GENERALIZED OPTIMAL LINEAR ORDERS

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science

by

Rishi Bommasani

August 2020

ABSTRACT

The sequential structure of language, and the order of words in a sentence specifically, plays a central role in human language processing. Consequently, in designing computational models of language, the *de facto* approach is to present sentences to machines with the words ordered in the same order as in the original human-authored sentence. The very essence of this work is to question the implicit assumption that this is desirable and inject theoretical soundness into the consideration of word order in natural language processing. In this thesis, we begin by uniting the disparate treatments of word order in cognitive science, psycholinguistics, computational linguistics, and natural language processing under a flexible algorithmic framework. We proceed to use this heterogeneous theoretical foundation as the basis for exploring new word orders with an undercurrent of psycholinguistic optimality. In particular, we focus on notions of dependency length minimization given the difficulties in human and computational language processing in handling long-distance dependencies. We then discuss algorithms for finding optimal word orders efficiently in spite of the combinatorial space of possibilities. We conclude by addressing the implications of these word orders on human language and their downstream impacts when integrated in computational models.

BIOGRAPHICAL SKETCH

Rishi Bommasani was born in Red Bank, New Jersey and raised in Marlboro, New Jersey. Rishi received his B.A. from Cornell University with degrees in Computer Science and in Mathematics. He graduated *magna cum laude* with distinction in all subjects. He continued studying at Cornell University to pursue a M.S. in Computer Science and was advised by Professor Claire Cardie. During his time as an undergraduate and M.S. student, Rishi received several awards including the Computer Science Prize for Academic Excellence and Leadership and multiple Outstanding Teaching Assistant Awards. He has been fortunate to have completed two internships at Mozilla in the Emerging Technologies team under the advisement of Dr. Kelly Davis in the DeepSpeech group. In his first summer at Mozilla, his work considered genuinely abstractive summarization systems; in his second summer, his research centered on interpreting pretrained contextualized representations via reductions to static embeddings as well as social biases encoded within these representations. He has been a strong advocate for advancing undergraduate research opportunities in computer science and was the primary organizer of numerous undergraduate reading groups, the first Cornell Days Event for Computer Science, and the inaugural Cornell Computer Science Research Night as well as its subsequent iterations. Rishi has been graciously supported by a NAACL Student Volunteer Award, ACL Student Scholarship, Mozilla Travel Grant, and NeurIPS Student Travel Grant. He will begin his PhD at Stanford University in the Computer Science Department and the Natural Language Processing group in Fall 2020. His PhD studies will be funded by a NSF Graduate Research Fellowship.

To my adviser, Claire, for your unrelenting support and unwavering confidence.

You will forever be my inspiration as a researcher and computer scientist.

In loving memory of Marseille.

ACKNOWLEDGEMENTS

There are many people to whom I am grateful and without whom the thesis would have been almost impossible to write (much less finish):[1]

My adviser, Claire Cardie, has shaped who I am as a researcher and computer scientist with seemingly effortless grace. There is truly no way for me to compress my gratitude for her into a few words here. In part, I must thank her for putting up with my constant flow of ideas and for having the patience to allow me to learn from my own errant ideas and mistakes. She has truly adapted herself to accommodate my research interests and it is exactly that freedom that permitted me to develop this very thesis. She occasionally (quite kindly) remarked that she "will be lost" when I leave but it is really me who will be lost without her. She has set an unimaginably high standard for both my PhD adviser(s) and myself to match in the future.

I am also thankful for Bobby Kleinberg for the many hats he has worn (one of which was as my minor adviser). While there are countless encounters and small nuances I have learned from him well beyond algorithms, I think I will always hope to match his relentless curiosity and desire to learn. And I would like to thank Marty van Schijndel as his arrival at Cornell NLP has drastically changed how I view language, psycholinguistics, computational linguistics, and NLP. There has yet to be a dull moment in any of our interactions.

I am deeply fortunate to have learned from and been guided by three rising stars — Vlad Niculae, Arzoo Katiyar, and Xanda Schofield. As three remarkable

[1] These words are also the first words of Claire's thesis.

young professors, I am quite lucky to have been one of their first students. What they might not know is that their theses were also quite inspiring in writing my own; thanks for that as well. In similar spirit, Kelly Davis has been a fabulous adviser during my two summers at Mozilla and I truly appreciate his willingness to let me explore and work on problems that I proposed. Thanks to Steven Wu for being a patient and insightful collaborator as well.

Cornell NLP has been my home for the past few years and I must thank Lillian Lee for the role she played many years prior to my arriving in helping build this exceptional group with Claire. Many of her papers from the late 90's and early 2000's are my exact inspiration for writing well-executed research papers; her characteristic and constant insightfulness is simply sublime. I must also especially note Yoav Artzi, who as a researcher and a friend has deeply inspired my work and my commitment to being disciplined and principled. Cristian Danescu-Niculescu-Mizil, Sasha Rush, David Mimno, and Mats Rooth have been great members at the weekly NLP seminar and have further broadened the set of diverse perspectives towards NLP that I was privy to, further enriching me as a young scholar. More recently, the `C.Psyd` group has become an exciting community for me to properly face the complexities of language and the intriguing perspectives afforded by psycholinguistics.

At a broader scale, Cornell CS has been truly formative in how I view the world. I hope I will do Bob Constable proud in viewing the world computationally. I am very grateful to Kavita Bala for her untiring efforts to make the department a positive community that supports learning. And I am thankful to Joe Halpern and Eva Tardos for being excellent all-encompassing role models of what it means to be

a great computer scientist and great faculty member. Similarly, Anne Bracy and Daisy Fan have been especially superlative for me in exemplifying great teaching. Adrian Sampson, Lorenzo Alvisi, Eshan Chattopadhyay, and countless others have all shown me the warm and collegial spirit that radiates throughout our department. I hope to carry this forward to the next places along my journey. Too often underappreciated, Vanessa Maley, Becky Stewart, Nicole Roy, Ryan Marchenese, and Randy Hess were all great resources that made my journey as an undergrad and masters student that much easier.

Ravi Ramakrishna is the person who showed me how exciting research can truly be and reignited my passion for mathematics. He might not know it, but counterfactually without him and the environment he fostered in MATH 2230, it is hard to imagine me being where I am now.

But that is enough with acknowledging faculty and old folks. I have been very fortunate to have a great number of research friends in Cornell NLP and across Cornell who have mentored me and been great to learn alongside:
Esin Durmus, Ge Gao, Forrest Davis, Tianze Shi, Maria Antoniak, Jack Hessel, Xilun Chen, Xinya Du, Kai Sun, Ryan Benmalek, Laure Thompson, Max Grusky, Alane Suhr, Ana Smith, Justine Zhang, Greg Yauney, Liye Fu, Jonathan Chang, Nori Kojima, Andrea Hummel, Jacob Collard, Matt Milano, Malcolm Bare.

Further, I have had many wonderful friends who have encouraged me, especially Linus Setiabrata, Janice Chan[2], Avani Bhargava, Isay Katsman, Tianyi Zhang, Cosmo Viola, and Will Gao. I have also cherished my time with:

[2]I am also grateful to Janice for proofreading parts of this thesis.

Andy Zhang, Eric Feng, Jill Wu, Jerry Qu, Haram Kim, Kevin Luo, Dan Glus, Sam Ringel, Maria Sam, Zach Brody, Tjaden Hess, Horace He, Kabir Kapoor, Yizhou Yu, Rachel Shim, Nancy Sun, Jacob Markin, Harry Goldstein, Chris Colen, Ayush Mittal, Cynthia Rishi, Devin Lehmacher, Brett Clancy, Daniel Nosrati, Victoria Schneller, Jimmy Briggs, Irene Yoon, Abrahm Magaña, Danny Qiu, Katie Borg, Katie Gioioso, Swathi Iyer, Florian Hartmann, Dave Connelly, Sasha Badov, Sourabh Chakraborty, Daniel Galaragga, Qian Huang, Judy Huang, Keely Wan, Amrit Amar, Daniel Weber, Ji Hun Kim, Victor Butoi, Priya Srikumar, Caleb Koch, Shantanu Gore, Grant Storey, Jialu Li, Frank Li, Seraphina Lee.

Throughout my time at Cornell CS, two sources of persistent inspiration were Rediet Abebe and Jehron Petty. It was truly remarkable to witness the change they drove in the department while I was there.

I am grateful to all of the students who I have TA-d for for helping me grow as a teacher. Of special note are the students of CS 4740 in Fall 2019 when I co-taught the course with Claire; I appreciated their patience in tolerating my first attempts to prepare lectures for a course.[3] Similarly, I have been extremely privileged to have worked with and advised a number of exceptional undergraduates and masters students. I hope that I have helped them grow as researchers and better appreciate the exciting challenges involved in pursuing NLP/computational linguistics research:

Aga Koc, Albert Tsao, Anna Huang, Anusha Nambiar, Joseph Kihang'a, Julie Phan, Quintessa Qiao, Sabhya Chhabria, Wenyi Guo, Ye Jiang.

[3] My intent is for this thesis to be understandable to any student who has completed CS 4740.

As I prepare for the next step in my academic journey as a PhD student in the Stanford Computer Science Department and Stanford NLP group, I would like to thank a number of faculty, current (or recently graduated) PhD students, and members of my graduate cohort who helped me during the decision process:
Faculty: Percy Liang[4], Dan Klein, Tatsu Hashimoto, Dan Jurafsky, Chris Potts, Chris Manning, John Duchi, Jacob Steinhardt, Noah Smith, Yejin Choi, Jason Eisner, Ben van Durme, Tal Linzen, Graham Neubig, Emma Strubell, Zach Lipton, Danqi Chen[5], Karthik Narasimhan.
PhD students during the process: Nelson Liu, John Hewitt, Pang Wei Koh, Urvashi Khandelwal, Aditi Raghunathan, Robin Jia, Shiori Sagawa, Kawin Ethayarajh, Eva Portelance, Sidd Karamcheti, Nick Tomlin, Eric Wallace, Cathy Chen, Sabrina Mielke, Adam Poliak, Tim Vieira, Ryan Cotterell, Ofir Press, Sofia Serrano, Victor Zhong, Julian Michael, Divyansh Kaushik.
2020 PhD admits: Alisa Liu, Han Guo, Suchin Gururangan, Katherine Lee, Lisa Li, Xikun Zhang, Aditya Gupta, Victor Sanh, Mengzhou Xia, Megha Srivastava.

A special thank you is also due to those who helped organize virtual visits in light of the unique challenges posed by the COVID-19 pandemic that spring.

Conference travel to present my research was funded by a NAACL Student Volunteer Award, ACL Student Scholarship, Mozilla Travel Grant, and NeurIPS Student Travel Grant in addition to funding from Cornell University and Claire.

[4]Percy's own masters thesis at MIT was quite influential in writing/formatting this thesis.
[5]This section was inspired by Danqi's own dissertation.

The final thank you must go to my parents, Ram and Saila Bommasani, for their patience to allow me to explore what made me happy and their enduring encouragement in allowing me to forge my own path. Few parents understand these subtleties of parenting better than you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

In this chapter, we set forth the motivations and contributions of this work.

1.1 Motivation

Natural language plays a critical role in the arsenal of mechanisms that humans use to communicate. Inherently, natural language is a rich code with fascinating linguistic structure that humans rely upon to transfer information and achieve communicative goals (Shannon, 1948; Miller, 1951; Chomsky, 1956, 1965; Hockett, 1960; Greenberg, 1963; Chomsky, 1986; Pinker and Bloom, 1990; Hawkins et al., 1994; Pinker, 2003; Pinker and Jackendoff, 2005; Pinker, 2007; Jaeger and Tily, 2011; Chomsky, 2014a,b; Gibson et al., 2019). In spite of the fact that natural language is fundamentally a mechanism for human-human discourse, in recent years we have witnessed the emergence of potent computational models of natural language. In particular, society as a whole has come to rely on a variety of language technologies. Prominent examples include machine translation (Weaver, 1949/55; Shannon and Weaver, 1963; Lopez, 2008; Koehn, 2010; Wu et al., 2016), speech recognition and synthesis (Dudley, 1939; Dudley et al., 1939; Yu and Deng, 2014; Chiu et al., 2018; Wang et al., 2017), information retrieval and search (Luhn, 1957; Salton, 1968, 1971; Spärck Jones, 1972; Salton, 1975; Salton et al., 1975; Salton and McGill, 1986; Salton, 1991; Page et al., 1999; Singhal, 2005; Manning et al., 2008; Dean, 2009; Nayak, 2019),

large-scale information extraction (Andersen et al., 1992; Chinchor et al., 1993; Grishman and Sundheim, 1996; Cardie, 1997; Califf and Mooney, 1997; Wilks, 1997; Gaizauskas and Wilks, 1998; Etzioni et al., 2004; Choi et al., 2005; Etzioni et al., 2008; Mintz et al., 2009; Etzioni et al., 2011; Navigli and Ponzetto, 2012; Piskorski and Yangarber, 2013), and sentiment analysis (Pang et al., 2002; Turney, 2002; Pang and Lee, 2004, 2005; Godbole et al., 2007; Pang and Lee, 2008; Bautin et al., 2008; Ye et al., 2009; Asur and Huberman, 2010; Liu, 2012; Chau and Xu, 2012; Li et al., 2014; Ravi and Ravi, 2015; Xing et al., 2017). And the scope for language technologies is only projected to grow even larger in the coming years (Hirschberg and Manning, 2015).

In designing computational models of language, a natural consideration is specifying the appropriate algorithmic primitives. Classical approaches to algorithm design have been considered but generally have struggled to model language faithfully; the exacting nature of deterministic algorithms like quick-sort is ill-suited to the myriad ambiguities found within natural language. Based on empirical findings, the field of natural language processing (NLP) has drifted towards machine learning and probabilistic methods (Charniak, 1994; Manning and Schütze, 1999; Jurafsky and Martin, 2000; Steedman, 2008; Hirschberg and Manning, 2015; Goldberg and Hirst, 2017; Eisenstein, 2019; McClelland et al., 2019) despite the initial dismissal of such statistical approaches by Chomsky (1956). However, this transition alone does not reconcile that the mathematical primitives used in machine learning and deep learning (Mitchell, 1997; Bishop, 2006; Goodfellow et al., 2016), i.e. vectors, affine transformations, and nonlinearities, are inconsistent with

those present in natural language, i.e. characters, words, sentences. One of the characteristic successes of NLP in the past decade has been the development of word embeddings (Bengio et al., 2003; Collobert and Weston, 2008; Collobert et al., 2011; Mikolov et al., 2013; Pennington et al., 2014; Faruqui, 2016): explicit methods for encoding words as vectors where the abstract semantic similarity between words is codified as concrete geometric similarity between vectors. In general, a hallmark of modern NLP is the inherent tension between linguistic representations and computational representations as, simply put, words are not numbers.

In this thesis, we study computational representations of a fundamental aspect of language: word order. Within natural language, sentences are a standard unit of analysis[1] and every sentence is itself a sequence of words. The central question that we consider in this thesis is whether the order of words in a sentence, ascribed by the human who produced it, is the appropriate order for computational models that attempt to comprehend the sentence (in order to perform some downstream task). In order to make principled progress towards answering this question, we contextualize our work against the backdrop of considerations of word order/linear order in the literature bodies of psycholinguistics and algorithms. From a psycholinguistic standpoint, the word order already attested by natural language sentences can be argued to be indicative of an optimization to facilitate human processing. Simultaneously, from an algorithmic perspective, word orders observed in natural

[1]The annual CUNY conference, now in its 34^{th} iteration, is entirely dedicated to the topic of sentence processing.

language may be computationally suboptimal with respect to certain combinatorial objectives, which naturally begs the question of how (computational) processing may change when presented with optimal word orders. In this sense, the unifying approach we adopt in this thesis is to interlace motivating prior work from both psycholinguistics and algorithms to specify novel word orders, which we then evaluate empirically for downstream NLP tasks.

1.2 Contributions

Generalized Optimal Linear Orders. The central contribution of this work is a framework for constructing novel word orders via an optimization procedure and, thereafter, studying the impacts of these orders on downstream NLP tasks. Consequently, we begin by extending and connecting previously disconnected literature from the algorithms community with work that focuses on modelling word order in NLP. We also present three novel word orders generated via the `Transposition Monte Carlo` algorithm that we introduce. These orders rely on a simple greedy heuristic that allows for (somewhat-transparent) balancing of the original sentence's word order, therefore preserving information encoded in the original word order, and optimization against the objectives we introduce. We demonstrate how to incorporate these novel word orders, which are optimal (with respect to a combinatorial objective), with downstream NLP. In particular, we propose the `pretrain-permute-finetune` framework, which seamlessly inte-

grates our novel orders with large-scale pretraining. We empirically evaluate the benefits of our method and show it can yield improvements for English language text classification tasks.

Quantified (sub)optimality of natural language. Due to the explicit computational framework we develop, we can further quantify the extent to which various natural languages are suboptimal with respect to objectives related to dependency length minimization. As we discuss subsequently, there has been significant work in the psycholinguistics community towards demonstrating that human languages are effective at dependency minimizing (compared to random word orders) and our work helps provide the dual perspective by clarifying the extent to which they are suboptimal.

Survey of word order in language processing. Research in human language processing, and sentence processing specifically, has a rich history of studying the influence of word order on processing capabilities in humans. While the corresponding study in natural language processing has arguably lacked similar rigor, this thesis puts forth a joint summary of how multiple communities have studied word order in language processing.

1.3 Organizational Outline

The remainder of this thesis is organized as follows.

We begin in Chapter 2 (§2) by introducing fundamental preliminaries. These include a self-contained introduction to dependency grammars as well as a discussion of the disparate treatments of word order within NLP. We then examine some of the literature on studying word order in human languages in Chapter 3 (§3), with a specific focus on cognitive and psycholinguistic arguments centered on human language processing. We pay special attention to the line of work focused on dependency length minimization and dependency locality effects (§3.4).

In Chapter 4 (§4), we shift gears by providing a generalized framework for studying notions of optimality with respect to dependency length and word order. We further provide several algorithms introduced in prior work that permit tractable (polynomial-time) optimization of various combinatorial objectives. We augment these with heuristic algorithms that allow for balance between retaining the original sentence's order and purely optimizing objectives related to dependency parses.

In Chapter 5 (§5), we consider how the novel word orders we have constructed influence dependency-related costs and downstream performance in NLP. We find that English already substantially optimizes for the objectives we study compared

to a random word order baseline. Further, we show that there is still a substantial margin for further optimization over English and that the heuristic algorithms we introduce perform slightly worse than algorithms that are established in the literature from an optimization perspective. Intriguingly, we find that optimizing for some objectives (most notably MINIMUM LINEAR ARRANGEMENT) can yield to improvements on other objectives but does not in all cases (especially for the BANDWIDTH objective). Given these observations, we then evaluate on downstream text classification tasks. We find that the standard English order is a strong baseline but can be improved over in four of the five datasets we study (by using a novel word order introduced in this work). In particular, we show that word orders generated by our heuristic approach often outperform those generated by standards algorithms, suggesting that word order design that strictly optimizes combinatorial objectives is arguably naive and may not be sufficient/desirable for modelling natural language.

We conclude this thesis by providing a contextualized summary of the results in Chapter 6 (§6). We further provide a discussion of open problems, future directions, and broader lessons. We complement this with a transparent reporting of the inherent limitations of this work.

In Appendix A, we provide an exhaustive set of details to fully reproduce this work. This includes references to code we used to conduct all experiments and generate all tables/figures used in this work. We further provide details for

accessing all datasets used in the work. In Appendix B, we provide additional results that we did not include in the main thesis. These results help clarify the performance of models for suboptimal hyperparameter settings (and, implicitly, the stability of the results to various hyperparameter settings).

1.4 Previous Works

The underlying foundation for this work was originally published in Bommasani (2019), which was presented at *ACL 2019* during the main conference in the *Student Research Workshop*. It was further presented to a machine learning audience at *NeurIPS 2019* in the *Context and Compositionality in Biological and Artificial Neural Systems Workshop*. In both past works, part of the content that appears in §4 and §5 was introduced. The remainder of the thesis was specifically created for the purpose of this thesis.

CHAPTER 2

BACKGROUND

In this chapter we introduce preliminary machinery that we will use throughout the thesis — the dependency parse — and the existing treatments of word order in NLP.

2.1 Primitives

In this thesis, we will denote a sentence by $\bar{s}$ which is alternatively denoted by a sequence of words $\langle w_1 \dots w_n \rangle$. For simplicity of prose, we will assume sentences contain no duplicates though none of the algorithms or results we present make use of this assumption. Given a sentence, the task of decomposing it into its corresponding sequence of words is known as *tokenization*.[1] In practice, while tokenization technically describes breaking "natural language text [...] into distinct meaningful units (or tokens)" (Kaplan, 2005), it is often conflated with various text/string normalization processes (e.g. lowercasing).

In 'separating' languages, such as English, the use of whitespace can be taken as a reasonably proxy for token boundaries whereas in other languages, such as

[1] In this thesis, we will make no distinction between the terms *word* and *token*. Similarly, we will not distinguish *word types* (lexical categories) from *word tokens* (individual occurrences of word types).

Mandarin Chinese, this is not feasible. In general, in this work we will assume access to a tokenizer for the language being studied and will not reconsider any errors introduced during tokenization. In particular, while tokenization is not strictly solved (Dridan and Oepen, 2012), high-quality tokenizers exist for a variety of languages, including some low-resource languages, in standard packages such as Stanford CoreNLP (Manning et al., 2014) and Stanza (Qi et al., 2020).

2.2 Dependency Grammars

In this work, we consider *syntactic* representations of language. Specifically, we focus our efforts on *dependency grammars*, which were first formalized in the modern sense by Lucien Tesnière (Tesnière, 1959).[2] Under a dependency grammar, every sentence has a corresponding *dependency parse* which encodes binary relations between words that mark syntactic dependencies. This approach for specifying a sentence-level syntactic structure differs greatly from the phrase-based/constituency grammars championed by Leonard Bloomfield and Noam Chomsky (Bloomfield, 1933; Chomsky, 1965). The central difference rests on how clauses are handled: phrase-structure grammars split clauses into subject noun phrases and predicate verb phrases whereas dependency grammars are verb-focused. Further, phrase-structure grammar may generate nodes that do not correspond to any single word in the sentence.

