-relevant text. We indicate up to 2 subanswers to the model by prepending "$>$" to the first subanswer paragraph and "$\gg$" to the second.

Methods for Selecting Subanswers We use the two supporting paragraphs as oracle subanswers. We also consider the answers to subquestions generated by 4 methods (details in Appendix §D.1). To answer generated subquestions, we use the ROBERTA$_{\text{LARGE}}$ (Liu et al., 2019) ensemble from Perez et al. (2020). We use the paragraphs containing predicted answer spans to subquestions as subanswers.

3.1.1 RESULTS

Fig. 1 shows codelengths (first) and MDL (second). Decompositions consistently reduce codelength and MDL. Decomposition methods reduces MDL to varying extents, ranked worst to best as: no

decomposition, Pseudo-Decomposition, Seq2Seq, ONUS, DLM, and oracle. Overall, the capability to answer subquestions reduces program length, in a way that depends on subquestion quality.

To understand when decompositions reduce codelength, we plot the codelength reduction from decomposition against the original codelength for LONGFORMER$_{\text{BASE}}$ in Fig. 1 (third). As the original codelength decreases, the benefit from decomposition increases, until the no-decomposition baseline reaches a certain loss, at which point the benefit from decomposition decreases. We hypothesize that a certain, minimum amount of task understanding is necessary before decompositions are useful. However, as loss decreases, the task-relevant subroutines can be learned from the data directly, without decomposition. The limited value in low-loss regimes occurs because models approach the same, minimum loss $H(y|x)$ in the limit of dataset size. Our observation partly explains why a few earlier studies (Min et al., 2019a; Jiang & Bansal, 2019; Chen & Durrett, 2019), which only evaluated final performance, drew the conclusion that HOTPOTQA does not benefit much from multi-step reasoning or question decomposition. In contrast, MDL *does* capture differences in performance across data regimes, demonstrating that RDA is the right approach going forward.

3.2 ARE EXPLANATIONS AND RATIONALES USEFUL?

Recent work proposes to generate reasons before predicting an answer to achieve better accuracy (e.g., Rajani et al., 2019; Wiegreffe et al., 2020). These studies often test on natural language inference (NLI), a task that involves determining if a premise entails or contradicts (or neither) a hypothesis. Using NLI as a testbed, we use RDA to evaluate if providing a reason is a useful capability.

Dataset and Model We use e-SNLI (Camburu et al., 2018), which annotated each example in SNLI (Bowman et al., 2015) with (1) an extractive rationale that marks entailment-relevant words and (2) a written explanation of the right answer. We randomly sample 10k examples from e-SNLI to examine the usefulness of rationales and explanations. As our model, we use an ensemble of 10 model classes – a FastText bag-of-words model and 9 transformer variants (details in Appendix D.5).

Adding explanations and rationales We add the rationale by surrounding each entailment-relevant word with asterisks, and we add the explanation before the hypothesis, separated by a special token. For comparison, we also evaluate MDL when including only the explanation or rationale patterns as input. For the latter, we include the rationale without the actual premise and hypothesis words by replacing each rationale word with "*" and other words with "_".

3.2.1 RESULTS

Fig. 1 (rightmost) shows MDL. Adding rationales greatly reduces MDL over using the normal input ("Input (I)") or rationale markings without input words ("Rationale (R)"), suggesting that rationales complement the input. The reduction comes from focusing on rationale words specifically. We see almost as large MDL reductions when only including rationale-marked words and masking non-rationale words ("R Words" vs. "I+R"). In contrast, we see little improvement over rationale markings alone when using only non-rationale words with rationale words masked ("Rationale (R)" vs."Non-R Words"). Our results show that a useful subroutine for NLI is to locate task-relevant words, suggesting directions for future work similar to Zhang et al. (2016b); Perez et al. (2019).

Similarly, explanations greatly reduce MDL (Fig. 1 right, rightmost two bars), especially when the input is also provided. Explanations, like rationales, are also complementary to the input. Interestingly, adding rationales to the input reduces MDL more than adding explanations, suggesting that while explanations are useful, they are harder to use for label compression than rationales. For additional analysis on other datasets, see Appendix §A, where we evaluate the usefulness of word order, different types of words, and male vs. female -gendered words across 12 NLP tasks.

4 CONCLUSION

Our work opens up ample opportunity for future work: uncovering which capabilities help on what tasks, detecting dataset biases for data statements (Gebru et al., 2018; Bender & Friedman, 2018), and expanding on RDA, e.g., investigating the underlying data distribution (Whitney et al., 2020). Overall, RDA is a theoretically-justified tool that provides valuable insights on a variety of datasets.

REFERENCES

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*, 2015.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020. URL https://arxiv.org/abs/2004.05150.

Emily M. Bender and Batya Friedman. Data statements for natural language processing: Toward mitigating system bias and enabling better science. *Transactions of the Association for Computational Linguistics*, 6:587–604, 2018. doi: 10.1162/tacl_a_00041. URL https://www.aclweb.org/anthology/Q18-1041.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09*, 2009. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.232.1231.

Léonard Blier and Yann Ollivier. The description length of deep learning models. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31, pp. 2216–2226. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/3b712de48137572f3849aabd5666a4e3-Paper.pdf.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5454–5476, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.485. URL https://www.aclweb.org/anthology/2020.acl-main.485.

Alselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Inf. Process. Lett.*, 24(6):377–380, April 1987. ISSN 0020-0190. doi: 10.1016/0020-0190(87)90114-1. URL https://doi.org/10.1016/0020-0190(87)90114-1.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 4349–4357. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/a486cd07e4ac3d270571622f4f316ec5-Paper.pdf.

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. doi: 10.18653/v1/D15-1075. URL https://www.aclweb.org/anthology/D15-1075.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In Sorelle A. Friedler and Christo Wilson (eds.), *Proceedings of the 1st Conference on Fairness, Accountability and Transparency*, volume 81 of *Proceedings of Machine Learning Research*, pp. 77–91, New York, NY, USA, 23–24 Feb 2018. PMLR. URL http://proceedings.mlr.press/v81/buolamwini18a.html.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. e-snli: Natural language inference with natural language explanations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31, pp. 9539–9549. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/4c7a167bb329bd92580a99ce422d6fa6-Paper.pdf.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo 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, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/S17-2001. URL https://www.aclweb.org/anthology/S17-2001.

Jifan Chen and Greg Durrett. Understanding dataset design choices for multi-hop reasoning. 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)*, pp. 4026–4032, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1405. URL https://www.aclweb.org/anthology/N19-1405.

Kyunghyun Cho, Bart van Merriënboer, 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, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://www.aclweb.org/anthology/D14-1179.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, USA, 2006. ISBN 0471241954. URL http://staff.ustc.edu.cn/~cgong821/Wiley.Interscience.Elements.of.Information.Theory.Jul.2006.eBook-DDU.pdf.

A. P. Dawid. Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *Journal of the Royal Statistical Society. Series A (General)*, 147(2): 278–292, 1984. ISSN 00359238. URL http://www.jstor.org/stable/2981683.

Shrey Desai and Greg Durrett. Calibration of pre-trained transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 295–302, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.21. URL https://www.aclweb.org/anthology/2020.emnlp-main.21.

Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. Queens are powerful too: Mitigating gender bias in dialogue generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8173–8188, Online, November 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.656. URL https://www.aclweb.org/anthology/2020.emnlp-main.656.

Emily Dinan, Angela Fan, Ledell Wu, Jason Weston, Douwe Kiela, and Adina Williams. Multi-dimensional gender bias classification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 314–331, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.23. URL https://www.aclweb.org/anthology/2020.emnlp-main.23.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://www.aclweb.org/anthology/I05-5002.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna M. Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *CoRR*, abs/1803.09010, 2018. URL http://arxiv.org/abs/1803.09010.

Peter Grünwald. A tutorial introduction to the minimum description length principle. *CoRR*, math.ST/0406077, 06 2004. URL https://arxiv.org/abs/math/0406077.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1321–1330. JMLR.org, 2017. URL http://proceedings.mlr.press/v70/guo17a.html.

Ashim Gupta, Giorgi Kvernadze, and Vivek Srikumar. Bert & family eat word salad: Experiments with text understanding, 2021. URL https://arxiv.org/abs/2101.03453.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 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)*, pp. 107–112, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL https://www.aclweb.org/anthology/N18-2017.

Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, COLT '93, pp. 5–13, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916115. doi: 10.1145/168304.168306. URL https://doi.org/10.1145/168304.168306.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017. URL https://github.com/explosion/spaCy.

Yichen Jiang and Mohit Bansal. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2726–2736, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1262. URL https://www.aclweb.org/anthology/P19-1262.

J. Johnson, B. Hariharan, Laurens van der Maaten, Li Fei-Fei, C. L. Zitnick, and Ross B. Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1988–1997, 2017. URL https://arxiv.org/abs/1612.06890.

Anna Jørgensen, Dirk Hovy, and Anders Søgaard. Challenges of studying and processing dialects in social media. In *Proceedings of the Workshop on Noisy User-generated Text*, pp. 9–18, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.18653/v1/W15-4302. URL https://www.aclweb.org/anthology/W15-4302.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pp. 427–431. Association for Computational Linguistics, April 2017. URL https://arxiv.org/abs/1607.01759.

Andrei Nikolaevic Kolmogorov. Three approaches to the quantitative definition of information. *International journal of computer mathematics*, 2(1-4):157–168, 1968. URL https://www.tandfonline.com/doi/abs/10.1080/00207166808803030.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1eA7AEtvS.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, pp. 552–561. AAAI Press, 2012. ISBN 9781577355601. URL https://cs.nyu.edu/faculty/davise/papers/WSKR2012.pdf.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://www.aclweb.org/anthology/2020.acl-main.703.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Charles Lovering, Rohan Jha, Tal Linzen, and Ellie Pavlick. Information-theoretic probing explains reliance on spurious features. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=mNtmhaDkAr.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. Compositional questions do not necessitate multi-hop reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4249–4257, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1416. URL https://www.aclweb.org/anthology/P19-1416.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6097–6109, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1613. URL https://www.aclweb.org/anthology/P19-1613.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4885–4901, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.441. URL https://www.aclweb.org/anthology/2020.acl-main.441.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://www.aclweb.org/anthology/P02-1040.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. URL https://ojs.aaai.org/index.php/AAAI/article/view/11671.

Ethan Perez, Siddharth Karamcheti, Rob Fergus, Jason Weston, Douwe Kiela, and Kyunghyun Cho. Finding generalizable evidence by learning to convince Q&A models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 2402–2411, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1244. URL https://www.aclweb.org/anthology/D19-1244.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 8864–8880, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.713. URL https://www.aclweb.org/anthology/2020.emnlp-main.713.