[2] Nivre (2005) provides a more comprehensive primer on dependency grammars and dependency parsing.

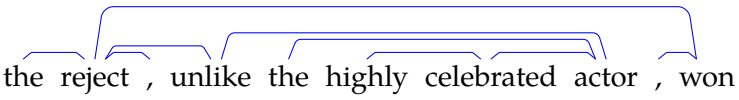

Figure 2.1: Dependency parse of the given sentence. Dependency arcs are drawn canonically (above the linear sequence of words) and the sequence has been lowercased and dependency parsed using the `spaCy` parser (Honnibal and Montani, 2017) for English.

Formally, a dependency grammar attributes a dependency parse $\mathcal{G}_{\bar{s}}$ to every sentence $\bar{s} = \langle w_1 \dots w_n \rangle$ where $\mathcal{G}_{\bar{s}}$ is a directed graph with vertex set $\mathcal{V} = \{w_i \mid i \in [n]\}$ and edge set $\mathcal{E}_\ell$ given by the directed binary dependency relations. Each dependency relation is labelled (hence a dependency parse is an edge-labelled directed graph) and the specific labels are based on the specific *dependency formalism* used, which we describe subsequently. The direction of the edges is from the syntactic head (the source of the edge) to the syntactic child (the target of the edge); the head is of greater syntactic prominence/salience than the child. Dependency parses are constrained to be trees and, since the main verb plays a central role, are often conceived as rooted trees that are rooted at the main verb.

In Figure 2.1, we provide an example of a dependency parse for the given sentence.[3] As is shown, we will drawn dependency parses in this canonicalized format where all arcs are strictly above the linear sequence of words. If the dependency parse, when depicted this way, has no intersecting edges (i.e. the drawing is a con-

[3]We do not illustrate the direction or labels of any dependency relations. The reasons for doing so will be made clear in §4.

structive proof that the underlying graph is planar), we call the dependency parse *projective* (Hays, 1964). Under many theories for dependency grammars, most/all sentences in most/all languages are argued to satisfy projectivity constraints. In particular, violations of projectivity in English are very infrequent (Gildea and Temperley, 2007) and McDonald et al. (2005a) estimates that in Czech, a language that is one of the most frequent to violate projectivity, non-projective sentences constitute less than 2% of all sentences. We revisit the notion of projectivity, as it will prove to be useful for algorithms we subsequently study, in §4.3.1.

2.2.1 Dependency Parsing

In this work, we consider sentences that are both annotated and not annotated with a gold-standard dependency parse. When sentences are annotated, they are taken from the Universal Dependencies Treebank[4] and were introduced by Nivre et al. (2016) with annotations following the Universal Dependencies dependency formalism. When sentences are not annotated, we parse them using off-the-shelf pretrained parsed that we describe in later sections. In particular, we strictly consider unannotated data for English. In English, there exist several high-quality pretrained parsers (Dozat and Manning, 2016; Dozat et al., 2017; Honnibal and Montani, 2017; Shi and Lee, 2018) and dependency parsing is relatively mature. Comparatively, for other natural languages, and especially low-resource languages, off-the-shelf dependency parsing is less viable (Vania et al., 2019) and we revisit

[4] https://universaldependencies.org/

this in §6.5.

2.3 Word Order in Natural Language Processing

In order to understand how word order should be modelled computationally, we begin by cataloguing the dominant approaches to word order in the NLP literature. We revisit the most pertinent methods more extensively in §5.

2.3.1 Order-Agnostic Models

Given the challenges of modelling word order faithfully, several approaches in NLP to word order have entirely sacrificed modelling order to prioritize other pertinent phenomena. In some settings, where document-scale representations are desired, it has been argued that the nuances of word order within sentences is fairly marginal. Two well-studied regimes are the design of *topic models* and *word embeddings*.

Topic Models. Topic models are (probabilistic) generative models of text collections that posit that the texts are generated from a small set of latent *topics*. This tradition of proposing generative models of text originates in information retrieval (Salton and McGill, 1986) and has led to a series of works towards designing topic models that yield topics that well-aligned with human notions of topics. Almost all topic

models represent documents by their bag-of-words representation, hence neglecting order. The most famous topic model is *Latent Dirichlet Allocation* (LDA) (Blei et al., 2003), which proposes a hierarchical Bayesian approach towards generative modelling; model parameters can be efficiently inferred via Markov Chain Monte Carlo (Griffiths and Steyvers, 2004), variational inference (Blei et al., 2003; Hoffman et al., 2010), Bayesian Poisson factorization (Gopalan et al., 2015; Schein et al., 2015, 2019), and spectral methods using either method of moments (Anandkumar et al., 2012) or anchor words/separability assumptions (Arora et al., 2012, 2013; Lee et al., 2019). Order-agnostic topic models have seen a wide array of applications in computational social science and the digital humanities; contributions have been made via textual analysis to the disciplines of political science (Gerrish and Blei, 2012), literature (Underwood, 2012), and history (Newman and Block, 2006) among several others. For a more extensive consideration of topic models, see Alghamdi and Alfalqi (2015); Jelodar et al. (2019); Schofield (2019).

Word Embeddings. Word embeddings (Bengio et al., 2003; Collobert and Weston, 2008; Collobert et al., 2011) are learned mappings from lexical items to vectors that encode natural language semantics in vector spaces. Most word embeddings hinge on a particular interpretation of the distributional hypothesis (Harris, 1954; Firth, 1957). Classical methods such as LSA (Deerwester et al., 1990) factorized co-occurrence statistics whereas more recent neural methods (Mikolov et al., 2013; Pennington et al., 2014; Bojanowski et al., 2017) predict various co-occurrence statistics (Baroni et al., 2014). In both cases, most approaches are largely order-agnostic

(or use low order n-gram statistics) and subsequent work has shown that many neural methods for word embedding can be re-interpreted as factorizations (of the pointwise mutual information) as well (Levy and Goldberg, 2014; Levy et al., 2015; Ethayarajh et al., 2019). Similar to topic models, word embeddings have seen a wide array of applications not just within NLP (as initializations using pretrained word representations) but also beyond NLP including to study diachronic societal biases (Garg et al., 2018) and cultural associations (Kozlowski et al., 2019). For a more extensive consideration of word embeddings, see Wang et al. (2019c); Faruqui (2016).

Bag-of-Words Classifiers. While topic models and word embeddings learn representations from a collection of documents, bag-of-words and order-agnostic techniques have also been considered in building representations of language within a document and even within a sentence. Many of these methods are classical approaches to text classification. Initial approaches adapted standard algorithms from the machine learning community (Mitchell, 1997) for linear and log-linear classification such as the Naive Bayes and maximum entropy algorithms (Lewis and Gale, 1994; Berger et al., 1996; Domingos and Pazzani, 1997; McCallum and Nigam, 1998; Lewis, 1998; Pang et al., 2002), whereas later works considered nonlinear classifiers such as SVMs (Joachims, 1998; Pang et al., 2002) and feed-forward neural networks (Collobert and Weston, 2008; Collobert et al., 2011). Simultaneously, many of these works such as those of Lewis (1998) and Turney (2002) had their origins in information retrieval and information theory. In these works, it was standard to use order-agnostic *term frequency* (TF) (Luhn, 1957) and *inverse document frequency* (IDF)

(Spärck Jones, 1972) features, commonly under the joint framing of the TF-IDF weighting schema (Salton, 1991) as in the Rocchio classifier (Rocchio, 1971; Joachims, 1997). Comprehensive surveys and analyses of models for text classification are provided by Yang and Pedersen (1997); Yang and Liu (1999); Yang (1999); Aggarwal and Zhai (2012).

Order-Agnostic Sentence Encoders. Following the introduction of Word2Vec and neural networks in NLP in the early 2010's, the community gravitated towards deep learning approaches that no longer required explicitly feature engineering. Consequently, order-agnostic approaches within sentences became less frequent. Nonetheless, order-agnostic representation learning over word representations for sentence encoding has proven to be effective as a strong (and cheap) baseline. Learning-based order-agnostic sentence encoding often uses variants of deep averaging networks for text classification tasks (Iyyer et al., 2015; Chen et al., 2018). However, subsequent work showed that the deep averaging was unnecessary and that simple averaging was sufficient (Wieting et al., 2016). Additionally, some works have viewed averaging methods theoretically (often as random walks) (Arora et al., 2016; Ethayarajh, 2018) and different weighting schema have emerged to better encode the fact that word-level representations do not contribute uniformly towards the meaning of a sequence (Arora et al., 2017).

2.3.2 Sequential Models

Given that natural language has an explicit sequential structure and this structure is informative (hence our interest in word order), a large family of approaches in NLP have attempted to model the sequential nature directly.

Markov Models. Markov models are a family of statistical models which make *Markovian assumptions* — assumptions that strictly bound the length of dependencies that can be modelled. In particular, a Markov model of Markov order n cannot model a distance of length at least $n + 1$ directly. Nonetheless, a recent line of theoretical results suggest that there are workarounds for modelling long-distance dependencies in such models (Sharan et al., 2017, 2018). Within NLP, hidden Markov models (HMMs) have been used for a variety of sequence-tagging applications including part-of-speech tagging, named entity recognition, and information extraction (Jelinek, 1976; Freitag and McCallum, 1999, 2000; Toutanova et al., 2002; Collins, 2002). In using HMMs in NLP, the causal factorization of the desired probabilities is generally estimated using n-gram statistics. In maximum entropy Markov models (MEMMs), a maximum entropy classifier is introduced to add expressiveness and this has been shown to be more effective in most settings (Lau et al., 1993; Ratnaparkhi et al., 1994; Ratnaparkhi, 1996; Reynar and Ratnaparkhi, 1997; Toutanova and Manning, 2000; McCallum et al., 2000). Alternatively, conditional random fields (CRFs) proved to be effective in weakening the strong

independence assumptions that are built into HMMs and the biases[5] that are inherent to MEMMs (Lafferty et al., 2001; Sha and Pereira, 2003; Pinto et al., 2003; Roark et al., 2004; Peng et al., 2004; Sutton et al., 2007; Sutton and McCallum, 2012).

Parsing. Sequence-tagging problems, which were extensively studied using Markov models, are a special case of *structured prediction* problems that are prevalent in NLP. In the well-studied setting of parsing, whether it was syntactic constituency parsing, syntactic dependency parsing, or semantic parsing, several approaches were taken to jointly model the structure of the parse and the sequential structure of language (Kay, 1967; Earley, 1970; Charniak, 1983; Pereira and Warren, 1983; Kay, 1986, 1989; Eisner, 1996; Collins, 1996, 1997; Charniak et al., 1998; Gildea and Jurafsky, 2002; Collins, 2003; Klein and Manning, 2003b,a; Taskar et al., 2004; McDonald et al., 2005b; McDonald and Pereira, 2006; Chen and Manning, 2014; Dozat and Manning, 2016; Dozat et al., 2017; Shi et al., 2017a,b; Gómez-Rodríguez et al., 2018; Shi and Lee, 2018, 2020). When compared to other settings where sequential modelling is required in NLP, parsing often invokes highly-specialized routines that center on the unique and rich structure involved.

Recurrent Neural Networks. Given the cognitive motivations for modelling language sequentially in computational methods, Elman (1990) pioneered the use of recurrent neural networks (RNNs). While these networks have a connectionist

[5] Towards states that had few successors.

interpretation (Rumelhart et al., 1986; Jordan, 1989), they ultimately proved to be ineffective due to technical challenges such as vanishing/exploding gradients in representing long-distance relationships. Consequently, later works introduced gated networks such as the long short-term memory (LSTM) network (Hochreiter and Schmidhuber, 1997). Analogous to the dramatic performance improvements experienced due to word embeddings such as Word2Vec, the community observed similarly benefits in the early to mid 2010's due to LSTMs. This prompted further inquiry into a variety of RNN variants (e.g. Cho et al., 2014a; Bradbury et al., 2017; Lei et al., 2018; Melis et al., 2020). More recently, a line of theoretical works has worked towards classifying the theoretical differences between these variants (Schwartz et al., 2018; Weiss et al., 2018; Peng et al., 2018; Suzgun et al., 2019b,a; Merrill, 2019; Lin et al., 2019). This has recently culminated in the work of Merrill et al. (2020) which establishes a formal taxonomy that resolves the relationship between various RNN varieties and other methods from classical automata theory such as weighted finite state machines.

Attention. The emergence of neural networks in NLP for sequence modelling naturally led to their adoption in natural language generation tasks such as machine translation (Kalchbrenner and Blunsom, 2013; Cho et al., 2014b; Sutskever et al., 2014) and summarization (Rush et al., 2015; Chopra et al., 2016; Nallapati et al., 2016). In these settings, attention came to be a prominent modelling innovation to help induce alignment between the source and target sequences (Bahdanau et al., 2015; Luong et al., 2015). Since then, attention has seen application in many

other settings that involve sequential modelling in NLP as it enables networks to model long-distance dependencies that would be otherwise difficult to model due to the sequential/recurrent nature of the networks. Given attention's widespread adoption, a line of work has been dedicated to adding sparsity and structure to attention (Martins and Astudillo, 2016; Kim et al., 2017; Niculae and Blondel, 2017; Mensch and Blondel, 2018; Malaviya et al., 2018; Peters et al., 2018a; Niculae, 2018; cor; Peters et al., 2019a) whereas a separate line of work has studied its potential utility as an interpretability tool for explaining model behavior (Jain and Wallace, 2019; Serrano and Smith, 2019; Strout et al., 2019; Wiegreffe and Pinter, 2019; Pruthi et al., 2020).

2.3.3 Position-Aware Models

Sequential models directly model the sequential nature of language. In recent years, there has been an emergence and considerable shift towards using position-aware models/set encoders. In particular, these models implicitly choose to represent a sequence $\langle w_1 \dots w_n \rangle$ as the set $\{(w_i, i) \mid i \in [n]\}$[6] as was described in Vinyals et al. (2016). In this sense, the encoder is aware of the position but does not explicitly model order (e.g. there is no explicit notion of adjacency or contiguous spans in this encoding process). Early works in relation extraction also considered position-

[6]The correspondence between arbitrary sequences and sets of this structure is bijective

aware representations (Zhang et al., 2017).[7]

Transformers. Vaswani et al. (2017) introduced the Transformer architecture, which has become the dominant position-aware architecture in modern NLP. In particular, all sequences are split into 512 subword units and subwords are assigned lexical embeddings and position embeddings, which are then summed to yield non-contextual subword representations. These 512 subword vectors are then iteratively passed through a series of Transformer layers, which decompose into a self-attentive layer[8] and a feed-forward layer. Since these operations are fully parallelizable, as they have no sequential dependence, large-scale training of Transformers on GPU/TPU computing resources has propelled performance forward on a number of tasks. Similarly, since these models can compute on more data per unit time than sequential models like LSTMS[9], they have led to a series of massive pretrained models that include: GPT (Radford et al., 2018), BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019). SpanBERT (Joshi et al., 2020a), ELECTRA (Clark et al., 2020), ALBERT (Lan et al., 2020) and T5 (Raffel et al., 2019).

Position Representations. Given that what differentiates position-aware models

[7]To the author's knowledge, this observation and citing of Vinyals et al. (2016) and Zhang et al. (2017) has been entirely neglected in all past works in the NLP community (c.f. Vaswani et al., 2017; Dai et al., 2019).

[8]Self-attention is attention in the sense of Bahdanau et al. (2015) where the given sequence is used in both roles in the attention computation.

[9]Given the constraints of current hardware.

from order-agnostic models is their position representations, surprisingly little work has considered these representations (Bommasani and Cardie, 2019). In the original Transformer paper, position embeddings were frozen using cosine waves to initialize them. Recent work has put forth alternative approaches for encoding position (Almarwani et al., 2019). In particular, Wang et al. (2020) demonstrate that using complex-valued vectors, where the amplitude corresponds to the lexical identity and the periodicity corresponds to the variation in position, can be a principled theoretical approach for better modelling word order in Transformer models. Separately, Shaw et al. (2018) and Dai et al. (2019) argue for encoding position in a relative fashion to accommodate modelling longer sequences (as the standard Transformer is constrained to 512 positions).

2.3.4 Alternative Word Orders

Given that natural language processing tasks often requiring understanding an input text, it is unsurprising that most works which model the input in an order-dependent way (generally implicitly) choose to specify the word order to be the same as the order already given in the input. A frequent exception is bidirectional models, which have seen applications in a number of settings. Beyond this, other approaches have considered alternative word orders as a mechanism for studying alignment between different sequences. Much of this literature has centered on machine translation.

Bidirectional Models. One natural choice for an alternative order is to use the reverse of the order given. For a language such as English which is read from left-to-right, this would mean the order given by reading the input sequence from right-to-left. While some works have compared between left-to-right models and right-to-left models (Sutskever et al., 2014), in most downstream settings, bidirectional models are preferred. A bidirectional model is simply one that integrates both the left-to-right and right-to-left models; the bidirectional RNN is a classic model of this type (Schuster and Paliwal, 1997). Shallowly bidirectional models do this by independently modelling the input from left-to-right and right-to-left and subsequently combining (generally by concatenation or vector addition) the resulting output representations. Such approaches have seen widespread application in NLP; the ELMo pretrained model is trained in a shallowly bidirectional fashion (Peters et al., 2018b). In comparison, with the emergence of Transformers, it is possible to process part of the input (e.g. a single token) while conditioning on the entirety of the remainder of the input at once. Such models are often referred to as deeply bidirectional; BERT (Devlin et al., 2019) is a model pretrained in this way by making use of a denoising objective in masked language modelling[10] as opposed to the standard causal language modelling used in ELMo.

Permutation Models. From a language modelling perspective, a unidirectional left-to-right (causal) language model factorizes the sequence probability $p(\langle w_1 \dots w_n \rangle)$

[10]Masked language modelling is a cloze task where the objective is to predict the masked word in the input sequence conditional on the remainder of the sequence, which is unmasked.

as

$$p\left(\langle w_1 \dots w_n \rangle\right) = \prod_{i=1}^{n} p\left(w_i \mid \langle w_1 \dots w_{i-1} \rangle\right). \tag{2.1}$$

In comparison, a unidirectional right-to-left language model factorizes the sequence probability as

$$p\left(\langle w_1 \dots w_n \rangle\right) = \prod_{i=1}^{n} p\left(w_i \mid \langle w_{i+1} \dots w_n \rangle\right). \tag{2.2}$$

In the recent work of Yang et al. (2019), the authors introduce a strikingly new approach which generalizes this perspective. In particular, any given ordering of the sequence $\langle w_1 \dots w_n \rangle$ corresponds to a unique factorization of this sequence probability. In their model, XLNet, the authors sample factorizations uniformly (hence considering the behavior in expectation across all $n!$ possible factorizations) and, alongside other modelling innovations, demonstrate that this can be effective in language modelling. As we will demonstrate, our approach could be seen adopting the perspective of trying to identify a single optimal order than sampling from all possible orders with equal likelihood.

Order Alignment. For tasks that involve multiple sequences, order plays an additional role of facilitating (or inhibiting) alignment between the difference sequences. In machine translation, the notion of alignment between the source and target languages is particularly important.[11] As a consequence, two sets of approaches towards ensuring improved alignment (explicitly) are *preorders* (changing the order of

[11] In fact, attention (Bahdanau et al., 2015; Luong et al., 2015) emerged in machine translation precisely for the purpose of better aligning the fixed input sequence and the autoregressively generated output sequence.

the source language input to resemble the target language) and *postorders* (changing the order of a monotone output translation to resemble the target language).

- *Preorders* — Preorders have been well-studied in several machine translation settings. In particular, preorders have been designed using handcrafted rules (Brown et al., 1992; Collins et al., 2005; Wang et al., 2007; Xu et al., 2009; Chang et al., 2009), using learned reorderings/rewritings based on syntactic patterns (Xia and McCord, 2004; Li et al., 2007; Genzel, 2010; Dyer and Resnik, 2010; Katz-Brown et al., 2011; Lerner and Petrov, 2013), or based on learning-based methods that induce hierarchical features instead of exploiting overt syntactic cues (Tromble and Eisner, 2009; DeNero and Uszkoreit, 2011; Visweswariah et al., 2011; Neubig et al., 2012). Much of the early work in this setting worked towards integrating the up-and-coming[12] (phrase-based) statistical machine translation with the longstanding tradition of using syntax in machine translation. With the emergence of purely neural machine translation, recent work has studied how to integrate preorders in an end-to-end fashion using neural methods as well (Hoshino et al., 2014; de Gispert et al., 2015; Botha et al., 2017; Kawara et al., 2018). Especially relevant to the current thesis is the work of Daiber et al. (2016), which studies the relationship between preorders (and their effectiveness) and the flexibility in word orders in different languages.
- *Postorders* — Given that there are two sequences involved in machine translation, it is natural to consider postorders as the complement to preorders.

[12] At the time.

However, there is a fundamental asymmetry in that preorders involve changing the input (which can be arbitrarily interacted with) whereas postorders involve changing the output after it has been generated. Therefore postorders require more complex inference procedures and (ideally) require joint training procedures. Given this, postorders have been comparatively under-studied and little evidence has been provided to indicate that there are significant advantages to compensate for these substantial complications when compared to preorders. The one caveat is when developing preorders would be challenging. For example, while generate a preorder for English to Japanese may be viable, generating a preorder for Japanese to English is far more complicated (due to the syntactic patterning of both languages). Therefore, one may use a preorder to improve English to Japanese translation but would struggle to do the same for improving Japanese to English translation. Given these difficulties, a postorder may be attractive in the Japanese to English setting as it is reminiscent of the English to Japanese preorder (and can leverage insights/learned features/parameters from generating an English to Japanese preorder). Sudoh et al. (2011) introduced postorders for precisely this reason and Goto et al. (2012) extended their method with superior postordering techniques. Further, Mehta et al. (2015) introduced an oracle algorithm for generating orders for ten Indian languages but their work received little traction thereafter due to empirical shortcomings.