Thang M. Pham, Trung Bui, Long Mai, and Anh Nguyen. Out of order: How important is the sequential order of words in a sentence in natural language understanding tasks?, 2020. URL https://arxiv.org/abs/2012.15180.

P. J. Phillips, Hyeonjoon Moon, S. A. Rizvi, and P. J. Rauss. The feret evaluation methodology for face-recognition algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22 (10):1090–1104, 2000. doi: 10.1109/34.879790.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pp. 180–191, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/S18-2023. URL https://www.aclweb.org/anthology/S18-2023.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4932–4942, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1487. URL https://www.aclweb.org/anthology/P19-1487.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL https://www.aclweb.org/anthology/D16-1264.

J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465 – 471, 1978. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(78)90005-5. URL http://www.sciencedirect.com/science/article/pii/0005109878900055.

J. Rissanen. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information Theory*, 30(4):629–636, 1984. doi: 10.1109/TIT.1984.1056936.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3): 379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, and Adina Williams. Unnatural language inference, 2020.

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, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.

Saku Sugawara, Pontus Stenetorp, Kentaro Inui, and Akiko Aizawa. Assessing the benchmarking capacity of machine reading comprehension datasets. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8918–8927, Apr. 2020. doi: 10.1609/aaai.v34i05.6422. URL https://ojs.aaai.org/index.php/AAAI/article/view/6422.

Rachael Tatman. Gender and dialect bias in YouTube's automatic captions. In *Proceedings of the First ACL Workshop on Ethics in Natural Language Processing*, pp. 53–59, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1606. URL https://www.aclweb.org/anthology/W17-1606.

Masatoshi Tsuchiya. Performance impact caused by hidden bias of training data for recognizing textual entailment. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May 2018. European Language Resources Association (ELRA). URL https://www.aclweb.org/anthology/L18-1239.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Elena Voita and Ivan Titov. Information-theoretic probing with minimum description length. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 183–196, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.14. URL https://www.aclweb.org/anthology/2020.emnlp-main.14.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJ4km2R5t7.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *arXiv preprint arXiv:1805.12471*, 2018.

William F. Whitney, Min Jae Song, David Brandfonbrener, Jaan Altosaar, and Kyunghyun Cho. Evaluating representations by the complexity of learning low-loss predictors, 2020.

Sarah Wiegreffe, Ana Marasovic, and Noah A. Smith. Measuring association between labels and free-text rationales, 2020.

Adina Williams, Nikita Nangia, and Samuel Bowman. 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)*, pp. 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://www.aclweb.org/anthology/N18-1101.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

Dani Yogatama, Cyprien de Masson d'Autume, Jerome Connor, Tomás Kociský, Mike Chrzanowski, Lingpeng Kong, Angeliki Lazaridou, Wang Ling, Lei Yu, Chris Dyer, and Phil Blunsom. Learning and evaluating general linguistic intelligence. *CoRR*, abs/1901.11373, 2019. URL http://arxiv.org/abs/1901.11373.

Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and Yang: Balancing and answering binary visual questions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016a.

Ye Zhang, Iain Marshall, and Byron C. Wallace. Rationale-augmented convolutional neural networks for text classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 795–804, Austin, Texas, November 2016b. Association for Computational Linguistics. doi: 10.18653/v1/D16-1076. URL https://www.aclweb.org/anthology/D16-1076.

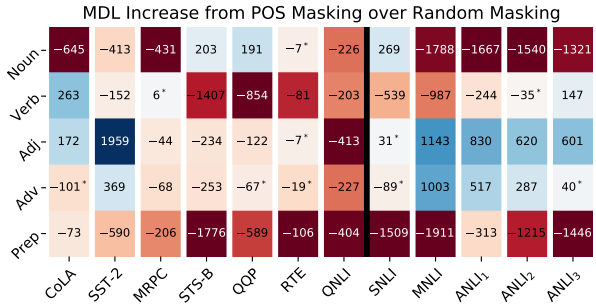

Figure 2: The importance of different POS words, given by $\text{MDL}_{-\text{POS}} - \text{MDL}_{-\text{Random}}$. 0 indicates that words of a given POS are as important as randomly-chosen words, while > 0 and < 0 indicate greater and lesser importance than randomly-chosen words, respectively. (*) indicates mean within std. error of 0 (measured over 5 runs). Color is normalized by column.

A Examining Text Datasets

So far, we used RDA to determine when adding input features helps reduce label description lengths. Similarly, we evaluate when removing certain features increases description length, to determine what features help achieve a small MDL. Here, we view the "original" input as having certain features missing, and we evaluate the utility of a capability f that recovers the missing features to return the normal task input. If f reduces the label-generating program length, then it is useful to have access to f (the ablated features). To illustrate, we examine the utility of different kinds of words and of word order on the General Language Understanding Evaluation benchmark (GLUE; Wang et al., 2019), a central evaluation suite in NLP, as well as SNLI and Adversarial NLI (ANLI; Nie et al., 2020).

Datasets GLUE consists of 9 tasks (8 classification, 1 regression[1]). Two are single-sentence classification; *CoLA* (Corpus of Linguistic Acceptability; Warstadt et al., 2018) involves determining if a sentence is linguistically acceptable or not, while *SST-2* (Stanford Sentiment Treebank 2; Socher et al., 2013) involves predicting if a sentence has positive or negative sentiment. Three tasks involve determining if two sentences are similar or paraphrases of each other: *MRPC* (Microsoft Research Paragraph Corpus; Dolan & Brockett, 2005), *QQP* (Quora Question Pairs)[2], and *STS-B* (Semantic Textual Similarity Benchmark; Cer et al., 2017). The rest are NLI tasks: *QNLI* (Question NLI, derived from SQuAD; Rajpurkar et al., 2016), *RTE* (Recognizing Textual Entailment; Bentivogli et al., 2009), *MNLI* (Multi-genre NLI; Williams et al., 2018), and *WNLI* (Winograd NLI; Levesque et al., 2012); we omit WNLI due to its size, 634 training examples. ANLI consists of NLI data collected in three rounds, where annotators wrote hypotheses that fooled state-of-the-art NLI models trained on data from the previous round. We consider each round as a separate dataset, to examine how NLI datasets have evolved over time, from SNLI to MNLI to ANLI_1, ANLI_2, and ANLI_3.

Experimental Setup We follow a similar setup as for e-SNLI (§3.2), using the 10-model ensemble and evaluating MDL on up to 10k examples per task.

A.1 The usefulness of part-of-speech words

We consider the original input to be the full input with words of a certain POS masked out (with "_") and evaluate the utility of a subroutine that fills in the masked words. To control for the number of words masked, we restrict the subroutine such that it returns a version of the input with exactly the same proportion of words masked, chosen uniformly at random. If the subroutine is useful, then words of a given type are more useful for compression than randomly-chosen input words. In particular, we report the difference between MDL when (1) words of a given POS are masked and (2)

[1] See Appendix §D.4 for details on regression.

[2] data.quora.com/First-Quora-Dataset-Release-Question-Pairs

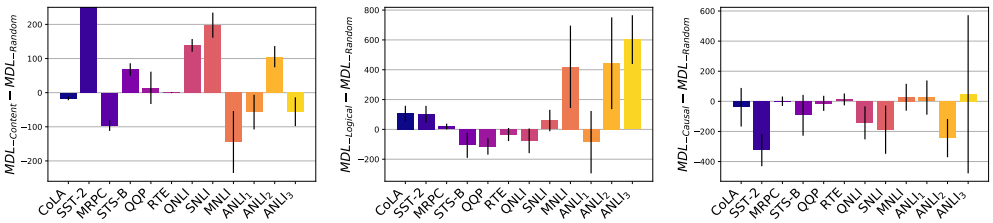

Figure 3: Difference between MDL when we mask input words that are (1) of a given type and (2) randomly chosen with the same frequency as (1). Mean and std. err. over 5 random seeds for content words (**left**), logical words (**middle**), and causal words (**right**).

the same fraction of words are masked uniformly at random: $\text{MDL}_{-\text{POS}} - \text{MDL}_{-\text{Random}}$. We evaluate nouns, verbs, adjectives, adverbs, and prepositions.[3]

We show results in Figure 2. Adjectives are much more useful than other POS for SST-2, a sentiment analysis task where relevant terms are evidently descriptive words (e.g., "the service was *terrible*"). For CoLA, verbs play an important role in determining if a sentence is linguistically acceptable, likely due to the many examples evaluating verb argument structure (e.g., "The toast burned" vs. "The toast buttered."). Other tasks (MRPC, RTE, and QNLI) do not rely significantly on any one POS, suggesting that they require reasoning over multiple POS in tandem. Nouns are consistently less useful on NLI tasks, suggesting that NLI datasets should be supplemented with knowledge-intensive tasks like open-domain QA that rely on names and entities, in order to holistically evaluate language understanding. Prepositions are not important for any GLUE task, suggesting where GLUE can be complemented with other tasks and illustrating how RDA can be used to help form comprehensive benchmarks in the future.

A.2 The usefulness of different word types

How useful are content words? Sugawara et al. (2020) hypothesized that "content" words are particularly useful for NLP tasks, taking content words to be nouns, verbs, adjectives, adverbs, or numbers. We test their utility on GLUE, SNLI, and ANLI using RDA, by evaluating $\text{MDL}_{-\text{Content}} - \text{MDL}_{-\text{Random}}$ (Fig. 3 left). The value is positive for SST-2, STS-B, QNLI, SNLI, and ANLI_2 and negative for MRPC, MNLI, ANLI_1, and ANLI_3. In particular, the value for SST-2 is very high (1732), indicating that content words are important for sentiment classification, likely due to the importance of adjectives as found in §A.1. For QNLI, content words are important, despite earlier findings that each individual POS group (nouns, verbs, adjectives, or adverbs) were not important for QNLI (§A.1 Fig. 2), indicating that QNLI requires reasoning over multiple POS in tandem.

How useful are "logical" words? Sugawara et al. (2020) hypothesized that words that have to do with the logical meaning of a sentence (e.g., quantifiers and logical connectives) are useful for NLP tasks. Using GLUE, SNLI, and ANLI, we test the usefulness of logical words, which we take as: *all, any, each, every, few, if, more, most, no, nor, not, n't, other, same, some,* and *than* (following Sugawara et al., 2020). As shown in Fig. 3 (middle), $\text{MDL}_{-\text{Logical}} - \text{MDL}_{-\text{Random}}$ is positive for CoLA, SST-2, MNLI, ANLI_2, and ANLI_3 and negative for STS-B and QQP. Notably, $\text{MDL}_{-\text{Logical}} - \text{MDL}_{-\text{Random}}$ is large for MNLI, ANLI_2, and ANLI_3, three entailment detection tasks, where we would expect logical words to be important.