While reordering to induce alignment has received the most interest in the machine translation, the phenomena is arguably more general. In the extreme, it may

be appropriate in *every* task where there are multiple sequences of any type. In particular, Wang and Eisner (2018) propose the inspired approach of constructing synthetic languages from high-resource languages (where parsing data is available) whose word order mimics a low-resource language of interest (where parsing data is unavailable/limited) to facilitate cross-lingual transfer in dependency parsing. Rasooli and Collins (2019) also consider a similarly reordering method on the source side to improve cross-lingual transfer in dependency parsing. In particular, it is likely that a similar approach may be of more general value in designing cross-lingual and multi-lingual methods, especially when in the low-resource regime for the language of interest. Very recently, Goyal and Durrett (2020) propose to adapt ideas from work on preorders in machine translation to generate paraphrase. In particular, they repurpose the notion of preorders to construct controllable and flexible preorders based on learned syntactic variations. While most other subareas of NLP have yet to consider word order in dynamic ways, the findings of Wang and Eisner (2016) may prove to be a valuable resource for such study. In this work, the authors introduce the Galactic Treebank, which is a collection of hundreds of synthetic languages that are constructed as hybrids or mutations of real/attested human languages (by intertwining the word order/syntactic patterns of the real natural languages to produce mutants).

CHAPTER 3

WORD ORDER IN HUMAN LANGUAGE PROCESSING

In this chapter, we examine how word order manifests across languages and within certain contexts. We go on to discuss the relationship between word order and sequential processing, honing in on a memory-based theory known as dependency length minimization.

3.1 Ordering Behaviors

The interpretation of word order is highly language-specific. In particular, the premise that ordering information is meaningful to begin with is itself language-dependent. Languages with *fixed* or *rigid* word orders tend to reliably order constituents in a certain way to convey grammaticality. English is an example of such a language. On the other hand, other languages, such as Russian and Nunggubuyu, may have more flexible word orders and are therefore said to have *free* or *flexible* word orders. Within these languages, some, like Russian, may exhibit multiple word ordering structures but prefer one in most settings; this is known as the *dominant* word order. For other languages, there is no dominant word order, as is the case for Nunggubuyu (Heath, 1984). In languages with flexible word orders, morphological markings (such as inflection) are frequently employed to convey information to listeners/comprehenders. In particular, Comrie (1981) and Haspelmath (1999) have argued that it is precisely these morphological markings that

allow flexible word order languages to "compensate" for the information that is not encoder in word order.[1] In discussing word order, it is natural to narrow the scope to certain aspects that are of linguistic interest.

Basic Word Orders. The ordering of constituents is a standard method for categorizing languages (Greenberg, 1963). At the coarsest granularity, languages can exhibit different canonical orderings of the *subject* (S), *main verb* (V), and *object* (O) within sentences that feature all three.[2] In fact, it is standard to refer to this as the language's *basic word order*. In Table 3.1, we depict languages that attest each of the six possible arrangements of S, V, and O as well as typological statistics regarding their relative frequencies. In general, we observe that subject-initial languages constitute an overwhelming fraction of the world's languages and that OSV is the minority ordering by a considerable margin. While such analyses are incomplete (Dryer, 2013a)[3], they offer an immediate illustration that word ordering properties can be of interest typologically (Dryer, 1997, 2013b). Next, we consider whether these order properties can be used to deduce underlying properties of language

[1] These claims have been disputed by Müller (2002), but the concern here is the causal relationship between flexible word order and morphological markers. In particular, prior works contest that morphological case is prerequisite to free word order whereas (Müller, 2002) finds evidence to the contrary. We take no position on this and simply note that morphological markings and flexible word orders often co-occur.

[2] From a linguistic perspective, the terms subject and object are ill-specified. In accordance with standard practice, we will think of the subject as the noun or noun phrase that generally exhibits agent-like properties and the object as the noun or noun phrase that generally exhibits patient-like properties.

[3] As many languages exhibit different basic word orders across sentences whereas others. In particular, in a language like German, both SOV and SVO orderings are quite common across sentences. Alternatively, in languages such as Latin and Wampiri, constituents may not be contiguous spans, which may complicate the notion of ordering constituents.

as a whole and whether we can formulate theories to explain why these orders arise.

Ordering	% Languages	Example Language	Reference
SOV	40.99	Japanese	(Kuno, 1973)
SVO	35.47	Mandarin	(Li and Thompson, 1981)
VSO	6.90	Irish	(Dillon and Ó Cróinin, 1961)
VOS	1.82	Nias	(Brown, 2001)
OVS	0.80	Hixkaryana	(Derbyshire, 1979)
OSV	0.29	Nadëb	(Weir, 1994)

Table 3.1: Basic word orders across the world's languages. Statistics regarding the fraction of the world's languages that primarily use a certain ordering come from Dryer (2013a). 1376 natural languages were the total number of languages in considering these statistics. References refer to entire works dedicated to studying the corresponding language which rigorously demonstrate the language's dominant word order. The unexplained probability mass corresponds to languages without a dominant word order (e.g. German) or with discontiguous constituents (e.g Wampiri).

3.2 Language Universals

Given the set of word ordering effects we have seen so far, it is natural to ask whether certain patterns emerge across language languages. More strongly, one can question whether there are certain *universal* properties which exist (and such hypotheses can be readily tested with experimental and statistical techniques at present). Greenberg (1963) initiated this study, with early work towards studying the basic word orders we have seen previously. Greenberg argued that there are three determining factors that specific a *basic typology* over languages:

1. A language's basic word order

2. The prevalence of *prepositions* or *postpositions*. In languages such as Turkish, arguments of a constituent systematically appear before it. In particular, adjectives precede nouns, objects precede words, adverbs precede adjectives, and so forth. For this reason, such a language is labelled **pre**positional. In contrast, in languages such as Thai, the argument of a constituent systematically appears after it. For this reason, such a language is labelled **post**positional. Since many languages, such as English display both prepositional behavior (e.g. adjectives before nouns) and postpositional behavior (e.g. objects after verbs), Greenberg determined the more prevalent of the two to assign this binary feature to languages.[4]

3. The relative position of adjectives with respect to the nouns they modify. Again, in English, the adjective precedes the noun whereas in Thai, the adjective follows the noun.

Given these features, there are $24 = 6 \times 2 \times 2$ possible feature triples that a language could display. As the wording of items 2 and 3 suggests, these can be viewed as instances of a broader class of local ordering preferences, we return to this point later in this section. Greenberg excluded all basic word orders that had objects preceding subjects since he argued that these were never dominant word orders in a language.[5] Greenberg then studied 30 natural languages and categorized them

[4]Greenberg did not consider circumpositional languages, such as Pashto and Kurdish, where aspects of the argument appear on either side of the constituent. Circumposition is generally observed more frequently at the morphological rather than syntactic level.

[5]While this claim is false in general, it can be argued to be true for the languages Greenberg studied.

into each of these 12 groups. While the statistical validity of his work has been questioned (Dryer, 1988; Hawkins, 1990; Dryer, 1998), subsequent works (especially in recent times when data is more readily accessible and large-scale corpus analyses can be conducted computationally) have clarified the validity of his theories (e.g. Dryer, 1992, 2013a; Hahn et al., 2020). More generally, the enterprise Greenberg initiated of unearthing *language universals* based on consistent patterns across a set of sampled languages has spawn important lines of work in cognitive science and linguistics.

Harmonic Word Orders. Of the language universals that Greenberg put forth, perhaps the most notable have been the harmonic word orders. The term *harmonic* refers to the fact that in some languages, the modifiers of a certain syntactic class (e.g. nouns) consistently either precede or succeed the class. For example, many languages have both numerals and adjectives precede the noun or both succeed the noun as compared to language where one precedes and the other follows; the latter collection of languages are referred to as *disharmonic*. While there has been significant inquiry towards enumerating languages and the types of (dis)harmonies observed (see Hawkins, 1983), our interest in harmonic word orders is the cognitive approach towards understanding how they may influence learning. In this sense, harmonic word orders have emerged as a direct line of attack for cognitive inquiry towards connecting word ordering effects, language learning and acquisition, and broader theories of human cognition and language processing.

In general, consideration of word order harmonies can be attributed to the reliable and overwhelming statistical evidence. Given this evidence, it is natural to further consider whether a broader cognitive explanation that extends beyond linguistics may be the source for the observed phenomena. One especially relevant line of work has argued that a bias towards word order harmonies can be indicative of general cognitive and/or processing constraints for humans (Culbertson and Kirby, 2016). In this sense, word order harmonies contribute to simpler grammars and a proclivity for shorter dependencies that is seen across other domains for human cognition. Culbertson et al. (2012) strengthen this position by demonstrating that adult language learners learning artificial/synthetic languages demonstrate strong tendencies towards word order harmonies. Culbertson and Newport (2015) further extend these results by showing similar behaviors for child language learners while clarifying the distinction with respect to adult language learners regarding the strength and nature of the bias towards harmonic word orders. More recently, Culbertson and Newport (2017) provide fairly resolute confirmation of this theory and separation of adult and child language learning with regards to harmonic word orders. When both children and adults are tasks with learning languages that are regularly disharmonic, children fail to learn the language correctly and instead innovate/fabricate novel word orders which are harmonic (where the correct harmonic is disharmonic). In contrast, adults are able to replicate the nonharmonic patterns correctly.

In our work, while we do not directly appeal to cognitive results for language

learning (especially for children), we take this to be motivation that insightful choice of word orders (perhaps in a way that aligns with a learner's inductive bias) can facilitate language acquisition. Conversely, suboptimal choices may complicate language learning substantially and can cause humans (and potentially machines) to resort to alternative strategies that better reconcile the nature of the input with the underlying latent biases.

3.3 Sequential and Incremental Processing

In the previous section, we catalogued a series of word ordering effects in natural language. Subsequent work has tried to directly explain the word ordering effects and the potential underlying language universals (e.g. Hawkins, 1988) In many of these cases, the corresponding works in linguistics, psycholinguistics, or cognitive science that studied these phenomena either offered theoretical explanations or empirical evidence. However, a loftier goal for psycholinguistics in particular is to create a broader theory for sequential language processing. In particular, such a theory might explain the word ordering behaviors we have described previously as special cases.

Language is processed incrementally (Sag et al., 2003). Consequently, any theory that explains general sequential language processing must grapple with this property of how humans process language. In the study of incremental language

processing, the *integration function* is defined to be the function describing the processing difficulty in processing a given word w_i given the preceding context $\langle w_1 \dots w_{i-1} \rangle$ (Ford et al. (1982), c.f. Tanenhaus and Trueswell, 1995; Gibson and Pearlmutter, 1998; Jurafsky, 2003).[6] In both theoretical and empirical inquiry towards understand human incremental language processing, most works make use of some mechanism that allows for controlled variation (e.g. minimal pair constructions) in the input and analyze the incremental processing difficulty of a human(s) comprehending the input. In empirical work, this analysis is often executed by considering differential effects using a measurement mechanism for human processing (e.g reading times, reading from the scalp, pupil dilation).

The consequence of this work is a canonicalized pair of theories: expectation-based incremental language processing and memory-based incremental language processing. The central tenet of the former is that most processing is done pre-emptively, since many words can be predicted by their context[7] and any further difficulty can be attributed to how surprising w_i is given the preceding context. In contrast, the latter theory posits that the integration cost of the current word w_i is proportional to the challenges of integrating it with units that must have been retained in memory. Given the longstanding tradition of studying incremental

[6] In some works (e.g. Venhuizen et al., 2019), additional context beyond the preceding linguistic context, such as the social context or world knowledge, is further modelled. We deliberately neglect any description of such work in our review of past work as we restrict ourselves to language understanding and language modelling that is fully communicated via the preceding linguistic context throughout this thesis.

[7] It is this principle that motivates causal language modelling.

language processing, joint theories that seek to reconcile the approaches have also been proposed. In particular, given there is strong evidence for both theories (and both often have been showed to be reasonably complementary in their explanatory power), joint theories seek to merge the two, as there is undisputed proof of both predictive processing and memory-driven forgetting in human language processing.

3.3.1 Expectation-based Theories

In positing a theory of incremental processing that hinges on prediction, it is necessary to operationalize what is predicted and how processing difficulty emerges from failures in prediction. For this reason, expectation-based theories have largely come to be dominated by surprisal-based theories (Hale, 2001; Levy, 2008a), as the predictions given rise to the processing difficulty inherently. In particular, surprisal is an information-theoretic measure that measures how *surprised* or unlikely a word w_i is given the preceding context $\langle w_1 \dots w_{i-1} \rangle$ as

$$\texttt{surp}\left(w_i \mid \langle w_1 \dots w_{i-1} \rangle\right) \triangleq -\log\left(p\left(w_i \mid \langle w_1 \dots w_{i-1} \rangle\right)\right). \tag{3.1}$$

We will use $\texttt{surp}_\theta$ as notation to denote when the probability distribution p is estimated using a model parameterized by weights θ. From a modelling perspective, many methods have been used to estimate surprisal. In particular, probabilistic context-free grammars (Hale, 2001; Levy, 2008a), classical n-gram language models

(Smith and Levy, 2013), recurrent neural network language models (van Schijndel and Linzen, 2018) and syntactically-enriched recurrent neural networks grammars (Dyer et al., 2016; Hale et al., 2018) have all been used as language models, i.e choices of θ, to estimate this probability. Crucially, surprisal has been shown to be a reliable predictor of human reading times (robust to six orders of magnitude) by Smith and Levy (2013).

Surprisal has emerged to be a workhorse of several lines of psycholinguistic inquiry since it provides a natural and strong linking hypothesis between density estimation and human behavior as well as due to its information-theoretic interpretation. In particular, surprisal can be attributed as exactly specifying the change to a representation that is caused by the given word w_i where the representation encodes $\langle w_1 \dots w_{i-1} \rangle$, i.e. the sequence seen so far. In this sense, surprisal codifies the optimal Bayesian behavior and has come to be part of a broader theory of cognition centered on prediction and predictive coding (Friston and Kiebel, 2009; Clark, 2013). Further, since it specifies the purely optimal behavior, surprisal retains both the advantages and disadvantages associated with being *representation-agnostic*. We will revisit these in considering motivations for joint theories.

Given these findings, among many others, surprisal theory has strong explanatory power in describing human incremental language processing. As it pertains

to this thesis, surprisal has also been recently[8] considered for the purposes of explaining word ordering behaviors. In particular, Hahn et al. (2018) demonstrate that surprisal and other information theoretic measures, such as point-wise mutual information, can be used to explain adjective ordering preferences in the sense of Greenberg (Greenberg, 1963). In particular, they are able to predict adjective orders reliably (96.2% accuracy) using their cognitive model that is grounded in mutual information and makes use of memory constraints. Futrell (2019) also provides similar evidence for word ordering behaviors being explained effectively via the use of information theory. Very recently, Hahn et al. (2020) strengthened this position by showcasing that surprisal-based methods can be used to demonstrate that Greenberg's language universals emerge out of efficient optimization within language to facilitate communication.

3.3.2 Memory-based Theories

Under memory-based theories of incremental processing, the processing difficulty of associated with the current word w_i is proportional to the difficulty in/error associated with retrieving units from the context $\langle w_1 \dots w_{i-1} \rangle$. In particular, consider the following four examples (reproduced from Futrell et al., 2020):

[8] Prior works (e.g Ferrer-i Cancho and Solé, 2003; Ferrer-i Cancho, 2006) also considered information theoretic approaches to language to explain word orders but were considerably less effective than the recent line of work. Further, these works used information theoretical tools but did not necessarily appeal to the expectation-based theories which we consider here.

(1) a. Bob **threw out** the trash.

b. Bob **threw** the trash **out**.

c. Bob **threw out** the old trash that had been sitting in the kitchen for several days.

d. Bob **threw** the old trash that had been sitting in the kitchen for several days **out**.

Observe that in the first pair of sentences, the sentences are perfectly lexically-matched and both convey identical semantic interpretations. For humans, these sentences have similar processing complexity. However, in the latter pair of sentences, while they are again perfectly lexically-matched and again convey identical semantic interpretations, they have starkly different processing complexities. Humans systematically find sentence (1d) to be more challenging to process than sentence (1c), as has been observed by Lohse et al. (2004). Under memory-based theories, many of which stem from the dependency locality theory of Gibson (1998, 2000), this difficulty arises due to the increased length of the dependency between **threw** and **out**. In other words, in (1d), a human must retrieve the information regarding **threw** when processing **out** and the error in this retrieval or its general difficulty increases as a function of the dependency's length. In particular, to interpret any of these four sentences, it is necessary to process the syntactic dependency linking **threw** and **out**; it is insufficient to only process only one lexical item or the other to obtain the correct semantic interpretation (Jackendoff, 2002).

Several hypotheses have been proposed to explain what underlying mechanisms explain the observed increase in dependency as a function of length. Some posit that there is an inherent decay in the quality of the representation in memory over time (consistent with other types of memory representations throughout human cognition) whereas others argue that the degradation is tightly connected with the nature of the intervening material and how it interferes with flawless retention of the context. Regardless, there are numerous effects in linguistics where processing difficulty has been showed to increase with increasing dependency length (e.g. multiple center-embeddings, prepositional phrase attachment; c.f. Futrell et al., 2020). Akin to surprisal theories, there is also evidence that dependency locality and memory-based theories are predictive of human behaviors (Grodner and Gibson, 2005; Bartek et al., 2011). However, some subsequent works have questioned whether dependency locality effects are strong predictors of human behavior beyond the laboratory setting; Demberg and Keller (2008a) find no such effects when evaluating using naturalistic reading time data.

3.3.3 Joint Theories

Given the representation-agnostic nature of expectation-based and surprisal theories of incremental processing and the representation-dependent nature of memory-based theories, joint theories must commit to being either representation-agnostic or representation-dependent, thereby adopting one theory as a basis. Then, these approaches engineer mechanisms by which to integrate the other theory. In general,

the motivation for studies towards building joint theories is to capitalize on the observation that expectation-based and memory-based theories of incremental processing have been generally shown to explain complementary phenomena.

The Psycholinguistically-Motivated Lexicalized Tree Adjoining Grammar of Demberg and Keller (2008b), which was further extended in Demberg and Keller (2009), Demberg (2010), and Demberg et al. (2013), was one of the first joint approaches. In particular, a parser (under the tree adjoining grammar formalism) is equipped with PREDICT and VERIFY operations. The PREDICT operation is akin to expectation-based predictions of processing difficulty. Dually, the VERIFY operation is is memory-driven as it requires validating that the previously predicted structures are indeed correct (and the cost of this verification scales in the length of the dependencies/varies inversely in the strength of the dependency locality effects). This approach more broadly adopts the perspective of endowing a representation-dependent framework (here specified using the tree adjoining grammar) with predict operations and further constraints.

Conversely, Futrell et al. (2020) have recently introduced the lossy-context surprisal model which extends the author's previous work on noisy-context surprisal (Levy, 2008b, 2011; Futrell and Levy, 2017). In this work, the authors adopt a representation-agnostic perspective grounded in surprisal theory. Based on the observation that pure surprisal theory, which uses information theoretic primitives,

cannot account for forgetting effects, the authors suggest making the representation of the context *lossy*. What the authors is a more general concern with information theory, in that information theory in the style of Shannon (1948) does not account for models of bounded or imperfect computation. Consequently, if any information can be recovered from the (possibly arbitrarily long) preceding context, information theory will account for this information. Recovering this information without error is likely not viable for humans.[9]

Information Locality. Given the constraints of humans (as have been implicitly shown in the literature on dependency locality), Futrell et al. (2020) argue for a theory of information locality, which was first introduced by Futrell (2019). Under such a theory, a memory representation m_t is build at every timestep t and this representation likely imperfectly encodes $\langle w_1 \dots w_t \rangle$.[10] Consequently, specifying the memory representation (and its forgetting effects) appropriately, via a noise model or other lossy information-encoding mechanism, provides the grounds for addressing the forgetting effects that surprisal theory is ill-equipped to handle. In particular, the authors suggest that operationalizing this by using RNNs with bounded context, as in Alemi et al. (2017); Hahn and Futrell (2019), may be an effective approach. We remark that a separate line of inquiry, that directly studies information theory under computational constraints, may be more elegant and sensible. In particular, the theory of $\mathcal{V}$-information put forth by Xu et al. (2020) may

[9] This fact also likely holds for machines and computational models, which have bounded memory and constrained reasoning capabilities.

[10] If it perfectly encodes the context, pure surprisal theory is recovered.

prove to be a strong formalism for encoding the information theoretic primitives that ground surprisal as well as the bounded computational resources that induce memory/forgetting effects.

3.4 Dependency Length Minimization

Both expectation-based theories such as surprisal theory and memory-based theories such as dependency locality theory have been reasonably effective is explaining human behavior in online language processing. Arguably, the evidence for expectation-based theories is stronger and it is this that prompts the recent development of joint theories that are primarily based on predictive processing (Futrell et al., 2020). However, dependency locality theories also have a longstanding tradition of enjoying explanatory power with respect to word ordering effects. In particular, dependency locality theory naturally predicts that humans will produce sentences that employ word orders that minimize dependency length *ceteris paribus*. While similar statement can be made regarding expectation-based theories — humans use word orders that maximize the predictability of subsequent words — there is comparatively less evidence.[11]

[11]However, it should be noted that recent works such as Futrell (2019) and Futrell et al. (2020) argue for this in instantiating a theory of information locality. In particular, Futrell et al. (2020) argue that the word ordering effects suggested by dependency length minimization are merely estimates or approximations of what is truly predicted under information locality by neglecting effects beyond those marked by syntactic dependencies.

A very early predecessor of dependency length minimization is attributed to Behaghel (1932), who stated that "what belongs together mentally is placed close together".[12] Similarly, Greenberg (1963) also provided early accounts of dependency length minimization. More nuance and statistically valid evidence of dependency length minimization has been discovered for many natural languages. Yamashita and Chang (2001) demonstrated statistically meaningful effects via corpus analysis for Japanese. More recently, Futrell et al. (2015) extended these results by showing strong dependency length minimization (well beyond what would be predicted by random word orders), with $p < 0.0001$ for 35 of the 37 languages considered and $p < 0.01$ for the other languages (Telugu and Latin), by making use of the Universal Dependencies Treebank (Nivre et al., 2016). Additional evidence has been introduced which suggests that the grammars of natural languages are designed such that word orders which necessitate long-distance dependencies are dispreferred (Rijkhoff, 1990; Hawkins, 1990). More broadly, dependency length minimization, and therefore the word order preferences it predicts, is a core aspect of a broader argument presented by Hawkins et al. (1994); Jaeger and Tily (2011); Gibson et al. (2019) that natural language emerges as an efficient symbolic system for facilitating human communication from the perspective of both a speaker (language production) and a listener (language comprehension). An excellent multi-faceted survey of the literature on dependency length minimization is provided by Temperley and Gildea (2018).