How useful are causal words? Another group of words that Sugawara et al. (2020) hypothesized are useful are words that express causal relationships: *as, because, cause, reason, since, therefore,* and *why*. As shown in Fig. 3 (right), $\text{MDL}_{-\text{Causal}} - \text{MDL}_{-\text{Random}}$ to be within std. error of 0 for all tasks except SST-2, QNLI, SNLI, and ANLI_2, where $\text{MDL}_{-\text{Causal}} - \text{MDL}_{-\text{Random}} < 0$. Thus, causal words do not appear particularly useful for GLUE.

[3]We use POS tags from spaCy's large English model (Honnibal & Montani, 2017). For computational reasons, we omit other POS, as they occur less frequently and masking them did not greatly impact MDL in preliminary experiments.

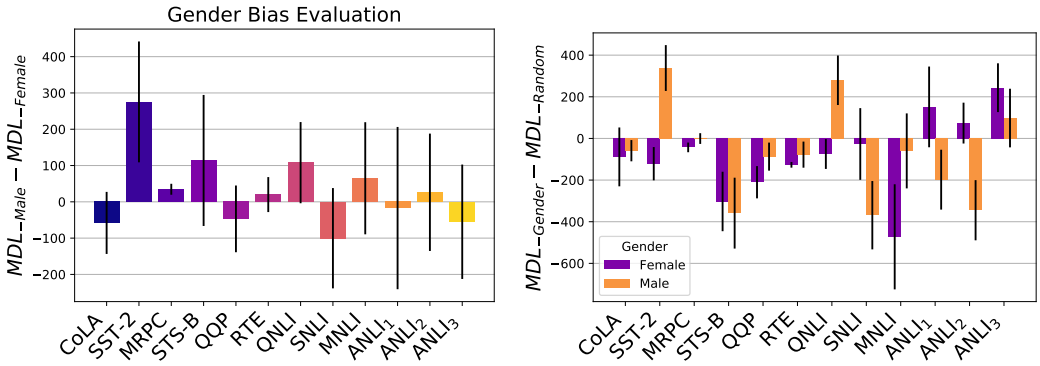

Figure 4: **Gender Bias Evaluation**: *Left*: MDL when masking masculine vs. feminine words (mean and std. err. over 5 random seeds). Values above zero (vs. below zero) indicate that male-gendered words (vs. female-gendered words) are more important for compressing labels. SST-2 shows the largest bias (male-gendered). *Right*: The difference between MDL when (1) masculine/feminine words are masked and (2) the same fraction of input words are masked uniformly at random.

A.3 Do datasets suffer from gender bias?

Gender bias in data is a prevalent issue in machine learning (Bolukbasi et al., 2016; Blodgett et al., 2020). For example, prior work found that machine learning systems are worse at classifying images of women (Phillips et al., 2000; Buolamwini & Gebru, 2018), at speech recognition for women and speakers from Scotland (Tatman, 2017), and at POS tagging for African American vernacular (Jørgensen et al., 2015). RDA can be used to diagnose such biases. Here, we do so by masking male-gendered words and evaluating the utility of an oracle subroutine that reveals male-gendered words while masking female-gendered words. If the subroutine is useful and $\text{MDL}_{-\text{Male}} - \text{MDL}_{-\text{Female}} > 0$, then masculine words are more useful than feminine words for the dataset (gender bias). We use male and female word lists from Dinan et al. (2020a;b). The two lists are similar in size ($\sim$530 words each) and POS distribution (52% nouns, 29% verbs, 18% adjectives), and the male- and female- gendered words occur with similar frequency.

Fig. 4 (left) shows the results. Masculine words are more useful for SST-2 and MRPC while no GLUE datasets have feminine words as more useful. For SST-2, feminine words occur more frequently than masculine words (2.7% vs. 2.2%, evenly distributed across class labels), suggesting that RDA uncovers a gender bias that word counts do not. This result highlights the practical value of RDA in uncovering where evaluation benchmarks under-evaluate the performance of NLP systems on text related to different demographic groups.

As hinted above, we may wish to focus on gender bias present in datasets beyond easy-to-detect differences in male- and female- gendered word frequency. To control for frequency, we evaluate $\text{MDL}_{-\text{Male}} - \text{MDL}_{-\text{Random}}$ and $\text{MDL}_{-\text{Female}} - \text{MDL}_{-\text{Random}}$, as we did for our word type experiments. We show results in Fig. 4 (right). For SST-2 and QNLI, masculine words are more useful than randomly-chosen words, while feminine words are less useful than randomly-chosen words, a sign of gender bias. Most tasks, however, do not show similar patterns of bias as SST-2 and QNLI do.

A.4 How useful is word order?

Recent work claims that state-of-the-art NLP models do not use word order for GLUE tasks (Pham et al., 2020; Sinha et al., 2020; Gupta et al., 2021), so we use RDA to examine the utility of word order on GLUE, by testing the value of a subroutine that unshuffles words when input words have been shuffled.

Fig. 5 (left) shows MDL with and without shuffling, normalized by the MDL of the label-only prior $p(y)$ as a baseline. Word order helps to obtain smaller MDL on all tasks. For example, on MNLI, adding word order enables the labels to be compressed from $75\% \rightarrow 50\%$ of the baseline compression rate. For CoLA, the linguistic acceptability task, input word order is necessary to compress labels

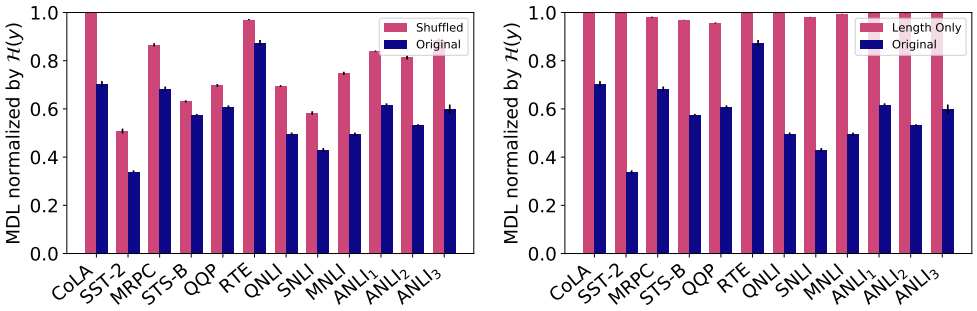

Figure 5: **Left**: *MDL with/without Word Shuffling*, normalized by the MDL when encoding labels with $p(y)$ for reference. Word order reduces MDL on all tasks. **Right**: *MDL with length-only input* compared to MDL with original input, normalized by the MDL when encoding labels with $p(y)$ for reference. Length input reduces MDL over $p(y)$ on MRPC, STS-B, QQP, and SNLI but not greatly.

at all. Prior work may have come to different conclusions about the utility of word order because they evaluate the behavior of trained models on out-of-distribution (word-shuffled) text, while RDA estimates an intrinsic property of the dataset.

A.5 How useful is input length?

Input text length can be highly predictive of the class label. RDA can be used to evaluate text datasets for such length bias. We evaluate MDL when only providing the input length, in terms of number of tokens (counted via spaCy).[4] As shown in Fig. 5 (right), the labels in MRPC, STS-B, QQP, and SNLI can be compressed using the input length, though not to a large extent. Other tasks cannot be compressed using length alone. Our results on SNLI agree with Gururangan et al. (2018) who found that hypotheses were generally shorter for entailment examples and longer for neutral examples. Similarly, they also found that length is less discriminative on MNLI than SNLI.

B Validating Rissanen Data Analysis

In this section, we verify that $\bar{\mathcal{L}}_p(y_{1:N}|x_{1:N}, f) < \bar{\mathcal{L}}_p(y_{1:N}|x_{1:N})$ holds in practice when we use an f that we know is helpful. To this end, we use CLEVR (Johnson et al., 2017), an image-based question-answering (QA) dataset. CLEVR is a synthetic dataset where many questions are carefully designed to benefit from answering subquestions. For example, to answer the CLEVR question "*Are there more cubes than spheres?*" it helps to know the answer to the subquestions "*How many cubes are there?*" and "*How many spheres are there?*" and then compare the resulting answers. We hypothesize that MDL decreases as we give a model answers to subquestions.

We test our hypothesis on three types of question in CLEVR which have 1-2 relevant subquestions. "Integer Comparison" questions ask to compare the numbers of two kinds of objects and have two subquestions, i.e., "Are there more cubes than spheres?" where the two subquestions are "*How many cubes are there?*" and "*How many spheres are there?*" "Attribute Comparison" questions ask to compare the properties of two objects, i.e., "*Is the metal object the same color as the rubber thing?*", where there are two subquestions which each ask about the property of a single object, i.e., "*What color is the metal object?*" and "*What color is the rubber thing?*" "Same Property As" questions ask whether or not one object has the same property as another object, i.e., "*What material is the sphere with the same color as the rubber cylinder?*", where there is one subquestion that asks about a property of one object, i.e., "*What color is the rubber cylinder?*" To obtain oracle subanswers, we use ground-truth programs given by CLEVR that can be executed over a symbolic, graph-based representation of the image to answer each question. For each question category above, we evaluate the subprogram corresponding to its subquestion(s) to generate oracle subanswer(s). We append subanswers to the question (in order), and we evaluate the utility of providing 0-2 subanswers.

[4]Masking all input tokens gave similar results.

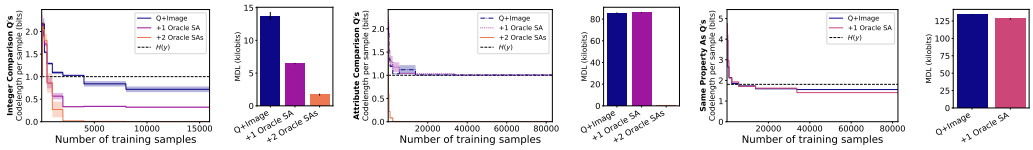

Figure 6: **Left**: Answer codelengths for different CLEVR question types with/without adding oracle answers to subquestions ("subanswers") to the input. **Right**: Subanswers reduce MDL, here the total required codelength to learn a near-perfect question-answering model.

Model We use the FiLM model from Perez et al. (2018) which combines a convolutional network for the image with a GRU for the question (Cho et al., 2014). The model minimizes cross-entropy loss (27-way classification). We follow training strategy from Perez et al. (2018) using the public code, except we train for at most 20 epochs (not 80), since we only train on subsets of CLEVR.