[12] This sentence is translated from German, as reproduced by Temperley and Gildea (2018).

Given the ubiquitous and diverse grounds for justifying dependency length minimization, computational research has considered the question of devising provably minimal artificial languages to push dependency length minimization to its extreme. While the associated optimization problem of minimizing the cumulative/average dependency length has previously been studied in the algorithms and theory computer science community, with sufficiently general results (Goldberg and Klipker, 1976; Chung, 1984), Gildea and Temperley (2007) introduce an algorithm for finding the word order that provably minimizes dependency subject to projectivity constraints. We discuss this algorithm in §4.3, finding that the algorithm is marginally incorrect, and study its impacts in §5.5. Further, in the parsing community, biasing parsers to generate short dependencies has proven to be a bona fide heuristic (Collins, 2003; Klein and Manning, 2004; Eisner and Smith, 2005). In Smith and Eisner (2006), the authors note that "95% of dependency links cover ≤ 4 words in English, Bulgarian, and Portuguese; ≤ 5 words in German and Turkish; and ≤ 6 words in Mandarin", which provides further evidence to the fact that dependency lengths are minimized and hence are fairly local.

CHAPTER 4

ALGORITHMIC FRAMING

In this chapter, we introduce the algorithmic perspective that we use to formalize our approach towards studying linear order in natural language.

4.1 Notation

Given a sentence $\bar{s} = \langle w_1, \ldots, w_n \rangle$ and it dependency parse $\mathcal{G}_{\bar{s}} = (\mathcal{V}, \mathcal{E}_\ell)$, we will define $\mathcal{E}$ as the unlabelled and undirected underlying edge set of $\mathcal{E}_\ell$.

Definition 4.1. *Linear layout* — A bijective mapping $\pi : \mathcal{V} \to [n]$.

Therefore, a linear layout specifies an ordering on the vertices of $\mathcal{G}_{\bar{s}}$ or, equivalently, a re-ordering (hence a permutation[1]) of the words in $\bar{s}$. Denote the space of linear layouts on $\bar{s}$ by S_n[2]. Since the linear order of a sentence innately specifies a linear layout, we define the *identity linear layout*.

Definition 4.2. *Identity linear layout* — A linear layout $\pi_I : \mathcal{V} \to [n]$ specified by:

$$\pi_I(w_i) = i \tag{4.1}$$

[1] This is why we denote linear layouts by π.

[2] Formally, S_n denotes the symmetric group on n elements.

Definition 4.3. *Edge distance/length* — A mapping $d_\pi : \mathcal{E} \to \mathbb{N}$ specified by:

$$d_\pi(w_i, w_j) = |\pi(w_i) - \pi(w_j)| \tag{4.2}$$

For example, $d_{\pi_I}(w_i, w_j) = |i - j|$.

We further introduce the sets L_i and R_i which are the set of vertices to the left (or at) position i or the right of position i:

$$L_\pi(i) = \{u \in \mathcal{V} : \pi(u) \leq i\} \text{ and } R_\pi(i) = \{v \in \mathcal{V} : \pi(v) > i\} \tag{4.3}$$

Definition 4.4. *Edge cut* — A mapping $\theta_\pi : [n] \to \mathbb{N}$ specified by:

$$\theta_\pi(i) = \left|\{(u, v) \in \mathcal{E} | u \in L_\pi(i) \wedge v \in R_\pi(i)\}\right| \tag{4.4}$$

For a more complete introduction on linear layouts, see the survey of Díaz et al. (2002) which details both problems and algorithms involving linear layouts.

4.2 Objectives

In studying human language processing, we are inherently constrained to view word order as specified by humans/according to the linear layout π_I. As we consider alternative orders, we begin by assuming we have a dataset $\mathcal{D}$ of N examples. For every sentence $\bar{s}_i = \langle w_1, \ldots, w_n \rangle \in \mathcal{D}$[3], there are many possible orderings. Consequently, we define an *ordering rule* $r : \mathcal{D} \to S_n$ as a mapping which specifies a re-ordering for every sentence in $\mathcal{D}$. Given that there are a superexponential number of ordering rules $(n!^N)$[4], it is intractable to explicitly consider every possible ordering rule for such a combinatorially-sized set, even for a single task/dataset.

Given that exhaustively considering all orderings is infeasible, we instead cast the problem of selecting a word order for a sentence as a combinatorial optimization problem. Consequently, we define an ordering rule r_f, parameterized by an objective function f, as follows:

$$r_f(\bar{s}_i) = \arg\min_{\pi \in S_n} f(\pi, \bar{s}_i) \tag{4.5}$$

for a cost function f. In §6.3, we revisit how we may handle the case when such an optimization is ill-posed/there exist multiple solutions.

[3]For simplicity, we assume every sentence in the dataset is length n in this section.
[4]Recall that we have assumed there are no duplicate words within any sentence.

the horse raced past the barn fell

Figure 4.1: A garden path construction with a long-distance dependency linking *horse* and *fell*.

4.2.1 Bandwidth

In the previous chapter, we presented several accounts that suggest that humans have fundamental limitations on their abilities to process long-distance dependencies. In general, long-distance dependencies can be a substantial complication in maintaining incremental parses of a sentence. As an illustration, consider the example given in Figure 4.1 as a particularly pathological case. Here, the long-distance dependency between *horse* and *fell* may contribute to the confusion in parsing this sentence for many readers on an initial pass.

In computational models, we have also seen treatments that restrict the ability to model arbitrarily long dependencies. Most saliently, in models with Markovian assumptions, such as the HMMs described in §2.3.2, there is a fundamental constraint that prohibits modelling dependencies of length greater than the Markov order. Similarly, in Transformer models when the stride size is the context window length, dependencies of length greater than the context window length can simply not be modelled.

Given the difficulty of handling long-distance dependencies in both the human language processing and computational language processing settings, we

can consider an ordering rule which ensures that the longest dependency in every re-ordered sentence is as short as possible. As such, we define the BANDWIDTH cost function as follows:

$$\text{BANDWIDTH}(\pi, \bar{s}) = \max_{(w_i, w_j) \in \mathcal{E}} d_\pi(w_i, w_j) \tag{4.6}$$

Consequently, this defines the ordering rule r_b under the framework given in Equation 4.5.

$$r_b(\bar{s}) = \arg\min_{\pi \in S_n} \text{BANDWIDTH}(\pi, \bar{s}) \tag{4.7}$$

The term *bandwidth* refers to the fact that the optimization problem given in Equation 4.7 is known in the combinatorial optimization literature as the BANDWIDTH problem. The problem was introduced in 1967 by Harary (1967) for graphs, though it has been posed previously for matrices in the mid 1950s. The matrix formulation is equivalent, as a graph can be viewed as its adjacency matrix and the bandwidth of a matrix is exactly the bandwidth of a graph as it measures the maximal distance from the main diagonal of any non-zero elements.

The bandwidth problem for matrices has a rich history that has been especially prompted by its applications in numerical analysis. Specifically, numerical computations can be improved (generally by reduction of space costs and) for matrices with low bandwidth in several matrix factorization (e.g. Cholesky) and matrix multiplication schemes. As a result, bandwidth reduction has been integrated

to various numerical analysis software (Fellows and Langston, 1994) and some libraries include algorithms for matrices with small bandwidth (Saad, 2003). In other contexts, bandwidth reduction has also been combined with other methods for various high-volume information retrieval problems (Botafogo, 1993).

Beyond its applied value, the problem has been also the subject of deep theoretical inquiry. Extensive surveys have been written on the problem (Chinn et al., 1982) as well as the corresponding complexity results (Garey et al., 1978). In particular, Papadimitriou (1976) demonstrated the problem was NP-Hard for general graphs.

4.2.2 Minimum Linear Arrangement

In general, using BANDWIDTH as a cost function to induce an ordering rule implies that there are many improvements that are (potentially) missed. For example, a linear layout for a graph with two edges, where the edge lengths are 6 and 5 achieves equivalent bandwidth as the the linear layout where the edge lengths are 6 and 1. In this sense, the BANDWIDTH objective may be at odds with the realities of language processing as both humans and computers must model every dependency and not just the longest one.

Once again, we turn to the prior work on human language processing for inspiration. In particular, we have seen in §3.4 that the literature on dependency length

minimization has suggested an alternative processing cost. In particular, the works of Gibson (1998, 2000) describe a cost function as given in Equation 4.8. This is the exact cost function used in work demonstrating large-scale evidence of dependency length minimization by Futrell et al. (2015).

$$\textsc{minla}(\pi, \bar{s}) = \sum_{(w_i, w_j) \in \mathcal{E}} d_\pi(w_i, w_j) \tag{4.8}$$

As we have seen before, this allows us to define the associated ordering rule r_m under the framework given in Equation 4.5.

$$r_m(\bar{s}) = \arg\min_{\pi \in S_n} \textsc{minla}(\pi, \bar{s}) \tag{4.9}$$

Reminiscent of the BANDWIDTH problem, we refer to this objective as the MINLA objective as a shorthand that refers to the MINIMUM LINEAR ARRANGEMENT problem in the algorithms literature.[5] Introduced by Harper (1964), the problem has been referred by various names including *optimal linear ordering* or *edge sum* and is sometimes conflated with its edge-weighted analogue. Harper considered the problem originally in the context of generating effective error-correcting codes (Harper, 1964, 1966).

The problem has arisen in a number of different applications. In particular, in

[5]While the objective in Equation 4.8 has also been studied in the psycholinguistics literature under the name of *dependency length*, we choose to use the more general algorithmic jargon. In particular, this helps to disambiguate this objective from others we have seen (such as BANDWIDTH).

wiring problems for circuit design (e.g. VLSI layout problems), reductions to the minimum linear arrangement problem are frequent (Adolphson and Hu, 1973). Similarly, the problem has been often used for job scheduling (Correa and Schulz, 2004; Ravi et al., 1991). The problem shares important theoretical connections with the *crossing number*, which emerges in aesthetic graph drawing problems (Pach et al., 1996). As we have seen, the problem and objective are also studied in more abstract settings, removed from the pure combinatorial optimization paradigm, including in the dependency minimization literature and for rudimentary models of neural behavior (Mitchison and Durbin, 1986).

Similar to the bandwidth problem, the problem has seen a number of theoretical techniques applied to it. This has led to a number of improvements in complexity results for the problem[6] for restricted graph families but the general problem is NP-Hard (Garey et al., 1974). Petit and Salgado (1998) have benchmarked the problem in several settings (along with providing approximation heuristics) and Liu and Vannelli (1995) have given general arguments for arriving at lower bounds on the problems.

Relating BANDWIDTH and MINLA. The BANDWIDTH and MINIMUM LINEAR ARRANGEMENT cost functions (Equation 4.6 and Equation 4.8) are related in that both operate over edge lengths with one invoking a max where the other invokes a sum. In this

[6] We consider this in §4.3.

sense, this is highly reminiscent of the relationship shared by p norms for $p = 1$ and $p = \infty$. More generally, we can define a family of ordering rules r_p parameterized by input $p \in \mathbb{N} \cup \infty$ as follows:

$$r_p(\pi, \bar{s}) = \arg\min_{\pi \in S_n} \left(\sum_{(w_i, w_j) \in \mathcal{E}} d_\pi(w_i, w_j)^p \right)^{1/p} \tag{4.10}$$

In particular, setting $p = 1$ recovers the ordering rule r_m for MINLA as in Equation 4.9 and setting $p = \infty$ recovers the ordering rule r_b for BANDWIDTH as in Equation 4.7.

4.2.3 Cutwidth

In introducing the BANDWIDTH and MINLA objectives, the motivation was that processing long-distance dependencies was challenging for both humans and machines. With that in mind, the length of the dependencies are not the sole property that may correlate with the complexity of processing (and therefore motivate re-ordering to facilitate processing). As a complement to the length of dependencies, humans also have limits to their processing capabilities regarding memory capacity. In this sense, having many dependencies simultaneously active/uncompleted may also correlate with cognitive load. This phenomenon has been shown for humans across a variety of fronts, perhaps most famously in the experiments of Miller (1956). Miller demonstrated that humans may have fundamental hard constraints on the number of objects they can simultaneously track in their working short-term mem-

ories.[7] Given that similar challenges have been found in computational language processing, memory mechanisms and attentional architectures have been proposed to circumvent this issue. Rather than introducing computational expressiveness, we next consider how to devise orders that explicitly minimize quantities related with the number of active dependencies.

In order to the track the number of active dependencies, we introduced the notion of the *edge cut* previously, which describes the number of dependencies that begin at or before position i under π but have yet to be completed (the other vertex of the edge appears after position i under π). Consequently, we define the CUTWIDTH cost:

$$\textsc{cutwidth}(\pi, \bar{s}) = \max_{w_i \in \mathcal{V}} \theta_\pi(i) \tag{4.11}$$

As we have seen before, this allows us to define the associated ordering rule r_c under the framework given in Equation 4.5.

$$r_c(\bar{s}) = \arg\min_{\pi \in S_n} \textsc{cutwidth}(\pi, \bar{s}) \tag{4.12}$$

Akin to the previous two settings, the CUTWIDTH problem is also a problem in the combinatorial optimization literature pertaining to linear layouts. The problem emerged in the 1970's due to Adolphson and Hu (1973) and in the late 80's from Makedon and Sudborough (1989) as a formalism for circuit layout problems. In

[7] Canonically, Miller claimed this was 7 ± 2.

particular, the cutwidth of a graph scales in the area needed for representing linear VLSI circuit layouts.

As with the previous problems, the CUTWIDTH problem has continued to arise in several other applications. Botafogo (1993) used the CUTWIDTH problem alongside the BANDWIDTH problem for information retrieval, Karger (2001) studied the problem for designing a *PTAS* for network reliability problems, and Mutzel (1995) considered the problem in automated graph drawing.

Somewhat unlike the previous two problems, the problem has seen less theoretical interest despite its numerous applications. Nonetheless, the problem was shown to be NP-Hard by Gavril (2011), which continues the trend seen for the other combinatorial optimization problems we consider.

Linking capacity and dependency-length. Previously, we introduced the MINLA cost function as arguably rectifying an issue with BANDWIDTH cost function. In particular, the BANDWIDTH cost function does not explicitly model the aggregate cost which is ultimately perceived in language processing. Both humans and machine must model and process all dependencies in a sequence in order to fully understand the sentential meaning. A similar inadequacy could be posited regarding the CUTWIDTH

optimization problem and objective. Consequently, we define SUM-CUTWIDTH as:

$$\text{SUM-CUTWIDTH}(\pi, \bar{s}) = \sum_{w_i \in \mathcal{V}} \theta_\pi(i) \tag{4.13}$$

As we have seen before, this allows us to define the associated ordering rule $r_{m'}$ under the framework given in Equation 4.5.

$$r_{m'}(\bar{s}) = \arg\min_{\pi \in S_n} \text{SUM-CUTWIDTH}(\pi, \bar{s}) \tag{4.14}$$

As the naming convention for $r_{m'}$ suggests, we note that we have already encountered $r_{m'}$ and SUM-CUTWIDTH previously.

Theorem 4.1 (Equivalence of average edge cut and average dependency length)**.** *SUM-CUTWIDTH = MINLA*

Proof. For every edge $(w_i, w_j) \in \mathcal{E}$, the edge contributes its length $d_\pi(w_i, w_j)$ to the MINLA cost. On the other hand, edge (w_i, w_j) contributes 1 to the edge cut θ_k for $k \in \big[\pi(w_i), \pi(w_j)\big)$.[8] Therefore, in aggregate, edge (w_i, w_j) contributes exactly $\Big|\big[\pi(w_i), \pi(w_j)\big)\Big| = d_\pi(w_i, w_j)$ to the CUTWIDTH cost. As this holds for every edge, it follows that SUM-CUTWIDTH = MINLA. □

Corollary 4.1.1. $r_m = r_{m'}$ up to the uniqueness of the solution of the combinatorial optimization problem.

[8] *WLOG* assume that $\pi(w_i) < \pi(w_j)$, the notation $[a, b)$ indicates $\{a, a+1, \ldots, b-1\}$ for integral a, b.

To the knowledge of the authors, the following argument has not been considered in the literature on modelling language processing with relation to costs pertaining to dependency length/capacity. From this perspective, the result affords an interesting reinterpretation that dependency length minimization is equivalent to minimizing the number of active dependencies. In this sense, it may suggest that related findings (such as those of Miller (1956)) may be more pertinent and that the relationship between dependency length and memory capacity may be much more direct than previously believed.

4.3 Algorithms for Combinatorial Optimization

In the previous section (§4.2), we introduced objectives and corresponding ordering rules that are motivated by challenges in both computational and human language processing. As a recap of the section, we consider Figure 4.2, which depicts a graph and the evaluation of the three cost functions on the graph (given the permutation depicted using vertex labels). Further, in Figure 4.3, we observe that solutions to the problem of finding the re-ordering that minimizes each of the three objectives can be different. Note that, in this specific case, the minimum linear arrangement-optimal solution is optimal for the other objectives and the cutwidth solution is optimal for the bandwidth objective but not for the minimum linear arrangement objective.

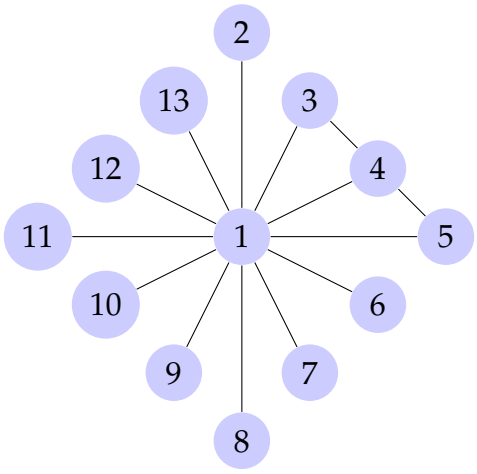

Figure 4.2: A graph $\mathcal{G}$ with a linear layout specified by the vertex labels in the figure. Given this linear layout, the bandwidth is 12 (this is $13-1$), the cutwidth is 12 (this is due to position 1), and the minimum linear arrangement score is 80 $\left(\text{this is } \sum_{i=2}^{13}(i-1)+(4-3)+(5-4)\right)$.

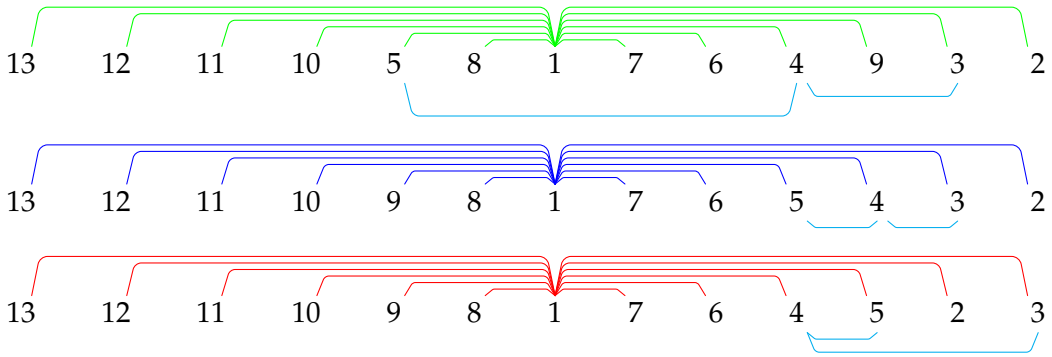

Figure 4.3: Solutions for optimizing each of the three objectives for the graph given in Figure 4.2. The linear layout is conveyed via the linear ordering and the numbers refer to the original vertices in the graph (as shown in Figure 4.2). The top/green graph is bandwidth-optimal (bandwidth of 6), the middle/blue graph is minimum linear arrangement-optimal (minimum linear arrangement score of 44), the bottom/red graph cutwidth-optimal (cutwidth of 6). The cyan edges drawn below the linear sequence convey the difference in the optimal solutions.

In order to make use of these ordering rules for natural language processing applications, it is necessary to tractable solve each of the associated optimization problems. Recall that each of these problems is NP-Hard for general graphs. In spite of this roadblock, also recall that we are considering applying this optimization to sentences equipped with dependency parses, where dependency parses are graphs that are guaranteed/constrained to be trees.

Bandwidth. Unfortunately, the BANDWIDTH problem remains NP-Hard for trees as well (Garey et al., 1978). In fact, the problem is well-known for remaining NP-Hard for a number of graph relaxations (Monien, 1986; Díaz et al., 1999) including fairly simple graphs like caterpillar with hair-length at most 3 (Monien, 1986). Given this, one natural approach to find tractable algorithms is to consider provable approximations. However, approximations to a factor of 1.5 do not even exist for both general graphs and trees (Blache et al., 1997).[9] Regardless, approximation guarantees are generally unhelpful as we are considering sentences-scale graphs and therefore *small* graphs, where the approximation factor may be quite significant. Instead, in this thesis, we consider heuristics for the BANDWIDTH problem. In particular, we make use of the frequently-employed heuristic of Cuthill and McKee (1969). We discuss this algorithm below and defer the implementation details to a subsequent chapter.

[9]This further implies that the BANDWIDTH problem does not admit a *PTAS*.

The Cuthill-McKee algorithm begins by starting at the vertex and conducting a breadth-first search (BFS) from that vertex. The key to the algorithm's empirical effectiveness is that vertices are visited based on their degree (hence the starting vertex is the vertex with lowest degree). Instead of using the standard algorithm, we use the Reverse Cuthill-McKee algorithm (Chan and George, 1980), which merely executes the algorithm with reversed index numbers. In theory, this modification has no benefits for general graphs but empirically, it seems this modification turns out to be reliably beneficial (Davis, 2006; Azad et al., 2017).

Minimum Linear Arrangement. Unlike the BANDWIDTH problem, the MINLA problem has poly-time solutions for trees. In particular, in a series of results, the runtime for the tree setting was improved from $\mathcal{O}(n^3)$ (Goldberg and Klipker, 1976) to $\mathcal{O}(n^{2.2})$ (Shiloach, 1979) to $\mathcal{O}(n^{1.58})$ (Chung, 1984). While there has been progress on developing lower bounds (Liu and Vannelli, 1995), matching bounds have been yet to be achieved. In this work, we elect not to use the algorithm of Chung (1984) and instead introduce an additional constraint and a corresponding algorithm in a subsequent section (§4.3.1).

Cutwidth. Similar to MINLA, CUTWIDTH also permits poly-time solutions for trees. While the problem remained open as to whether this was possible for a number of years, Yannakakis (1985) gave a $\mathcal{O}(n \log(n))$ algorithm. Further akin to MINLA, we forego this general algorithm (for trees) for one that involves projectivity constraints

(§4.3.1).

4.3.1 Projectivity Constraints

Recall that poly-time algorithms exist for both the MINLA and CUTWIDTH problems. In both cases, the works introducing the corresponding algorithms we discussed previously develop significant algorithmic machinery to arrive at the final algorithms. We will next see that a linguistically-motivated constraint in *projectivity* yields faster[10] and simpler algorithms. Recall that projectivity refers to the property of a dependency parse that when the nodes are ordered on a line and the edges are drawn above the line, the parse has no intersecting edges. If we constraint the order π outputted by either r_m or r_c to be projective, linear time algorithms are known for MINLA (Gildea and Temperley, 2007) and CUTWIDTH (Yannakakis, 1985).[11] We next discuss the algorithms of Gildea and Temperley (2007) and Yannakakis (1985), showing how they both can be generalized using the framework of *disjoint strategies* (Yannakakis, 1985).

Dynamic Programming for tree-based optimization. Since we are considering both the CUTWIDTH and minLA problems in the setting of trees, dynamic program-

[10] From a practical perspective, the algorithms given in §4.3 are likely sufficiently fast for almost any language data. In particular, sentences are generally short from a complexity perspective and hence algorithms with asymptotic complexity of $\mathcal{O}(n^{1.58})$ and $\mathcal{O}(n \log n)$ are unlikely to be of concerning cost, especially since the methods we study in §5 are one-time costs.

[11] In some algorithmic contexts, the problem of returning a linear layout that is constrained to be projective is known as the PLANAR version of a problem, e.g. PLANAR-CUTWIDTH.

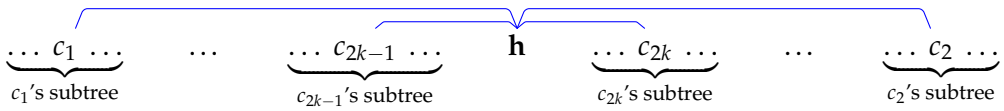

Figure 4.4: Illustration of the disjoint strategy. The root h is denoted in **bold** and it has $2k$ children denoted by $c_1, \ldots, c_{2k}$. Its children and their subtrees are organized on either side. The order within each child subtree is specified by a linear layouts that has previously been computed in the dynamic program. The order of the children and their subtrees alternates and moving from outside to inside based on their score according to some scoring function. Hence, the subtree rooted at child c_1 receives the highest score and the subtree roots at child c_{2k} receives the lowest score. If the root h had $2k+1$ (an odd number) of children, the strategy is followed for the first $2k$ and we discuss the placement of the last child subsequently.

Algorithm 1: `Disjoint Strategy`

Input: A tree rooted at h with children $c_1, \ldots, c_{2k}$.
Input: Optimal linear layouts $\pi_1, \ldots, \pi_{2k}$ previously computed in the dynamic program. π_i is the optimal linear layout for the tree rooted at c_i.
$\pi_h \leftarrow \{h : 1\}$
ranked-children $\leftarrow$ `sort`$\left([1, 2, \ldots, 2k], \lambda x.\texttt{score}\,(c_x) \right)$
$\pi \leftarrow \left(\bigoplus_{i=1}^{k} \pi_{\text{ranked-children}[2i-1]} \right) \oplus \pi_h \oplus \left(\bigoplus_{i=0}^{k-1} \pi_{\text{ranked-children}[2(k-i)]} \right)$
return π

ming approaches are well-motivated. In particular, we will consider how optimal linear layouts for each of the children at a given node can be integrated to yield the optimal linear layout at the given node. As we will show, both algorithms we consider can be thought of as specific instantiations of the abstract *disjoint strategy* introduced by Yannakakis (1985).[12] In Figure 4.4, we depict what the disjoint strategy looks like and in Algorithm 1, we provide the template for the disjoint strategy algorithm. We denote linear layouts programmatically as dictionaries, hence π_h is the function $\pi_h : \{h\} \to \{1\}$ given by $\pi_h(h) = 1$. We define the $\oplus$ operator over

[12] **Remark:** Gildea and Temperley (2007) develop the same general framework in their own work. Since both algorithms share a similar template, we prefer the standardized template to their *ad hoc* description.

linear layouts below.

Definition 4.5. $\oplus$ — A binary operator for arbitrary parameters n, m of type $\oplus$: $S_n \times S_m \to S_{n+m}$ specified by:

$$\oplus(\pi_x, \pi_y)(w_i) = \begin{cases} \pi_x(w_i) & w_i \in Dom(\pi_x) \\ \pi_y(w_i) + n & w_i \in Dom(\pi_y) \end{cases} \tag{4.15}$$

$\bigoplus$ is given by the repeated application of $\oplus$ (analogous to the relationship between $+$ and $\sum$ or $\cup$ and $\bigcup$).

Minimum Linear Arrangement. The function `score` in Algorithm 1 is defined such that $\texttt{score}(c_i)$ is the size of the subtree rooted at c_i.

Cutwidth. The function `score` in Algorithm 1 is defined such that $\texttt{score}(c_i)$ is the modified cutwidth of the subtree rooted at c_i under π_i. The modified cutwidth of a tree under π is the cutwidth of the tree under π plus a bit indicating whether that are positions on either side of the root of the tree at which the cutwidth (the maximum edge cut) is obtained. Hence, the modified cutwidth is the cutwidth if all positions at which the edge cut is maximized occur strictly on the left of the root XOR if all positions at which the edge cut is maximized occur strictly on the right of the root. Otherwise, the modified cutwidth is the cutwidth plus one.

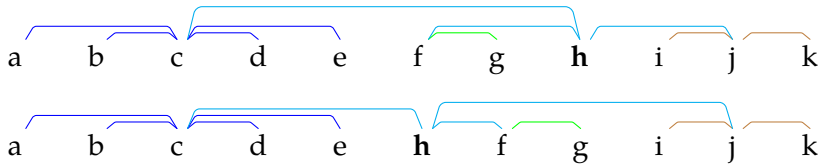

Figure 4.5: Linear layouts exemplifying the difference between the solutions produced by the Gildea and Temperley (2007) algorithm (top) and our algorithm (bottom). The root h is denoted in **bold**. In both algorithms, the linear layouts for the children with the largest subtrees — the blue subtree rooted at c and the brown subtree rooted at j — are placed on opposite sides. The difference is the placement of the green subtree rooted at child f. The arcs whose edge lengths change across the two layouts are those in cyan, notably (c, h), (f, h), and (j, h). However, the sum of the edge lengths for (c, h) and (j, h) is constant across the linear layouts. Hence, the difference in minimum linear arrangement scores between the linear layouts is solely dictated by the length of (f, h), which is shorter in our algorithm's layout (bottom layout).

Proofs of correctness. Intuitively, both proofs of correctness hinge on the fact that the disjoint strategy involves placing 'larger' subtrees in an alternating outside-to-inside fashion around the current node/root. By selecting the correct measure of 'large', the 'adverse' effects of the large subtrees affect as few other subtrees as possible (since they are outside the link from these subtrees to the root, which is the only way in which the costs interact across subtrees/with the root). For readers interested in the fully formal proof of correctness, we direct them to the corresponding works: Yannakakis (1985) and Gildea and Temperley (2007).

Correcting the algorithm of Gildea and Temperley (2007). We note that one small correction is made in our algorithm that was not originally addressed by Gildea and Temperley (2007). In particular, consider the case when the given node has an odd number of children (i.e. $2k + 1$ children for some non-negative integer k). In this case, the $2k$ largest children and their associated subtrees are alternated as dictated by the *disjoint strategy*. Gildea and Temperley (2007) claim that the

placement of the final child does not matter, however this is incorrect. We show this in Figure 4.5. That said, it is fairly simple to correct this error. The final child's subtree is *left-leaning* if more of the child's subtree is oriented to its left than right (according to the linear layout already computed), *right-leaning* if more of the child's subtree oriented to its right than left, and *centered* otherwise. If the child's subtree is leaning in either direction, it should be placed on that side of the root (closest to the root relative to the root's other children, i.e. in accordance with the *disjoint strategy*). Otherwise, the side it is placed on does not matter.

The proof of correctness that this globally improves the outputted linear layout is fairly straightforward. In particular, since there are k children of the root on either side, the objective will be incremented by $k *$ the size of the $2k + 1$ child's subtree in irrespective of this decision (where to place child $2k + 1$ and its subtree). All arcs above the root and within any of the root's children's subtrees will not change length. Hence the only arc of interest is the one connecting child $2k + 1$ and the root. In this case, clearly placing the root on the side opposite of the way the child's subtree is leaning will yield a smaller such arc (as depicted in Figure 4.5).

As a concluding remark, we note that the error we presented with the algorithm of Gildea and Temperley (2007) is exceptionally pervasive. In particular, for every child with an odd number of children in the tree, the algorithm they present may have contributed to a misplacement with probability $\frac{1}{2}$. When evaluated over data

we describe in §5.3, we find over 95% of the sentences contain such an instance. Additionally, the runtime analysis of Gildea and Temperley (2007) indicates that their algorithm is $\mathcal{O}(n)$. Given the sorting required, this is not naively true but can be rectified using appropriate data structures and bucket sort as suggested by Yannakakis (1985).

4.4 Heuristics for Mixed-Objective Optimization

In the previous section, we considered an algorithm for each of the BANDWIDTH, MINLA, and CUTWIDTH problems. In all cases, the algorithm had the goal of a producing that a linear layout that minimized the corresponding objective to the extent possible.[13] However, in our setting, we are considering re-ordering the words of a sentence for modelling sentences. From this perspective, there is a clear risk that re-ordering the words to optimize some objective regarding dependencies may obscure other types of order-dependent information in the original sentence. We next consider a template that specifies a simple heuristic for each of these optimization problems. In particular, the heuristic involves a parameter T which bears some relationship with the notion of trading off the extent to which the original sentence's word order is distorted with the extent to which the combinatorial objective is minimized.

[13] Perhaps with additional constraints such as projectivity.

4.4.1 Transposition Monte Carlo

Algorithm 2: `Transposition Monte Carlo`

Input: A sentence $\bar{s}$ and its dependency parse $\mathcal{G}_{\bar{s}} = (\mathcal{V}, \mathcal{E})$.
Initialize $\pi = \pi_I$
Initialize $c = \texttt{OBJ}(\pi, \bar{s})$
for $t \leftarrow 1, \ldots, T$ **do**
    $w_i, w_j \sim \mathcal{U}_{\mathcal{V}}$
    $\pi_{\text{temp}} \leftarrow \pi$
    $\pi_{\text{temp}}(w_i), \pi_{\text{temp}}(w_j) \leftarrow \pi_{\text{temp}}(w_j), \pi_{\text{temp}}(w_i)$
    $c_{\text{temp}} \leftarrow \texttt{OBJ}(\pi_{\text{temp}}, \bar{s})$
    if $c > c_{temp}$ **then**
        $\pi \leftarrow \pi_{\text{temp}}$
        $c \leftarrow c_{\text{temp}}$
end
return π

In Algorithm 2, we present the template we consider for our three heuristic algorithms. Intuitively, the algorithm is a Monte Carlo algorithm that at each time step randomly samples a transposition[14] and considers altering the current permutation π according to this transposition. For this reason, we call the algorithm the `Transposition Monte Carlo` algorithm.

We refer to this algorithm as a heuristic since, like any Monte Carlo algorithm, there is no provable guarantee that the algorithm produces the optimal solution. In particular, we note that the algorithm greedily decides whether to update in accordance with a candidate transposition and hence makes *locally optimal* decisions. Consequently, there is no guarantee that this procedure will lead to the *globally*

[14]The notation permits the $w_i = w_j$ but there is no benefit to this, so the notation should be taken as sampling w_i and then sampling w_j without replacement from the resulting distribution.

optimal linear layout. However, unlike the algorithms in the section on algorithms for producing linear layouts under projectivity constraints (§4.3.1), the permutation produced by this algorithm need not be projective. We will revisit this in empirically considering the quality of the optimization on natural language data (§5.3).

The `Transposition Monte Carlo` algorithm is parameterized by two quantities: the objective/cost function being minimized `OBJ` and the stopping criterion/threshold T.

1. `OBJ` — In this work, we specify cost functions `OBJ` in accordance with those which we have seen previously — the BANDWIDTH cost in Equation 4.6, the MINLA cost in Equation 4.8, and the CUTWIDTH cost in Equation 4.11. We will refer to the associated permutations as $\tilde{\pi}_b$, $\tilde{\pi}_m$, and $\tilde{\pi}_c$ respectively and will similarly denote the induced ordering rules as $\tilde{r}_b$, $\tilde{r}_m$, and $\tilde{r}_c$.
2. T — In this work, we fix $T = 1000$ which approximately corresponds to `Transposition Monte Carlo` taking the same amount of wall-clock time as it takes to run any of the algorithms in §4.3 on the same data. However, varying T allows for the possibility of flexibility controlling the extent to which the returned linear layout is distorted. For $T = 0$, we recover the ordering rule r_I which returns π_I for a given input sentence. For $T = \infty$, we are not guaranteed to find a global optima to the author's knowledge (due to the local greediness of the algorithm), but we are guaranteed to find a local optima (in the sense that no transposition from the solution yields

a better solution). Treating T as a task-dependent hyperparameter in NLP applications and optimizing for it (perhaps on validation data) may encode the extent to which dependency locality is important for the given task.

CHAPTER 5

OPTIMAL LINEAR ORDERS FOR NATURAL LANGUAGE PROCESSING

In this chapter, we describe how to integrate the novel word orders we have considered with models in natural language processing. We specifically introduce the `pretrain-permute-finetune` framework which fully operationalizes this. We then conduct an empirical evaluation of our method using the word orders we have introduced previously.

5.1 Motivation

In §2.3, we discuss several approaches to addressing order, and word order specifically, in computational models of language. The dominant recent paradigm has been to introduce computational expressiveness (e.g. LSTMs in place of simple RNNs, attention, syntax-augmented neural methods (Socher et al., 2013a; Bowman et al., 2015; Dyer et al., 2016; Dyer, 2017; Kim et al., 2019)) as a mechanism for handling the difficulties of long-distance dependencies and word order more generally. Alternatively, position-aware Transformers have been considered but their widespread effectiveness hinges on large amounts of data with sufficient hardware resources to exploit the increased ability for parallelism. These methods also tend to be significantly more challenge to optimize in practice (Liu et al., 2020). Ultimately, it remains unclear whether it is necessary to rely on additional model complexity or whether restructuring the problem (and the input specifically) may

be more appropriate. To this end, we consider the orders which we have introduced in §4 as a mechanism for codifying new views to the problem of sequential modelling language. Next, we explore the potential newfound advantages (and costs/limitations) of using these novel word orders in computational models of language.

5.2 Methods

We began by considering training models where, for every sentence in the training data, we instead use its re-ordered analogue. In initial experiments, we found the results to be surprisingly promising. However, we found that this paradigm may not be particularly interesting given there are very few, if any, tasks in NLP that are currently conducted using representations built upon strictly task-specific data. Instead, almost all NLP models leverage pretrained (trained on large amounts of unlabelled data in a downstream task-agnostic fashion) representations to serve as potent initializations (Ruder, 2019). The past few years have seen the introduction of pretrained *contextualized representations*, which are functions $c : \bigcup_{i=1}^{\infty} \mathcal{V}^i \rightarrow \bigcup_{i=1}^{\infty} \left(\mathbb{R}^d\right)^i$, that map word sequences (e.g. sentences) to vector sequences of equal length. In particular, the resulting vectors encode the contextual meaning of the input words. Initial pretrained contextualized representations include: CoVe (McCann et al., 2017), ELMo (Peters et al., 2018b), and GPT (Radford et al., 2018); the current best[1]

[1]It is not sufficiently specified to rank pretrained representations in a task-agnostic fashion given that they may have disparate and inconsistent performance across different downstream

pretrained contextualized representations include: BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), XLNet (Yang et al., 2019), RoBERTa (Liu et al., 2019). SpanBERT (Joshi et al., 2020a), ELECTRA (Clark et al., 2020), ALBERT (Lan et al., 2020) and T5 (Raffel et al., 2019).

Given the widespread use of transfer learning and pretraining methods in NLP (Ruder, 2019), we begin by describe the `pretrain-and-finetune` framework that has featured prominently across NLP. In particular, the mapping from an input sentence $\bar{s}$ to the predicted output $\hat{y}$ is given by the following process:

1. Tokenize the input sentence $\bar{s}$ into words $\langle w_1 \dots w_n \rangle$.
2. Embed the sentence $\langle w_1 \dots w_n \rangle$ using the pretrained encoder/contextualized model c to yield vectors $\vec{x}_1, \dots, \vec{x}_n$ where $\forall i \in [n], \vec{x}_i \in \mathbb{R}^d$.
3. Pass $\vec{x}_1, \dots, \vec{x}_n$ into a randomly initialized component F that is trained on the task of interest that outputs the prediction $\hat{y}$.

In order to modify this process to integrate our novel word orders, one straightforward approach would be simply feeding the permuted sequence of inputs into the pretrained encoder. Perhaps unsurprisingly, we found this to perform quite poorly in initial experiments. After all, the pretrained encoder never observed

tasks/datasets. In labelling some of these representations as the 'best', we refer to the `GLUE` (Wang et al., 2019b) and `SuperGLUE` (Wang et al., 2019a) benchmarks for natural language understanding and similar benchmarks for natural language generation (c.f Raffel et al., 2019).

such permuted constructions during training and these inputs are effectively out-of-distribution for the model. Another natural approach that is more reasonable would be to re-pretrain the encoder using the same training data but with all sentences permuted according to the order being studied. Unfortunately, this is well beyond our computational constraints and likely is not practical for almost all research groups given the tremendous time, cost, and unique GPU and TPU resources required to pretrain modern models. Even for such entities who can bear these expenses, this is unlikely to be feasible if multiple orders are to be experimented with and there are substantial ethical considerations given the sizeable environmental impact (Strubell et al., 2019).

Given these roadblocks, we propose permuting the vectors $\vec{x}_1, \ldots, \vec{x}_n$ in between steps 2 and 3. In doing so, we seamlessly integrate our permuted orders in a model-agnostic and task-agnostic fashion while preserving the benefits of pretraining. Further, since the permutation for a given example can be pre-computed (and is a one-time cost), the additional runtime cost of our method during both training and inference is simply the cost of computing the optimal orders over the dataset.[2] We name our framework `pretrain-permute-finetune` and we explicitly state the procedure below.

[2] As a reference, for the 5 datasets we study (and over 40000 examples), optimization across all six orders takes less than 3 hours on a single CPU. Several aspects of this are embarrassingly parallel and others can be more cleverly designed using vector/matrix operations in place of dynamic programming and for loops. This suggests that highly parallel GPU implementations can render this runtime cost to be near-zero. Regardless, it is already dramatically dwarfed by the costs of training.

1. Tokenize the input sentence $\bar{s}$ into words $\langle w_1 \dots w_n \rangle$.
2. Embed the sentence $\langle w_1 \dots w_n \rangle$ using the pretrained encoder/contextualized model c to yield vectors $\vec{x}_1, \dots, \vec{x}_n$ where $\forall i \in [n], \vec{x}_i \in \mathbb{R}^d$.
3. Permute the vectors $\vec{x}_1, \dots, \vec{x}_n$ according to linear layout π. In other words, $\forall i \in [n], \vec{z}_{\pi(w_i)} \triangleq \vec{x}_i$.
4. Pass $\vec{z}_1, \dots, \vec{z}_n$ into a randomly initialized component F that is trained on the task of interest that outputs the prediction $\hat{y}$.

Pretrained Contextualized Representations. In this thesis, we use a pretrained ELMo (Peters et al., 2018b) encoder to specify c. ELMo is a pretrained shallowly-bidirectional language model that was pretrained on 30 million sentences, or roughly one billion words, using the 1B Word Benchmark (Chelba et al., 2013). The input is first tokenized and then each word is deconstructed into its corresponding character sequences. Each character is embedded independently and then a convolutional neural network is used to encode the sequence and produce word representations. The resulting word representations are passed through a two-layer left-to-right LSTM and a two-layer right-to-left LSTM. Two representations are produced for every word w_i. The first is $\texttt{ELMo}_1(w_i \mid \langle w_1 \dots w_n \rangle) \in \mathbb{R}^d$, which is the concatenated hidden states from the first layer of both LSTMs. Similarly, the second is $\texttt{ELMo}_2(w_i \mid \langle w_1 \dots w_n \rangle) \in \mathbb{R}^d$, which is the concatenated hidden states from the second layer of both LSTMs. In our work, $\vec{x}_i \triangleq [\texttt{ELMo}_1(w_i \mid \langle w_1 \dots w_n \rangle); \texttt{ELMo}_2(w_i \mid \langle w_1 \dots w_n \rangle)] \in \mathbb{R}^{2d}$.[3]

[3]; denotes concatenation

Task-specific Component. We decompose the task-specific model F into a bidirectional LSTM, a max pooling layer, and a linear classifier:

$$\overleftarrow{h_{1:n}}, \overrightarrow{h_{1:n}} \leftarrow \texttt{BidiLSTM}(\vec{z}_1, \ldots, \vec{z}_n) \tag{5.1}$$

$$\vec{h} \leftarrow \left[\max\left(\overleftarrow{h_{1:n}}\right) \; ; \; \max\left(\overrightarrow{h_{1:n}}\right)\right] \tag{5.2}$$

$$\hat{y} \leftarrow \texttt{Softmax}\left(\mathbf{W}\vec{h} + \vec{b}\right) \tag{5.3}$$

where `BidiLSTM`, $\mathbf{W}, \vec{b}$ are all learnable parameters and $\max(\cdot)$ is the element-wise max operation.