Results Fig. 6 shows codelengths and MDL. For all question types, $\bar{\mathcal{L}}_p(y_{1:N}|x_{1:N}, f) < \bar{\mathcal{L}}_p(y_{1:N}|x_{1:N})$ when all oracle subanswers are given, as expected. For "Integer Comparison" (top) and "Attribute Comparison" (middle), the reduction in MDL is larger than for "Same Property As" questions (Fig. 6 bottom). For comparison question types, the subanswers alone can be used to determining the answer, explaining the larger decreases in MDL. Our results align with our expectations about when answers to subquestions are helpful, empirically validating RDA.

C Experimental Setup

To evaluate MDL, we first randomly sort examples in the dataset. We use $S = 9$ blocks where $t_0 = 0$ and $t_1 = 64 < \cdots < t_S = N$ such that $\frac{t_{s+1}}{t_s}$ is constant (log-uniform spacing). To train a model on the first s blocks, we randomly split the available examples into train (90%) and dev (10%) sets, choosing hyperparameters and early stopping epoch using dev loss (codelength). We otherwise follow each model's training strategy and hyperparameter ranges as suggested by its original paper. We then evaluate the codelength of the $(s + 1)$-th block.

Various random factors impact MDL, such as the order of examples, model initialization, and randomness during training. Thus, we report the mean and std. error of MDL over 5 random seeds. For computational efficiency, we only sweep over hyperparameters for the first random seed and reuse the best hyperparameters for the remaining seeds.

D Model Details

D.1 Question Decomposition Methods

Three of the question decomposition we test are unsupervised methods from Perez et al. (2020): pseudo-decomposition (retrieval-based subquestions), seq2seq (subquestions from a sequence-to-sequence model), and ONUS (One-to-N Unsupervised Sequence transduction). Last, we test the ability of a more recent, large language model (GPT3; Brown et al., 2020) to generate subquestions using a few labeled question-decomposition examples. Since generating with GPT3 is expensive, we use its generated subquestions as training data for a smaller T5 model (Raffel et al., 2020), a "Distilled Language Model" (explained below).

D.2 Distilled Language Model Decompositions

D.2.1 Language Model Decompositions

Large language models are highly effective at text generation (Brown et al., 2020) but have not yet been explored in the context of question decomposition. In particular, one obstacle is the sheer computational and monetary cost associated with such models. We thus use a language model to generate question decompositions while conditioning on a few labeled question-decomposition pairs,

and then we train a smaller, sequence-to-sequence model on the generated question-decomposition pairs, which we use to efficiently decompose many questions. Our approach, which we call Distilled Language Model (DLM), leverages the large language model to produce pseudo- training data for a more efficient model.

As our language model, we use the 175B parameter, pretrained GPT-3 model (Brown et al., 2020) via the OpenAI API.[5] We label the maximum number of question-decomposition pairs that fit in the context window of GPT-3 (2048 tokens or 46 question-decompositions). For labeling, we sample questions randomly from HOTPOTQA's training set. To condition the language model, we format question-decomposition pairs as "[Question] = [Decomposition]", where the decomposition consists of several consecutive sub-questions. We concatenate the pairs, each on a new line, with a new question on the final line to form a prompt. We then generate from the LM, conditioned on the prompt. For decoding, we found that GPT-3 copies the question as the decomposition with greedy decoding. Therefore, we use a sample-and-rank decoding strategy, to choose the best decoding out of several possible candidates. We sample 16 decompositions with top-p sampling (Holtzman et al., 2020) with $p = 0.95$, rank decompositions from highest to lowest based their average token-level log probability, and choose the highest-ranked decomposition which satisfies the basic sanity checks for decomposition from Perez et al. (2020). The sanity checks avoid the question-copying failure mode by checking if a decomposition has (1) more than one sub-question (question mark), (2) no sub-question which contains all words in the multi-hop question, and (3) no sub-question longer than the multi-hop question. We generate decompositions for HOTPOTQA dev questions, which we estimate costs $\$0.15$ per example or $\$1.1$k for the 7405 dev examples via the OpenAI API. Decomposing all 90447 training examples would roughly cost an extra $\$13.3$k, motivating distillation.

D.2.2 DISTILLING DECOMPOSITIONS

As our distilled, sequence-to-sequence model, we use the 3B parameter, pretrained T5 model (Raffel et al., 2020) via HuggingFace Transformers (Wolf et al., 2020). We finetune T5 on our question-decomposition examples and then use it to generate subquestions for all training questions.

To finetune T5, we split our question-decomposition examples into train (80%), dev (10%), and test (10%) splits. We finetune T5 with a learning rate of $1e-4$, and we sweep over label smoothing $\in \{0.1, 0.2, 0.4, 0.6\}$, number of training epochs $\in \{3, 5, 10\}$, and batch size in $\in \{16, 32, 64\}$, choosing the best hyperparmeters $(0.1, 3, 64$, respectively) based on dev BLEU (Papineni et al., 2002). We stop training early when dev BLEU does not increase after one training epoch. We generate decompositions using beam search of size 4 and length penalty of 0.6 as in Raffel et al. (2020), achieving a test BLEU of 50.7. We then finetune a new T5 model using the best hyperparameters on all question-decomposition examples except for a small set of 200 examples used for early stopping.