5.3 Data

In this work, we evaluate our methods and baselines on five single-sentence text classification datasets. Summary statistics regarding the distributional properties of these datasets are reported in Table 5.1. We use official dataset splits[4] when available and otherwise split the dataset randomly into 80% training data, 10% validation data, and 10% test data.[5] We also report the fraction of examples that the `spaCy` parser generates an invalid parse. For examples with invalid parse, since

[4]For datasets where the standard split is only into two partitions, we further partition the training set into $\frac{8}{9}$ training data and $\frac{1}{9}$ validation data.

[5]When we create data splits, we elect to ensure that the distribution over labels in each dataset partition is equal across partitions.

	Train	Validation	Test	$\frac{\text{Words}}{\text{ex.}}$	Unique Words	Classes	Fail %
CR	3016	377	378	20	5098	2	19.2
SUBJ	8000	1000	1000	25	20088	2	13.6
SST-2	6151	768	1821	19	13959	2	6.7
SST-5	7594	949	2210	19	15476	5	6.8
TREC	4846	605	500	10	9342	6	1.0

Table 5.1: Summary statistics for text classification datasets. Train, validation, and test refer to the number of examples in the corresponding dataset partition. $\frac{\text{Words}}{\text{ex.}}$ refers to the average number of words in the input sentence for each example in the union of the training data and the validation data. Unique words is the number of unique words in the union of the training data and the validation data. Classes is the size of the label space for the task. Fail % is the percentage of examples in the union of the training and validation set where the spaCy dependency parser emits an invalid parse (e.g multiple syntactic roots, not a connected graph).

we cannot meaningfully compute optimal linear layouts, we back-off to using the identity linear layout π_I. All data is publicly available and means for accessing the data are described in §A.3.

Customer Reviews Sentiment Analysis. This dataset was introduced by Hu and Liu (2004) as a collection of web-scraped customer reviews of products. The task is to predict whether a given review is positive or negative. This dataset will be abbreviated CR hereafter.

Movie Reviews Subjectivity Analysis This dataset was introduced by Pang and Lee (2004) as a collection of movie-related texts from rottentomatoes.com and imdb.com. The task is to predict whether a given sentence is subjective or objective. Subjective examples were sentences extracted from movie reviews from rottentomatoes.com and objectives examples were sentences extracted from

movie plot summaries from imdb.com. This dataset will be abbreviated SUBJ hereafter.

Movie Reviews Sentiment Analysis This dataset was introduced by Socher et al. (2013b) as the *Stanford Sentiment Treebank*. The dataset extends the prior dataset of Pang and Lee (2005), which is a set of movie reviews from rottentomatoes.com, by re-annotating them using Amazon Mechanical Turk. In the binary setting, the task is to predict whether the review is positive or negative. In the fine-grained or five-way setting, the task is to predict whether the review is negative, somewhat negative, neutral, somewhat positive, or positive. This dataset will be abbreviated as SST-2 to refer to the binary setting and as SST-2 to refer to the five-way setting hereafter.

Question Classification This dataset was introduced by Li and Roth (2002) as a collection of data for question classification and was an aggregation of data from Hovy et al. (2001) and the TREC 8 (Voorhees and Harman, 2000a), 9 (Voorhees and Harman, 2000b), and 10 (Voorhees and Harman, 2001) evaluations. The task is to predict the semantic class[6] of a question from the following categories: abbreviation, entity, description, human, location, and numerical value.[7] This dataset will be

[6]The notion of a *semantic class* for questions follows Singhal et al. (2000) and is distinct from the conceptual classes of Lehnert (1977a,b, 1986). The distinction primarily centers on the handling of factual questions (c.f. Li and Roth, 2002).

[7]The data was annotated a course-grained granularity of six classes and a fine-grained granularity of 50 classes. In this work, we use the coarse-grained granularity.

abbreviated TREC hereafter.

Why single-sentence text classification? Text classification is a standard and simple test-bed for validating that new methods in NLP have potential. It is also one of the least computationally demanding settings, which allowed us to be more comprehensive and rigorous in considering more datasets, ordering rules, and hyperparameter settings. In this work, we choose to further restrict ourselves to single-sentence datasets as the optimization we do is ill-posed. Consequently, without further constraints/structure, the solutions our algorithms generate may lack fluency across sentence boundaries. In the single-sentence setting, models must learn to grapple with this lack of systematicity in modelling across examples but the task is significantly simpler as there are no modelling challenges within a single example due to this lack of systematicity. As we were mainly interested in what could be done with relatively pure algorithmic optimization, we began with this simpler setting with fewer confounds. We discuss this as both a limitation and opportunity for future work in §6.

5.4 Experimental Conditions

In this work, we tokenize input sentences using spaCy (Honnibal and Montani, 2017). We additionally dependency parse sentences using the spaCy dependency

parser, which uses the CLEAR dependency formalism/annotation schema.[8] We then embed tokens using ELMo pretrained representations. We then freeze these representations, permute them according to the order we are studying, and then pass them through our bidirectional LSTM encoder to fine-tune for the specific task we are considering. We use a linear classifier with a `Softmax` function on top of the concatenated max-pooled hidden states produced by the LSTM.

Why `spaCy`? Using `spaCy` for both dependency parsing and tokenizing ensured that there were no complications due to misalignment between the dependency parser's training data's tokenization and our tokenization. Further, the `spaCy` dependencer parser is reasonably high-quality for English and is easily used off-the-shelf.

Why ELMo? ELMo representations are high-quality pretrained representations. While superior pretrained representations exist (e.g. BERT), we chose to use ELMo as it is was compatible with the tokenization schema of `spaCy` and introduced no further complications due to subwords. While in other works of ours, we rigorously mechanisms for converting from subword-level to word-level representations for BERT and other transformers (Bommasani et al., 2020), we chose to avoid this complication in this work as it was not of interest and was merely a confound.

[8] https://github.com/clir/clearnlp-guidelines/blob/master/md/specifications/dependency_labels.md

Why frozen representations? Peters et al. (2019b) provides compelling and comprehensive evidence the frozen representations perform better than fine-tuned representations when using ELMo as the pretrained model. Further, in this work we are interested in integrating our novel word orders with pretrained representations and wanted to isolate this effect from possible further confounds due to the nature of fine-tuning with a different word order from the pretraining word order.

Why bidirectional LSTM encoder?

As we discuss in §2.3.2, LSTMs have proven to be reliably and performant encoders across a variety of NLP tasks. Further, bidirectional LSTMs have almost uniformly led to further improvemenets, as we discuss in §2.3.4. In our `pretrain-permute-finetune` framework, it was necessary to use a task-specific encoder that was not order-agnostic as otherwise the permutations would have no effect.

Why max-pooling? In initial experiments, we found that the pooling decision had an unclear impact on results but that there was marginal evidence to suggest that max-pooling outperformed averaging (which is the only other pooling choice we considered; we did not consider the concatenation of the two as in Howard and Ruder (2018)). Further, recent work that specifically studies ELMo for text classification with extensive hyperparameter study also uses max-pooling (Peters et al., 2019b). Separately, Zhelezniak et al. (2019) demonstrate that max-pooling may have both practical advantages and theoretical justification.

5.4.1 Hyperparameters

We use input representations from ELMo that are 2048 dimensional. We use a single layer bidirectional LSTM with output dimensionality h and dropout (Srivastava et al., 2014) is introduced to the classifier input with dropout probability p as form of regularization. The weights of the LSTM and classifier are initialized according to random samples from PyTorch default distribution.[9] Optimization is done using the Adam optimizer (Kingma and Ba, 2015) and the standard cross-entropy loss, which has proven to be a robust pairing of (optimizer, objective) in numerous NLP applications. Optimizer parameters are set using PyTorch defaults.[10] Examples are presented in minibatches of size 16. The stopping condition for training is the model after training for 12 epochs. We found this threshold after looking at performance across several different epochs (those in $\{3, 6, 9, 12, 15\}$). We found that standard early-stopping methods did not reliably lead to improvements, perhaps due to the fact that models converge so quickly (hence early-stopping and a fixed threshold are near-equal). All hyperparameters that were specified above were optimized on the `SUBJ` dataset using the identity word order (r_I) to ensure this baseline was as well-optimized as possible. Parameter choices for h and p were initially optimized over $\{16, 32, 64, 128, 256\} \times \{0.0, 0.02, 0.2\}$ for the `SUBJ` (one of the easiest datasets) and `SST-5` (one of the hardest datasets), again using r_I to ensure the optimization was done to favor the baseline. From these 15 candidate hyperparameter settings,

[9] https://pytorch.org/docs/stable/nn.init.html
[10] https://pytorch.org/docs/stable/optim.html

the six highest performing were chosen[11] and all results are provided for these settings (for all word orders and all datasets) in Appendix B. All results reported subsequently are for hyperparameters individually optimized for the (word order, dataset) pair being considered. All further experimental details are deferred to §A.1.

5.5 Results and Analysis

Orders. We analyze the results for the following eight orders:

1. r_I — Each sentence is ordered as written.
2. r_r — Each sentence is ordered according to a linear layout sampled from the uniform distribution over S_n.
3. r_b — Each sentence is ordered according to the Cuthill-McKee heuristic to minimize bandwidth.
4. r_c — Each sentence is ordered according to the algorithm of Yannakakis (1985) to minimize cutwidth. The linear layout is constrained to be optimal among all projective orderings.
5. r_m — Each sentence is ordered according to the algorithm of Gildea and Temperley (2007) to minimize the minimum linear arrangement objective. The linear layout is constrained to be optimal among all projective orderings.

[11] We did this initial reduction of the hyperparameter space due to computational constraints. In doing so, we tried to ensure that the settings yielded the strongest possible baselines.

	CR			SUBJ			SST-2			SST-5			TREC		
	B	C	M	B	C	M	B	C	M	B	C	M	B	C	M
r_r	16.03	10.07	146.9	20.42	13.20	221.9	15.92	10.92	153.1	15.84	10.90	151.6	7.55	6.05	39.24
r_I	11.52	4.86	49.87	16.12	5.49	69.52	13.16	5.03	54.29	13.04	5.02	53.84	6.87	3.95	21.99
r_b	5.44	5.44	55.21	6.37	6.37	78.94	5.52	5.52	58.20	5.50	5.50	57.68	3.36	3.36	17.76
r_c	6.58	3.34	34.68	8.41	3.62	46.78	6.54	3.20	35.69	6.51	3.20	35.43	3.57	2.45	14.76
r_m	6.19	3.35	34.13	7.69	3.64	45.70	6.00	3.21	34.90	5.98	3.21	34.65	3.46	2.45	14.62
$\tilde{r}_b$	6.64	5.29	55.84	8.68	6.40	81.12	7.11	5.61	61.74	7.08	5.57	60.97	3.84	3.75	21.88
$\tilde{r}_c$	10.42	4.02	47.00	14.60	4.57	66.03	11.66	4.08	50.89	6.96	4.07	50.44	3.79	3.00	19.66
$\tilde{r}_m$	6.85	3.29	35.68	8.60	3.66	49.17	7.00	3.29	37.64	11.57	3.29	37.32	5.77	2.54	15.40

Table 5.2: Bandwidth (B), cutwidth (C), and minimum linear arrangement (M) scores for every (dataset, ordering rule) pair considered.

6. $r_{\tilde{b}}$ — Each sentence is ordered according the `Transposition Monte Carlo` algorithm to minimize bandwidth.

7. $r_{\tilde{c}}$ — Each sentence is ordered according the `Transposition Monte Carlo` algorithm to minimize cutwidth.

8. $r_{\tilde{m}}$ — Each sentence is ordered according the `Transposition Monte Carlo` algorithm to minimize the minimum linear arrangement objective.

Optimization effects. Beyond the standard distributional properties of interest in NLP for datasets (Table 5.1), it is particularly pertinent to consider the effects of our optimization algorithms on the bandwidth, minimum linear arrangement score, and cutwidth across these datasets. We report this in Table 5.2.

We begin by considering the relationship between the random word orders r_r and the standard English word orders r_I (top band of Table 5.2). In particular, we observe that across all five datasets, standard English substantially optimizes these three costs better than a randomly chosen ordering would. In the case of minimum

linear arrangement, this provides further evidence to corpus analyses conducted by Futrell et al. (2015). Similarly, for the bandwidth and cutwidth objectives, this suggests that these objectives at least correlate with costs that humans may optimize for in sentence production and processing.

We then consider the optimization using existing algorithms in the literature as compared to standard English and random word orders (top and middle bands of Table 5.2). We observe that across all datasets, all optimized ordering rules perform random word orders across all objectives. While it is unsurprising that the optimal order for a given objective outperforms the other orders and standard English, we do note that the margins are quite substantial in comparing standard English to each rule. That is to say, standard English can still be substantially further optimized for any of the given orders. Additionally, we consider the scores associated for an ordering rule that are *not* being optimized for. In particular, we see that optimizing for either cutwidth or minimum linear arrangement yields similar scores across all three objectives and all five datasets. We hypothesize this is due to both algorithms have a shared algorithmic subroutine (the disjoint strategy). Optimizing for either order yields substantial improvements over standard English across all three objectives and only marginally underperforms the bandwidth-optimal order r_b. On the other hand, we find an interesting empirical property that the cutwidth and bandwidth scores for r_b are consistently equal. This may suggest that this is a theoretical property of the algorithm that be formally proven. Further, r_b generally (except for `TREC`) yields greater minimum linear arrangements compared to English.

	CR			SUBJ			SST-2			SST-5			TREC		
	B	C	M	B	C	M	B	C	M	B	C	M	B	C	M
r_r	16.03	10.07	146.9	20.42	13.20	221.9	15.92	10.92	153.1	15.84	10.90	151.6	7.55	6.05	39.24
r_I	11.52	4.86	49.87	16.12	5.49	69.52	13.16	5.03	54.29	13.04	5.02	53.84	6.87	3.95	21.99
r_b	5.44	5.44	55.21	6.37	6.37	78.94	5.52	5.52	58.20	5.50	5.50	57.68	3.36	3.36	17.76
r_c	6.58	3.34	34.68	8.41	3.62	46.78	6.54	3.20	35.69	6.51	3.20	35.43	3.57	2.45	14.76
r_m	6.19	3.35	34.13	7.69	3.64	45.70	6.00	3.21	34.90	5.98	3.21	34.65	3.46	2.45	14.62
$\tilde{r}_b$	6.64	5.29	55.84	8.68	6.40	81.12	7.11	5.61	61.74	7.08	5.57	60.97	3.84	3.75	21.88
$\tilde{r}_c$	10.42	4.02	47.00	14.60	4.57	66.03	11.66	4.08	50.89	6.96	4.07	50.44	3.79	3.00	19.66
$\tilde{r}_m$	6.85	3.29	35.68	8.60	3.66	49.17	7.00	3.29	37.64	11.57	3.29	37.32	5.77	2.54	15.40

Table 5.3: Duplicated from Table 5.2 for convenience. Bandwidth (B), cutwidth (C), and minimum linear arrangement (M) scores for every (dataset, ordering rule) pair considered.

Next, we consider the optimization using the algorithms we introduce with the `Transposition Monte Carlo` method as compared to English and random word orders (top and bottom bands of Table 5.2). We observe that the same comparison between random word orders and the word orders we introduce to optimize objectives holds as in the case of the algorithms from the prior literature. Further, the relationship between standard English and these heuristic-based word orders mimics the trends between standard English and the word orders derived from prior algorithms. However, we find that the margins are substantially smaller. This is not particularly surprising, as it suggests that our heuristics are less effective at pure optimization than the more well-established algorithms in the literature.

When we strictly consider the word orders generated by the heuristics we introduce (bottom band of Table 5.2), we observe one striking finding immediately. In particular, the cutwidth objective evaluated on the cutwidth-minimizing order $\tilde{r}_c$ is often greater than the same objective evaluated on the minimum linear arrangement-

minimizing order $\tilde{r}_m$. A similar result holds between $\tilde{r}_b$ and $\tilde{r}_m$ for the SUBJ and SST-2 datasets. What this implies is that the greediness and transposition-based nature of Transposition Monte Carlo may more directly favor optimizing minimum linear arrangement (and that this coincides with minimizing the other objectives). Alternatively, this may suggest that a more principled and nuanced procedure is needed for the stopping parameter T in the algorithm.

Finally, we discuss the relationship between the algorithms given in prior work (and the corresponding word orders) and the algorithms we present under the Transposition Monte Carlo framework (and the corresponding word orders). Here, we compare the middle and bottom bands of Table 5.2. As observed previously, we see that for the score being optimized, the corresponding word order based on an algorithm from the literature outperforms the corresponding word order based on an algorithm we introduce. More broadly, we see the quality of the optimization across almost all objectives and datasets for all ordering rule pairs $r_k, \tilde{r}_k$ for $k \in \{b, c, m\}$ is better for r_k than $\tilde{r}_k$. This is consistent with our intuition previously — our heuristics sacrifice the extent to which they purely optimize the objective being considered to retain aspects of the original linear layout π_I. In the analysis of downstream results, we will address whether this compromise provides any benefits in modelling natural language computationally.

Downstream Performance. Given the framing and backdrop we have developed thus far, we next consider the downstream effects of our novel word orders. In

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.852	0.955	**0.896**	0.485	0.962
r_r	0.842	0.95	0.877	0.476	0.954
r_b	*0.854*	0.952	0.873	0.481	*0.966*
r_c	**0.86**	0.953	0.874	0.481	0.958
r_m	0.841	0.951	0.874	0.482	*0.962*
$\tilde{r}_b$	*0.852*	0.949	0.882	0.478	0.956
$\tilde{r}_c$	0.849	*0.956*	0.875	**0.494**	**0.968**
$\tilde{r}_m$	0.844	**0.958**	0.876	0.476	*0.962*

Table 5.4: Full classification results where the result reported is the max across hyperparameter settings. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`. The best performing ordering rule for a given dataset is indicated in **bold**. Any ordering rule (that is neither the best-performing order rule nor r_I) that performs at least as well as r_I for a given dataset is indicated in *italicized magenta*.

Table 5.4, we report these results as well as the results for the random word order baseline r_r and the standard English word order r_I. In particular, recall that the r_I performance is reflective of the performance of the state-of-the-art paradigm in general: `pretrain-and-finetune`.

Each entry reflects the optimal choice of hyperparameters (among those we considered) for the corresponding model on the corresponding dataset (based on validation set performance). In Appendix B, we provide additional results for all hyperparameter settings we studied[12] as well as results for all hyperparameters

[12]These results appear in Tables B.1–B.6.

with the stopping condition of training for 15 epochs (we observed no sustained improvements for any model using any order on any dataset beyond this threshold).[13]

We begin by considering the comparison between the random word order r_r and the standard English word order r_I (top band of Table 5.4). Note that for r_r, we are using a bidirectional LSTM in the task-specific modelling as a type of set encoder. While previous works such as Bommasani et al. (2019) have also used RNN variants in order-invariant settings, this is fairly nonstandard and requires that the model learns permutation equivariance given that the order bears no information. Unsurprisingly, we see that r_I outperforms r_r across all datasets. However, the margin is fairly small for all five datasets. From this, we can begin by noting that ELMo already is a powerful pretrained encoder and much of the task-specific modelling could have just been achieved by a shallow classifier on top of the ELMo representations. Further, we note that this attunes us to a margin that is fairly significant (the difference between whatever can be gained from the English word order and the pretrained contextualized word representations versus what can be gleaned only from the pretrained contextualized word representations).

Next, we consider the comparison of the word orders derived via combinatorial optimization algorithms in the literature and the baselines of standard English

[13]These results appear in Tables B.7–B.12.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.852	0.955	**0.896**	0.485	0.962
r_r	0.842	0.95	0.877	0.476	0.954
r_b	*0.854*	0.952	0.873	0.481	*0.966*
r_c	**0.86**	0.953	0.874	0.481	0.958
r_m	0.841	0.951	0.874	0.482	*0.962*
$\tilde{r}_b$	*0.852*	0.949	0.882	0.478	0.956
$\tilde{r}_c$	0.849	*0.956*	0.875	**0.494**	**0.968**
$\tilde{r}_m$	0.844	**0.958**	0.876	0.476	*0.962*

Table 5.5: Duplicated from Table 5.4 for convenience. Full classification results where the result reported is the max across hyperparameter settings. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`. The best performing ordering rule for a given dataset is indicated in **bold**. Any ordering rule (that is neither the best-performing order rule nor r_I) that performs at least as well as r_I for a given dataset is indicated in *italicized magenta*.

and the randomized word order (top and middle bands of Table 5.4). We begin by noting that the optimized word orders *do not* always outperform r_r. This is most salient in looking at the results for `SST-2`. While the margin is exceedingly small when the optimized word orders underperform against r_r, this suggests that the lack of systematicity or regularity in the input (perhaps due to the optimization being underdetermined) makes learning from the word order difficult. While none of r_b, r_c, r_m consistently perform as well or better than r_I (r_b performs slightly better on two datasets, slightly worse on two datasets, and substantially worse on `SST-2` as, arguably, the best-performing of the three), there are multiple instances where, for specific datasets (`CR` and `TREC`), they do outperform r_I. This alone may suggest

that a more careful design of ordering rules could yield improvements. Again, recall this is a highly controlled comparison with something that is a reasonable stand-in for the state-of-the-art regime of `pretrain-and-finetune`.

Looking at the comparison between the word orders derived via algorithms from the literature and their analogues derived from `Transposition Monte Carlo` variants (middle and bottom bands of Table 5.5), we observe that neither set of orders strictly dominates the other. However, on four of the five datasets (all but `CR`), the best-performing order is one produced by `Transposition Monte Carlo`. This provides evidence to the claim that the sole goal in constructing order rules for downstream NLP of pure combinatorial optimization is insufficient and arguably naive. Instead, attempting a balance between retaining information encoded in the original linear layout π_I while performing some reduction of dependency-related costs may be beneficial. We revisit this in §6.3.

Finally, we compare the word orders produced using our heuristic to the baselines (top and bottom bands of Table 5.5). We begin by observing that in this case, almost every ordering rule outperforms the random word order (the sole exception being $\tilde{r}_b$ for `SUBJ` by a margin of just 0.001). Further, while no single ordering rule reliably outperforms r_I ($\tilde{r}_c$ significantly outperforms r_I on two datasets, is significantly outperformed on one dataset, performs marginally better on one dataset, and performs marginally worse on dataset), we do find a word order that outperforms r_I

among $\tilde{r}_b, \tilde{r}_c, \tilde{r}_m$ in four of the five datasets (the exception being `SST-2`). In fact, the majority (i.e. on three of the five datasets) of the best-performing ordering rules of all considered are word orders produced using `Transposition Monte Carlo`. This is additional evidence to suggest the merits of novel ordering rules (that are not standard English) for downstream NLP and that their design likely should try to maintain some of the information-encoding properties of standard English.

CHAPTER 6

CONCLUSIONS

To conclude, in this chapter, we review the findings presented. We further provide the open problems we find most pressing, the future directions we find most interesting, and the inherent limitations we find most important.

6.1 Summary

In this thesis, we have presented a discussion of the current understanding of word order as it pertains to human language. Dually, we provide a description of the set of approaches that have been taken towards modelling word order in computational models of language.

We then focus on how psycholinguistic approaches to word order can be generalized and formalized under a rich algorithmic framework. In particular, we concentrate on word orders that can be understood in terms of linear layout optimization using dependency parses as a scaffold that specifies the underlying graph structure. We describe existing algorithms for the BANDWIDTH, MINLA, and CUTWIDTH problems with a special interest on algorithms when projectivity constraints are imposed. In doing so, we show that the algorithms of Gildea and Temperley (2007) and Yannakakis (1985) can be interpreted on a shared framework. Further, we correct some

nuances in the algorithm and analysis of Gildea and Temperley (2007). As a taste of how algorithmic interpretation could be used to reinterpret existing psycholinguistic approaches, we also prove Theorem 4.1. This result establishes a connection between capacity and memory constraints that was previously unknown in the psycholinguistics literature (to the author's knowledge). To accompany these provable results, we also introduce simple heuristic algorithms (i.e. `Transposition Monte Carlo` algorithms for each of the three objectives we considered). These algorithms provide a mechanism for balancing pure algorithmic optimization with the additional constraints that might be of interest in language (e.g preserving other types of information encoded in word order).

With these disparate foundations established, the last portion of this thesis discusses the relationship between novel/alternative word orders and computational models of language. We introduce the `pretrain-permute-finetune` framework to seamlessly integrate our novel orders into the *de facto* `pretrain-and-finetune` paradigm. In particular, our approach incurs a near-trivial one-time cost and makes no further changes to the pretraining process, training and inference costs (both with respect to time and memory), or model architecture. We then examine the empirical merits of our method on several tasks (and across several hyperparameter settings), showing that our heuristic algorithms may sometimes outperform their provable brethren and that, more broadly, novel word orders can outperform using standard English. In doing so, we establish the grounds for considering the utility of this approach for other languages, for other tasks, and for other models.

6.2 Open Problems

Weighted optimization. In this work, we consider orders specified based on objectives evaluated over the dependency parse $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. In representing the parse in this way, we ignore the labels and directionality of the edges.[1] While using a bidirectional encoder in the finetuning step of the `pretrain-permute-finetune` framework may alleviate concerns with directionality, we are ultimately neglecting information made available in the dependency parse. We take this step in our approach since there are known algorithms for the unweighted and undirected version of the combinatorial optimization problems we study. Nonetheless, studying provable and heuristic algorithms for the weighted version of these problems may be of theoretical interest and of empirical value. In particular, this would allow one to specify a weighting on edges dependent on their dependency relation types, which may better represent how certain dependencies are more or less important for downstream modelling.

Alternative scaffolds. In this work, we use dependency parses as the syntactic formalism for guiding the optimization. While this choice was well-motivated given the longstanding literature on dependency locality effects and that the linear tree structure of dependency parses facilitates algorithmic framing in the language of linear layouts, alternative syntactic structures such as constituency trees or semantic

[1] Recall how we defined $\mathcal{E}$ as the unlabelled and undirected version of the true edge set of the dependency parse, $\mathcal{E}_\ell$.

formalisms such as AMR semantic parses are also interesting to consider. This yields natural open questions of what appropriate scaffolds are, what resulting objectives and algorithmic approaches are appropriate, and how both the linguistic and algorithmic primitive interact with the downstream NLP task. Further inquiry might also consider the merits of combining multiple approaches given the benefits of joint and multi-task learning and fundamental questions regarding what one linguistic representation may provide that is unavailable from others.

Information in Order. A central scientific question in studying human language is understanding what information is encoded in word order and how this information is organized. In this work, given that we argue for the merits of representing sentences using two distinct orders (one during the embedding stage and one during the fine-tuning stage), this question is of added value. In particular, a formal theory of what information is available with one order and another (from an information theoretic perspective) might be of value. Arguable an even more interesting question revolves around considering the ease of extracting the information given one ordering or another. Such a framing of orders facilitating the ease of extraction of information (from computational agents with bounded capacities, which is quite different from standard information theory) might thereafter induce a natural optimization-based approach to studying order — select the order that maximizes the eases of extraction of certain information that is of interest.

6.3 Future Directions

In the previous section, we state open problems that are of interest and merit study. In this section, we provide more concrete future directions and initial perspectives on how to operationalize these directions.

End-to-end permutation learning. In this work, we generate novel word orders that are task-agnostic and that optimizes objectives that are not directly related with downstream performance. While there are fundamental questions about the relationships between discrete algorithmic ideas, classical combinatorial optimization, and modern deep learning, our approach runs stylistically counter to the dominant paradigms of NLP at present. In particular, the notion of attention emerged in machine translation as a soft way of aligning source and target and has proven to empirically more effective than harder alignments. Dually, such end-to-end optimization of attention weights implies that attention can be optimized in a task-specific way, which likely implies better performance. For these reason, studying end-to-end methods for (differentiably) finding orders may be particularly attractive as this would imply the permutation component could be dropped into existing end-to-end differentiable architecture. While reinforcement learning approaches to permutation generation/re-ordering may be natural, we believe a promising future direction would be direct learning of permutations. In this sense, a model would learn permutations of the input that correspond with improved downstream performance.

At first glance, such optimization of permutations seems hard to imagine given that permutations are sparse/discrete and therefore unnatural for differentiable optimization. A recent line of work in the computer vision and machine learning communities has proposed methods towards differentiable/gradient-based optimization (Santa Cruz et al., 2019; Mena et al., 2018). In particular, many of these methods consider a generalization of the family of permutation matrices to the family of double stochastic matrices (DSMs). Importantly, DSMs specify a polytope (the Birkhoff polytope) which can be reached in a differentiable way via Sinkhorn iterations (Sinkhorn, 1964; Sinkhorn and Knopp, 1967). Given these works, a natural question is whether these methods can be extended to sequential data (which are not of a fixed length, unlike image data), perhaps with padding as is done in Transformers, and what that might suggest for downstream NLP performance.

Additional constraints to induce well-posed optimization. In our approach, the solution to the optimization is not necessarily unique. In particular, for all three objectives, the reverse of an optimal sequence is also optimal. Further, Gildea and Temperley (2007) claim (without proof) that there can be $2^{\frac{n}{2}}$ optimal solutions for the optimization problem they study. Given that there may be many optima and potentially exponentially many, downstream models are faced with uncertainty about the structure of the input for any given input. In particular, the word orders we are considering may lack regularity and systematicity properties that are found in natural languages, though the structure of the disjoint strategy may help alleviate

this.[2] Developing additional constraints to further regularize the optima may benefit modelling as different examples will have more standardized formats. Given the observations we discuss in §3.2 regarding harmonic word orders improving language acquisition for child language learners (by appealing to their inductive biases), such as those of Culbertson and Newport (2017), enforcing word order harmonies globally may be a natural means for increases regularity and systematicity in the resultant word orders. Similarly, such approaches may provide additional benefits when extended to the (more general) setting where inputs (may) contain multiple sentences.

Information locality. In §3.3.3, we discuss the recent proposal for information locality as a generalization of dependency locality. As syntactic dependencies are clearly only a subset of the linguistic information of interest for a downstream model, a richer theory of locality effects and a broad classes of information may prove fruitful both in study human language processing and in motivating further novel orders. More broadly, we specifically point to the work of (Xu et al., 2020) which introduces the $\mathcal{V}$-information theory. In particular, their work builds the standard preliminaries and primitives of Shannon's information theory (Shannon, 1948) with the notion of computational constraints. While they envision this in the context of motivating modern representation learning, we believe this may also be better theoretical machinery for information theoretic treatments of human

[2] They also may lack overt linear proxies for compositionality to the extent found in the corresponding natural language.

language processing. After all, humans also have severely bounded processing capabilities.

Understanding impact of novel orders. In this work, we put forth a rigorous and highly controlled empirical evaluation of the word orders we introduce. However, our results and analysis do not explain *why* and *how* the novel word orders lead to improved performance. In particular, a hypothesis that serves as an undercurrent for this work is that the novel word orders we introduce *simplify* computational processing by reducing the need for modelling long-distance dependencies and/or many dependencies simultaneously. Future work that executes a more thorough and nuanced analysis of model behavior and error patterns would help to validate the extent to which this hypothesis is correct. Additionally, it would help to disambiguate what modelling improvements that the work yields coincide with other modelling techniques and what improvements are entirely orthogonal to suggest how this work can integrate with other aspects of the vast literature on computational modelling in NLP.

Broader NLP evaluation. Perhaps the most obvious future direction is to consider a broader evaluation of our framework. In particular, evaluating across pretraining architectures, fine-tuning architectures, datasets, tasks, and languages are all worthwhile for establishing the boundaries of this method's viability and utility. Additionally, such evaluations may help to further refine the design of novel word

orders and suggest more nuanced procedures for integrating them (especially in settings where the dominant paradigm itself of `pretrain-and-finetune` is insufficient). Additionally, we posit that there may be natural interplay with our work and work on cross-lingual and multi-lingual methods that may especially merit concentrated study. Concretely, one could imagine applying the same ordering rule cross-linguistically and therefore normalizing some of the differences across languages at the input level. Consequently, there may be potential for this to improve alignment (e.g. between word embeddings, between pretrained encoders) for different language or better enable cross-lingual transfer/multi-lingual representation learning.

6.4 Consequences

Previously, we have summarized our contributions (§6.1). In this section, we instead provide more abstract lessons or suggestive takeaways from this work.

Word order in NLP. In this work, we clearly substantiate that there is tremendous scope for improving how word order is considered within NLP and explicitly illuminate well-motivated directions for further empirical work. While our empirical results legitimately and resolutely confirm that theoretically-grounded algorithmic optimization may coincide with empirical NLP improvements, it remains open to what extent this is pervasive across NLP tasks and domains.

Generalizing measures of language (sub)optimality. In this work, we put forth alternative objectives beyond those generally considered in the dependency length minimization literature that may merit further psycholinguistic inquiry. In particular, by adopting the generality of an algorithmic framework, one can naturally pose many objectives that intuitively may align with human optimization in language production and comprehension. Further, in this work we clarify the extent to which human language is optimal and we note that the suboptimality of human language may be equally useful for understanding language's properties as its optimality.

Interlacing psycholinguistic or algorithmic methods with NLP. In this work, we present work that is uniquely at the triple intersection of psycholinguistics, algorithms, and NLP. While such overtly interdisciplinary work is of interest, we also argue that there is great value in considering the interplay between psycholinguistics and NLP or between algorithms and NLP, if not all three simultaneously. There is a long-standing tradition of connecting algorithms/computational theory with computational linguistics and NLP in the study of grammars and parsing (Chomsky, 1957, 1965; Kay, 1967; Earley, 1970; Joshi et al., 1975; Charniak, 1983; Pereira and Warren, 1983; Kay, 1986; Steedman, 1987; Kay, 1989; Eisner, 1996; Collins, 1996, 1997; Charniak et al., 1998; Gildea and Jurafsky, 2002; Collins, 2003; Klein and Manning, 2003b,a; Steedman, 2004; McDonald et al., 2005b; McDonald and Pereira, 2006; Mitkov and Joshi, 2012; Chomsky, 2014b,a). Trailblazers of the field, such as

Noam Chomsky and the recently-passed Arvind Joshi made great contributions in this line. Similarly, there are has been a recent uptick in the borrowing of methods from psycholinguistics to rigorously study human language processing[3] to interpret and understand neural models of language (Linzen et al., 2016; Gulordava et al., 2018; van Schijndel and Linzen, 2018; Wilcox et al., 2018; Marvin and Linzen, 2018; Ravfogel et al., 2019; van Schijndel et al., 2019; McCoy et al., 2019; Futrell and Levy, 2019; Wilcox et al., 2019; Futrell et al., 2019; van Schijndel and Linzen, 2019; Prasad et al., 2019; Ettinger, 2020; Davis and van Schijndel, 2020).[4] And once again, another seminal mind of the field, Ron Kaplan, spoke to precisely this point in his keynote talk "*Computational Psycholinguistics*" (Kaplan, 2020) at ACL 2019 as he received the most recent ACL Lifetime Achievement Award. However, in the present time, beyond parsing and interpretability research, we see less interplay between either algorithms or psycholinguistics and NLP. Perhaps this thesis may serve as an implicit hortative to reconsider this.

6.5 Limitations

In the spirit of diligent science, we enumerate the limitations we are aware of with this work. We attempt to state these limitations fairly without trying to undercut or lessen their severity. We also note that we have considered the ethical ramifications

[3]The first blackbox language learner we have tried to understand.
[4]The second blackbox language learner we have tried to understand.

of this work[5] and remark that we did not come to find any ethical concerns.[6]

Dependence on Dependencies. In this work, we generate novel word orders contingent on access to a dependency parse. Therefore, the generality of our method is dependent on access to such a parse. Further, since we exclusively consider English dependency parsers, which are substantially higher quality than dependency parsers for most languages and especially low-resource languages, and datasets which are drawn from similar data distributions as the training data of the dependency parse, it is unclear how well are model with perform in setting where weaker dependency parsers are available or there are domain-adaptation concerns.

Rigid Optimization. By design, our approach is to purely optimize an objective pertaining to dependency locality. While the heuristic algorithms we introduce under the `Transposition Monte Carlo` framework afford some consideration of negotiating optimization quality with retention of the original sentence's order (through the parameter T), our methods provide no natural recipe for integration of arbitrary metadata or auxiliary information/domain knowledge.

[5] This was inspired by the NeurIPS 2020 authorship guidelines, which require authors to consider the social and ethical impacts of their work.

[6] While not an ethical concern of the research, we do note that all figures and tables in this work are designed to be fully colorblind-friendly. In particular, while many figures/tables use color, they can also be interpreted using non-color markers and these markers are referenced in the corresponding caption.

Evaluation Settings. As we note in §5.4, we elect to evaluate on single-sentence text classification datasets. While we do provide valid and legitimate reasons for this decision, it does imply that the effectiveness of our methods for other tasks and task types (e.g. sequence labelling, natural language generation, question answering) is left unaddressed. Further, this means that the difficulties of intra-example irregularity across sentences (due to optimization not being sufficiently determined/constrained) are not considered. As a more general consideration, Linzen (2020) argues that the evaluation protocol used in this work may not be sufficient for reaching the desired scientific conclusions regarding word order.

Alternative Orders in Pretraining. Due to computational constraints, we only study introducing permutations/re-orderings we generate after embedding using a pretrained encoder. However, a more natural (and computationally intensive/environmentally detrimental (Strubell et al., 2019)) approach that may offer additional value is to use the orders during pretraining. As such, our work does not make clear that `pretrain-permute-finetune` is the best way to introduce our permutations into the standard `pretrain-and-finetune` regime and fails to consider an "obvious" alternative.

English Language Processing. In studying the effects of our orders empirically, we only use English language data.[7] Several works have shown that performance

[7] In this work, we did not reconsider the source of the pretraining data for ELMo or the datasets we used in evaluating our method. Therefore we are not certain, but it seems likely that data mainly

on English data is neither inherently representative nor likely to be representative of performance in language processing across the many languages of the world (Bender, 2009, 2011, 2012; Joshi et al., 2020b). Further, recent works have shown that English may actually be favorable for approaches that make use of LSTMs and RNNs (Dyer et al., 2019; Davis and van Schijndel, 2020), which are models we explicitly use in this work. And finally, the entire premise of dependency locality heavily hinges on typological features of the language being studied.[8] Therefore, the results of this work are severely limited in terms of claims that can be made for natural languages that are not English.

represents Standard American English and not other variants of English such as African American Vernacular English.

[8]This can be inferred from our discussion of dependency locality and word ordering effects in various natural languages in §3; a particularly salient example is that the notion of dependency locality is quite different in morphologically-rich languages as information is often conveyed through morphological markers and intra-word units rather than word order.

BIBLIOGRAPHY

Donald L. Adolphson and T. C. Hu. 1973. Optimal linear ordering. *SIAM Journal on Applied Mathematics*, 25(3):403–423.

Charu C. Aggarwal and ChengXiang Zhai. 2012. A survey of text classification algorithms. In *Mining text data*, pages 163–222. Springer.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. 2017. Deep variational information bottleneck. In *International Conference on Learning Representations*.

Rubayyi Alghamdi and Khalid Alfalqi. 2015. A survey of topic modeling in text mining. *International Journal of Advanced Computer Science and Applications*, 6.

Nada Almarwani, Hanan Aldarmaki, and Mona Diab. 2019. Efficient sentence embedding using discrete cosine transform. 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)*, pages 3672–3678, Hong Kong, China. Association for Computational Linguistics.

Anima Anandkumar, Dean P. Foster, Daniel J. Hsu, Sham M. Kakade, and Yi-Kai Liu. 2012. A spectral algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 917–925.

Peggy M. Andersen, Philip J. Hayes, Steven P. Weinstein, Alison K. Huettner, Linda M. Schmandt, and Irene B. Nirenburg. 1992. Automatic extraction of

facts from press releases to generate news stories. In *Third Conference on Applied Natural Language Processing*, pages 170–177, Trento, Italy. Association for Computational Linguistics.

Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. 2013. A practical algorithm for topic modeling with provable guarantees. In *International Conference on Machine Learning*, pages 280–288.

Sanjeev Arora, Rong Ge, and Ankur Moitra. 2012. Learning topic models–going beyond svd. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 1–10. IEEE.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016. A latent variable model approach to PMI-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International Conference on Learning Representations*.

Sitaram Asur and Bernardo A. Huberman. 2010. Predicting the future with social media. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 1, pages 492–499.

Ariful Azad, Mathias Jacquelin, Aydin Buluç, and Esmond G. Ng. 2017. The reverse cuthill-mckee algorithm in distributed-memory. In *2017 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 22–31.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, Baltimore, Maryland. Association for Computational Linguistics.

Brian Bartek, Richard L Lewis, Shravan Vasishth, and Mason R. Smith. 2011. In search of on-line locality effects in sentence comprehension. *Journal of experimental psychology. Learning, memory, and cognition*, 37 5:1178–98.

Mikhael Bautin, Lohit Vijayarenu, and Steven Skiena. 2008. International sentiment analysis for news and blogs. pages 19–26.

Otto Behaghel. 1932. *Deutsche Syntax: eine geschichtliche Darstellung : Wortstellung ; Periodenbau*. v. 4.

Emily M. Bender. 2009. Linguistically naïve != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece. Association for Computational Linguistics.

Emily M. Bender. 2011. On achieving and evaluating language-independence in nlp. *Linguistic Issues in Language Technology*, 6.

Emily M. Bender. 2012. 100 things you always wanted to know about linguistics but were afraid to ask*. In *Tutorial Abstracts at the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada. Association for Computational Linguistics.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg.

Gunter Blache, Marek Karpinski, and Jürgen Wirtgen. 1997. On approximation intractability of the bandwidth problem. Technical report.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Leonard Bloomfield. 1933. *Language*. The University of Chicago Press.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Rishi Bommasani. 2019. Long-distance dependencies don't have to be long: Simplifying through provably (approximately) optimal permutations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student*

Research Workshop, pages 89–99, Florence, Italy. Association for Computational Linguistics.

Rishi Bommasani and Claire Cardie. 2019. Towards understanding position embeddings. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, Florence, Italy. Association for Computational Linguistics.

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting pretrained contextualized representations via reductions to static embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Rishi Bommasani, Arzoo Katiyar, and Claire Cardie. 2019. SPARSE: Structured prediction using argument-relative structured encoding. In *Proceedings of the Third Workshop on Structured Prediction for NLP*, pages 13–17, Minneapolis, Minnesota. Association for Computational Linguistics.

Rodrigo A. Botafogo. 1993. Cluster analysis for hypertext systems. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 116–125.

Jan A. Botha, Emily Pitler, Ji Ma, Anton Bakalov, Alex Salcianu, David Weiss, Ryan McDonald, and Slav Petrov. 2017. Natural language processing with small feed-forward networks. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2879–2885, Copenhagen, Denmark. Association for Computational Linguistics.

Samuel R. Bowman, Christopher Potts, and Christopher D. Manning. 2015. Recursive neural networks can learn logical semantics. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 12–21, Beijing, China. Association for Computational Linguistics.

James Bradbury, Stephen Merity, Caiming Xiong, and Richard Socher. 2017. Quasi-recurrent neural networks. In *International Conference on Learning Representations*.

Lea Brown. 2001. A grammar of nias selatan.

Peter Brown, Stephen Della Pietra, Vincent Dellapietra, John Lafferty, and L. Mercer. 1992. Analysis, statistical transfer, and synthesis in machine translation. In *Proc. International Conference on Theoretical Methodological Issues in Machine Translation, 1992*.

Mary Elaine Califf and Raymond J. Mooney. 1997. Relational learning of pattern-match rules for information extraction. In *CoNLL97: Computational Natural Language Learning*.

Ramon Ferrer-i Cancho. 2006. Why do syntactic links not cross? *Europhysics Letters (EPL)*, 76(6):1228–1235.