D.3 LONGFORMER

We train the model to predict the span's start token and end token by minimizing the negative log-likelihood for each prediction. We treat yes/no questions as span prediction as well by prepending "yes" and "no" to the input, following Perez et al. (2020). We use the implementation from Wolf et al. (2020). Similar to Beltagy et al. (2020), we train LONGFORMER models for up to 6 epochs, stopping training early if dev loss doesn't decrease after one epoch. We sweep over learning rate $\in \{3 \times 10^{-5}, 5 \times 10^{-5}, 1 \times 10^{-4}\}$.

D.4 REGRESSION

STS-B is a regression task in GLUE where labels are continuous values in $[0, 5]$. Here, we learn to minimize mean-squared error, which is equivalent to minimizing log-likelihood and thus codelength.[6] We treat each scalar prediction as the mean of a Gaussian distribution and tune a single standard deviation parameter shared across all predictions from a single model. We choose the variance based on dev log-likelihood using grid search over $[10^{-2.5}, 10^{1.5}]$ with 1000 log-uniformly spaced samples. To send the first block of labels, Alice and Bob use a uniform distribution over the interval $[0, 5]$.

[5] https://beta.openai.com/

[6] Cover & Thomas (2006) justifies the relationship between log-likelihood and codelength for continuous random variables.

Hyperparam	LONGFORMER	ROBERTA	BART	ALBERT	GPT2
Learning Rate	{3e-5, 5e-5, 1e-4}	{1e-5, 2e-5, 3e-5}	{5e-6, 1e-5, 2e-5}	{2e-5, 3e-5, 5e-5}	{6.25e-5, 3.125e-5, 1.25e-4}
Batch Size	32	{16, 32}	{32, 128}	{32, 128}	32
Max Epochs	6	10	10	3	3
Weight Decay	0.01	0.1	0.01	0.01	0.01
Warmup Ratio	0.06	0.06	0.06	0.1	0.002
Adam β_2	0.999	0.98	0.98	0.999	0.999
Adam ϵ	1e-6	1e-6	1e-8	1e-6	1e-8
Grad. Clip Norm	∞	∞	∞	1	1

Table 1: Training hyperparameters for all transformer models, based on those from each model's original paper. Column names refer to model types, including models of different sizes or trained from scratch with the same architecture.

The FastText library only supports classification, so we turn STS-B into a 26-way classification task by rounding label values to the nearest 0.2, following T5 (Raffel et al., 2020). We compute a real-valued, mean prediction by evaluating the average class label value when marginalizing over class probabilities. We then tune variance on dev as usual.

D.5 ENSEMBLE MODEL

To limit the effect of the choice of learning algorithm $\mathcal{A}$, we may ensemble many model classes. To do so, we have Alice train M models of different classes and send the next block's labels using the model that gives the shortest codelength. To tell Bob which model to use to decompress a block's labels, Alice also sends $\log_2 M$ bits per block $s = 1, \ldots, S-1$, adding $(S-1)\ \log_2 M$ to MDL. In this way, MDL relies less on the behavior of a single model class.

For experiments in §3.2 (e-SNLI) and Appendix A, we use a 10-model ensemble, with the following model classes: FastText Bag-of-Words (Joulin et al., 2017), transformers (Vaswani et al., 2017) trained from scratch (110M and 340M parameter versions), $\text{BART}_{\text{BASE}}$ (encoder-decoder; Lewis et al., 2020), $\text{ALBERT}_{\text{BASE}}$ (encoder-only; Lan et al., 2020), $\text{ROBERTA}_{\text{BASE}}$ and $\text{ROBERTA}_{\text{LARGE}}$ (encoder-only; Liu et al., 2019) and the distilled version DISTILROBERTA (Sanh et al., 2019), and GPT2 (decoder-only; Radford et al., 2019) and DISTILGPT2 (Sanh et al., 2019). For each model, we minimize cross-entropy loss and tune softmax temperature[7] on dev to alleviate overconfidence on unseen examples (Guo et al., 2017; Desai & Durrett, 2020). We follow each models' official training strategy and hyperparameter sweeps (details below), using the FastText codebase[8] and HuggingFace Transformers (Wolf et al., 2020) for other models.

D.5.1 FASTTEXT

For the FastText classifier, we initialize with the 2M pretrained, 300-dimensional word vectors trained on Common Crawl (600B tokens).[9] We tune hyperparameters using the official implementation of automatic hyperparameter tuning, which we run for 6 hours, which is generally sufficient for 20+ hyperparmeter trials and convergence on dev accuracy. The tuning implementation chooses the hyperparameters based on dev accuracy instead of loss as we typically do, but our procedure of tuning a softmax temperature parameter helps FastText reach significantly below-baseline loss.

D.5.2 TRANSFORMER MODELS

Other models in our ensemble are transformer-based models trained with HuggingFace Transformers (Wolf et al., 2020). Table 1 shows hyperparameter ranges used for each model, chosen based on those in each model's original paper for GLUE. To train the TRANSFORMER from scratch, we use the $\text{ROBERTA}_{\text{BASE}}$ and $\text{ROBERTA}_{\text{LARGE}}$ architecture with ROBERTA hyperparameters but with larger batch sizes ($\in \{64, 128\}$) for the LARGE transformer (for better results).

[7] Search over $[10^{-1}, 10^{2}]$, 1000 log-uniformly spaced samples.

[8] https://github.com/facebookresearch/fastText

[9] https://fasttext.cc/docs/en/english-vectors.html