Ramon Ferrer-i Cancho and Ricard V. Solé. 2003. Least effort and the origins of scaling in human language. *Proceedings of the National Academy of Sciences of the United States of America*, 100(3):788–791.

Claire Cardie. 1997. Empirical methods in information extraction. *AI Magazine*, 18(4):65–80.

W. M. Chan and Alan George. 1980. A linear time implementation of the reverse cuthill-mckee algorithm. *BIT*, 20:8–14.

Pi-Chuan Chang, Daniel Jurafsky, and Christopher D. Manning. 2009. Disambiguating "DE" for Chinese-English machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 215–223, Athens, Greece. Association for Computational Linguistics.

Eugene Charniak. 1983. A parser with something for everyone. *Parsing natural language*, pages 117–149.

Eugene Charniak. 1994. *Statistical Language Learning*. MIT Press, Cambridge, MA, USA.

Eugene Charniak, Sharon Goldwater, and Mark Johnson. 1998. Edge-based best-first chart parsing. In *Sixth Workshop on Very Large Corpora*.

Michael Chau and Jennifer Xu. 2012. Business intelligence in blogs: Understanding consumer interactions and communities. *MIS quarterly*, pages 1189–1216.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. Technical report, Google.

Danqi Chen and Christopher D. Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Xilun Chen, Yu Sun, Ben Athiwaratkun, Claire Cardie, and Kilian Weinberger. 2018. Adversarial deep averaging networks for cross-lingual sentiment classification. *Transactions of the Association for Computational Linguistics*, 6:557–570.

Nancy Chinchor, Lynette Hirschman, and David D. Lewis. 1993. Evaluating message understanding systems: An analysis of the third message understanding conference (MUC-3). *Computational Linguistics*, 19(3):409–450.

Phyllis Z. Chinn, Jarmila Chvátalová, Alexander K. Dewdney, and Norman E. Gibbs. 1982. The bandwidth problem for graphs and matrices — a survey. *Journal of Graph Theory*, 6(3):223–254.

Chung-Cheng Chiu, Tara N. Sainath, Yonghui Wu, Rohit Prabhavalkar, Patrick Nguyen, Zhifeng Chen, Anjuli Kannan, Ron J. Weiss, Kanishka Rao, Ekaterina Gonina, et al. 2018. State-of-the-art speech recognition with sequence-to-sequence models. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4774–4778. IEEE.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. 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), pages 1724–1734, Doha, Qatar. Association for Computational Linguistics.

Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. 2005. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Noam Chomsky. 1956. Three models for the description of language. *IRE Transactions on Information Theory*, 2(3):113–124.

Noam Chomsky. 1957. Syntactic structures.

Noam Chomsky. 1965. *Aspects of the Theory of Syntax*. The MIT Press, Cambridge.

Noam Chomsky. 1986. *Knowledge of language: Its nature, origin, and use*. Greenwood Publishing Group.

Noam Chomsky. 2014a. *Aspects of the Theory of Syntax*, volume 11. MIT Press, Cambridge, MA, USA.

Noam Chomsky. 2014b. *The minimalist program*. MIT Press, Cambridge, MA, USA.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 93–98, San Diego, California. Association for Computational Linguistics.

Fan Chung. 1984. On optimal linear arrangements of trees. *Computers & mathematics with applications*, 10(1):43–60.

Andy Clark. 2013. Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences*, 36(3):181–204.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. In *International Conference on Learning Representations*.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 16–23, Madrid, Spain. Association for Computational Linguistics.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 1–8. Association for Computational Linguistics.

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Michael Collins, Philipp Koehn, and Ivona Kučerová. 2005. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 531–540, Ann Arbor, Michigan. Association for Computational Linguistics.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *34th Annual Meeting of the Association for Computational Linguistics*, pages 184–191, Santa Cruz, California, USA. Association for Computational Linguistics.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 160–167, New York, NY, USA. ACM.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Bernard Comrie. 1981. *Language universals and linguistic typology*. Oxford.

José R Correa and Andreas S Schulz. 2004. Single machine scheduling with precedence constraints. In *International Conference on Integer Programming and Combinatorial Optimization*, pages 283–297. Springer.

Jennifer Culbertson and Simon Kirby. 2016. Simplicity and specificity in language: Domain-general biases have domain-specific effects. *Frontiers in Psychology*, 6.

Jennifer Culbertson and Elissa L. Newport. 2015. Harmonic biases in child learners: In support of language universals. *Cognition*, 139:71 – 82.

Jennifer Culbertson and Elissa L. Newport. 2017. Innovation of word order harmony across development. *Open Mind*, 1(2):91–100.

Jennifer Culbertson, Paul Smolensky, and Géraldine Legendre. 2012. Learning biases predict a word order universal. *Cognition*, 122(3):306 – 329.

Elizabeth Cuthill and James McKee. 1969. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th national conference*, pages 157–172.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Joachim Daiber, Miloš Stanojević, Wilker Aziz, and Khalil Sima'an. 2016. Examining the relationship between preordering and word order freedom in machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 118–130, Berlin, Germany. Association for Computational Linguistics.

Forrest Davis and Marten van Schijndel. 2020. Recurrent neural network language models always learn english-like relative clause attachment. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Timothy A. Davis. 2006. *Direct Methods for Sparse Linear Systems*.

Jeffrey Dean. 2009. Challenges in building large-scale information retrieval systems:

invited talk. In *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*, pages 1–1, New York, NY, USA.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

Vera Demberg. 2010. Broad-coverage model of prediction in human sentence processing.

Vera Demberg and Frank Keller. 2008a. Data from eye-tracking corpora as evidence for theories of syntactic processing complexity. *Cognition*, 109(2):193 – 210.

Vera Demberg and Frank Keller. 2008b. A psycholinguistically motivated version of TAG. In *Proceedings of the Ninth International Workshop on Tree Adjoining Grammar and Related Frameworks (TAG+9)*, pages 25–32, Tübingen, Germany. Association for Computational Linguistics.

Vera Demberg and Frank Keller. 2009. A computational model of prediction in human parsing: Unifying locality and surprisal effects. In *CogSci 2009 Proceedings*, pages 1888–1893. Cognitive Science Society.

Vera Demberg, Frank Keller, and Alexander Koller. 2013. Incremental, predictive parsing with psycholinguistically motivated tree-adjoining grammar. *Computational Linguistics*, 39(4):1025–1066.

John DeNero and Jakob Uszkoreit. 2011. Inducing sentence structure from parallel corpora for reordering. In *Proceedings of the 2011 Conference on Empirical*

Methods in Natural Language Processing, pages 193–203, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Desmond C. Derbyshire. 1979. *Hixkaryana*, volume 1 of *Lingua Descriptive Studies*. North-Holland, Amsterdam.

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.

Josep Díaz, Mathew Penrose, Jordi Petit, and Maria Serna. 1999. Layout problems on lattice graphs. volume 1627, pages 103–112.

Josep Díaz, Jordi Petit, and Maria Serna. 2002. A survey of graph layout problems. *ACM Computing Surveys (CSUR)*, 34(3):313–356.

Myles Dillon and Donncha Ó Cróinin. 1961. *Teach Yourself Irish*. The English Universities Press Ltd., London.

Jesse Dodge, Suchin Gururangan, Dallas Card, Roy Schwartz, and Noah A. Smith. 2019. Show your work: Improved reporting of experimental results. 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)*, pages 2185–2194, Hong Kong, China. Association for Computational Linguistics.

Pedro Domingos and Michael Pazzani. 1997. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29(2-3):103–130.

Timothy Dozat and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *ArXiv*, abs/1611.01734.

Timothy Dozat, Peng Qi, and Christopher D. Manning. 2017. Stanford's graph-based neural dependency parser at the CoNLL 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Rebecca Dridan and Stephan Oepen. 2012. Tokenization: Returning to a long solved problem — a survey, contrastive experiment, recommendations, and toolkit — . In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 378–382, Jeju Island, Korea. Association for Computational Linguistics.

Matthew S. Dryer. 1988. Object-verb order and adjective-noun order: Dispelling a myth. *Lingua*, 74(2):185 – 217. Papers in Universal Grammar: Generative and Typological Approaches.

Matthew S. Dryer. 1992. The greenbergian word order correlations. *Language*, 68(1):81–138.

Matthew S. Dryer. 1997. On the six-way word order typology. *Studies in Language. International Journal sponsored by the Foundations of Language*, 21(1):69–103.

Matthew S. Dryer. 1998. Why statistical universals are better than absolute universals. pages 1–23.

Matthew S. Dryer. 2013a. Determining dominant word order. In Matthew S. Dryer and Martin Haspelmath, editors, *The World Atlas of Language Structures Online*. Max Planck Institute for Evolutionary Anthropology, Leipzig.

Matthew S. Dryer. 2013b. On the six-way word order typology, again. *Studies in Language. International Journal sponsored by the Foundations of Language*, 37(2):267–301.

Homer Dudley. 1939. The vocoder. pages 122–126. Bell Labs.

Homer Dudley, RR Riesz, and SSA Watkins. 1939. A synthetic speaker. *Journal of the Franklin Institute*, 227(6):739–764.

Chris Dyer. 2017. Should neural network architecture reflect linguistic structure? In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, page 1, Vancouver, Canada. Association for Computational Linguistics.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Chris Dyer, Gábor Melis, and Phil Blunsom. 2019. A critical analysis of biased parsers in unsupervised parsing.

Chris Dyer and Philip Resnik. 2010. Context-free reordering, finite-state translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 858–866, Los Angeles, California. Association for Computational Linguistics.

Jay Earley. 1970. An efficient context-free parsing algorithm. *Commun. ACM*, 13(2):94–102.

Jacob Eisenstein. 2019. *Introduction to Natural Language Processing*. Adaptive Computation and Machine Learning series. MIT Press.

Jason Eisner and Noah A. Smith. 2005. Parsing with soft and hard constraints on dependency length. In *Proceedings of the Ninth International Workshop on Parsing Technology*, pages 30–41, Vancouver, British Columbia. Association for Computational Linguistics.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Jeffrey L. Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Kawin Ethayarajh. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 91–100, Melbourne, Australia. Association for Computational Linguistics.

Kawin Ethayarajh, David Duvenaud, and Graeme Hirst. 2019. Towards understanding linear word analogies. In *Proceedings of the 57th Annual Meeting of the*

Association for Computational Linguistics, pages 3253–3262, Florence, Italy. Association for Computational Linguistics.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Commun. ACM*, 51(12):68–74.

Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th International Conference on World Wide Web*, WWW 2004, pages 100–110, New York, NY, USA. Association for Computing Machinery.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open information extraction: The second generation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One*, IJCAI 2011, pages 3–10. AAAI Press.

Manaal Faruqui. 2016. *Diverse Context for Learning Word Representations*. Ph.D. thesis, Carnegie Mellon University.

Michael R. Fellows and Michael A. Langston. 1994. On search, decision, and the efficiency of polynomial-time algorithms. *Journal of Computer and System Sciences*, 49(3):769–779.

John Rupert Firth. 1957. Applications of general linguistics. *Transactions of the Philological Society*, 56(1):1–14.

Marilyn Ford, Joan Bresnan, and Ronald M. Kaplan. 1982. A competence-based theory of syntactic closure. *The Mental Representation of Grammatical Relations*, pages 727–796.

Dayne Freitag and Andrew McCallum. 1999. Information extraction with hmms and shrinkage.

Dayne Freitag and Andrew McCallum. 2000. Information extraction with hmm structures learned by stochastic optimization. In *AAAI/IAAI*.

Karl Friston and Stefan Kiebel. 2009. Predictive coding under the free-energy principle. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 364(1521):1211–1221.

Richard Futrell. 2019. Information-theoretic locality properties of natural language. In *Proceedings of the First Workshop on Quantitative Syntax (Quasy, SyntaxFest 2019)*, pages 2–15, Paris, France. Association for Computational Linguistics.

Richard Futrell, Edward Gibson, and Roger P. Levy. 2020. Lossy-context surprisal: An information-theoretic model of memory effects in sentence processing. *Cognitive Science*, 44(3):e12814.

Richard Futrell and Roger Levy. 2017. Noisy-context surprisal as a human sentence processing cost model. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 688–698, Valencia, Spain. Association for Computational Linguistics.

Richard Futrell and Roger P. Levy. 2019. Do RNNs learn human-like abstract word order preferences? In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 50–59.

Richard Futrell, Kyle Mahowald, and Edward Gibson. 2015. Large-scale evidence of dependency length minimization in 37 languages. *Proceedings of the National Academy of Sciences*, 112(33):10336–10341.

Richard Futrell, Ethan Wilcox, Takashi Morita, Peng Qian, Miguel Ballesteros, and Roger Levy. 2019. Neural language models as psycholinguistic subjects: Representations of syntactic state. 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 32–42, Minneapolis, Minnesota. Association for Computational Linguistics.

Robert Gaizauskas and Yorick Wilks. 1998. Information extraction: beyond document retrieval. *Journal of Documentation*, 54(1):70–105.

Michael R. Garey, Ronald L. Graham, David S. Johnson, and Donald E. Knuth. 1978. Complexity results for bandwidth minimization. *SIAM Journal on Applied Mathematics*, 34(3):477–495.

Michael R. Garey, David S. Johnson, and Larry Stockmeyer. 1974. Some simplified np-complete problems. In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 47–63.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word em-

beddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences*, 115(16):E3635–E3644.

Fanica Gavril. 2011. Some np-complete problems on graphs. In *CISS 2011*.

Dmitriy Genzel. 2010. Automatically learning source-side reordering rules for large scale machine translation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 376–384, Beijing, China. Coling 2010 Organizing Committee.

Sean Gerrish and David M. Blei. 2012. How they vote: Issue-adjusted models of legislative behavior. In *Advances in Neural Information Processing Systems*, pages 2753–2761.

Edward Gibson. 1998. Linguistic complexity: locality of syntactic dependencies. *Cognition*, 68(1):1 – 76.

Edward Gibson. 2000. The dependency locality theory: A distance-based theory of linguistic complexity.

Edward Gibson, Richard Futrell, Steven P. Piantadosi, Isabelle Dautriche, Kyle Mahowald, Leon Bergen, and Roger Levy. 2019. How efficiency shapes human language. *Trends in Cognitive Sciences*, 23(5):389 – 407.

Edward Gibson and Neal J Pearlmutter. 1998. Constraints on sentence comprehension. *Trends in Cognitive Sciences*, 2(7):262 – 268.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288.

Daniel Gildea and David Temperley. 2007. Optimizing grammars for minimum dependency length. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 184–191, Prague, Czech Republic. Association for Computational Linguistics.

Adrià de Gispert, Gonzalo Iglesias, and Bill Byrne. 2015. Fast and accurate preordering for SMT using neural networks. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1012–1017, Denver, Colorado. Association for Computational Linguistics.

Namrata Godbole, Manjunath Srinivasaiah, and Steven Skiena. 2007. Large-scale sentiment analysis for news and blogs (system demonstration). In *ICWSM*.

Mark K. Goldberg and I. A. Klipker. 1976. Minimal placing pf trees on a line.

Yoav Goldberg and Graeme Hirst. 2017. *Neural Network Methods in Natural Language Processing*. Morgan Claypool Publishers.

Carlos Gómez-Rodríguez, Tianze Shi, and Lillian Lee. 2018. Global transition-based non-projective dependency parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2664–2675, Melbourne, Australia. Association for Computational Linguistics.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. The MIT Press.

Prem Gopalan, Jake M. Hofman, and David M. Blei. 2015. Scalable recommendation

with hierarchical poisson factorization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 326–335.

Isao Goto, Masao Utiyama, and Eiichiro Sumita. 2012. Post-ordering by parsing for Japanese-English statistical machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 311–316, Jeju Island, Korea. Association for Computational Linguistics.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Joseph Greenberg. 1963. Some universals of grammar with particular reference to the order of meaningful elements. *In J. Greenberg, ed., Universals of Language. 73-113. Cambridge, MA.*

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences*, 101(suppl 1):5228–5235.

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Daniel Grodner and Edward Gibson. 2005. Consequences of the serial nature of linguistic input for sentenial complexity. *Cognitive science*, 29 2:261–90.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. In *Pro-*

ceedings 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, New Orleans, Louisiana. Association for Computational Linguistics.

Michael Hahn, Judith Degen, Noah D. Goodman, Dan Jurafsky, and Richard Futrell. 2018. An information-theoretic explanation of adjective ordering preferences. *Cognitive Science*.

Michael Hahn and Richard Futrell. 2019. Estimating predictive rate–distortion curves via neural variational inference. *Entropy*, 21(7).

Michael Hahn, Dan Jurafsky, and Richard Futrell. 2020. Universals of word order reflect optimization of grammars for efficient communication. *Proceedings of the National Academy of Sciences*, 117(5):2347–2353.

John Hale. 2001. A probabilistic earley parser as a psycholinguistic model. In *Second Meeting of the North American Chapter of the Association for Computational Linguistics*.

John Hale, Chris Dyer, Adhiguna Kuncoro, and Jonathan Brennan. 2018. Finding syntax in human encephalography with beam search. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2727–2736, Melbourne, Australia. Association for Computational Linguistics.

Frank Harary. 1967. Problem 16. *Theory of Graphs and its Applications*, page 161.

Lawrence H. Harper. 1964. Optimal assignments of numbers to vertices. *Journal of the Society for Industrial and Applied Mathematics*, 12(1):131–135.

Lawrence H. Harper. 1966. Optimal numberings and isoperimetric problems on graphs. *Journal of Combinatorial Theory*, 1(3):385–393.

Zellig S. Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Martin Haspelmath. 1999. Optimality and diachronic adaptation. *Zeitschrift für Sprachwissenschaft*, 18:180 – 205.

J.A. Hawkins. 1983. *Word Order Universals*. Quantitative analyses of linguistic structure. Academic Press.

J.A. Hawkins, S.R. Anderson, J. Bresnan, B. Comrie, W. Dressler, C.J. Ewen, and R. Huddleston. 1994. *A Performance Theory of Order and Constituency*. Cambridge Studies in Linguistics. Cambridge University Press.

John A Hawkins. 1988. *Explaining language universals*. Blackwell.

John A. Hawkins. 1990. A parsing theory of word order universals. *Linguistic Inquiry*, 21(2):223–261.

David G. Hays. 1964. Dependency theory: A formalism and some observations. *Language*, 40(4):511–525.

Jeffrey Heath. 1984. *A Functional Grammar of Nunggubuyu*. Humanities Press / Australian Institute of Aboriginal Studies, Atlantic Highlands N. J. / Canberra.

Julia Hirschberg and Christopher D. Manning. 2015. Advances in natural language processing. *Science*, 349(6245):261–266.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Charles F. Hockett. 1960. The origin of speech. *Scientific American*, 203(3):88–97.

Matthew Hoffman, Francis R. Bach, and David M. Blei. 2010. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pages 856–864.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*.

Sho Hoshino, Hubert Soyer, Yusuke Miyao, and Akiko Aizawa. 2014. Japanese to English machine translation using preordering and compositional distributed semantics. In *Proceedings of the 1st Workshop on Asian Translation (WAT2014)*, pages 55–63, Tokyo, Japan. Workshop on Asian Translation.

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. 2001. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In

Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 168–177.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. 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)*, pages 1681–1691, Beijing, China. Association for Computational Linguistics.

Ray Jackendoff. 2002. *English particle constructions, the lexicon, and the autonomy of syntax*, pages 67 – 94. De Gruyter Mouton, Berlin, Boston.

T. Florian Jaeger and Harry Tily. 2011. On language âĂŸutilityâĂŹ: processing complexity and communicative efficiency. *WIREs Cognitive Science*, 2(3):323–335.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Fred Jelinek. 1976. Continuous speech recognition by statistical methods. *Proceedings of the IEEE*, 64(4):532–556.

Hamed Jelodar, Yongli Wang, Chi Yuan, Xia Feng, Xiahui Jiang, Yanchao Li, and Liang Zhao. 2019. Latent dirichlet allocation (lda) and topic modeling: models, applications, a survey. *Multimedia Tools and Applications*, 78(11):15169–15211.

Thorsten Joachims. 1997. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the 14th International Conference on Machine Learning*.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer.

Michael I. Jordan. 1989. Serial order: A parallel distributed processing approach. *Advances in Connectionist Theory*.

Aravind K. Joshi, Leon S. Levy, and Masako Takahashi. 1975. Tree adjunct grammars. *Journal of Computer and System Sciences*, 10(1):136 – 163.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020a. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. 2020b. The state and fate of linguistic diversity and inclusion in the nlp world. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Dan Jurafsky. 2003. Probabilistic modeling in psycholinguistics: Linguistic comprehension and production. In *PROBABILISTIC LINGUISTICS*, pages 39–96. MIT Press.

Daniel Jurafsky and James H. Martin. 2000. *Speech and Language Processing: An*

Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, 1st edition. Prentice Hall PTR, USA.

Nal Kalchbrenner and Phil Blunsom. 2013. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA. Association for Computational Linguistics.

Ronald M. Kaplan. 2005. A method for tokenizing text. *Inquiries into words, constraints and contexts*, 55.

Ronald M. Kaplan. 2020. Computational psycholinguistics. *Computational Linguistics*, 45(4):607–626.

David R. Karger. 2001. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problem. *SIAM review*, 43(3):499–522.

Jason Katz-Brown, Slav Petrov, Ryan McDonald, Franz Och, David Talbot, Hiroshi Ichikawa, Masakazu Seno, and Hideto Kazawa. 2011. Training a parser for machine translation reordering. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 183–192, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Yuki Kawara, Chenhui Chu, and Yuki Arase. 2018. Recursive neural network based preordering for English-to-Japanese machine translation. In *Proceedings of ACL 2018, Student Research Workshop*, pages 21–27, Melbourne, Australia. Association for Computational Linguistics.

Martin Kay. 1967. Experiments with a powerful parser. In *COLING 1967 Volume 1: Conference Internationale Sur Le Traitement Automatique Des Langues*.

Martin Kay. 1986. Parsing in functional unification grammar. In *Readings in natural language processing*, pages 125–138.

Martin Kay. 1989. Head-driven parsing. In *Proceedings of the First International Workshop on Parsing Technologies*, pages 52–62, Pittsburgh, Pennsylvania, USA. Carnegy Mellon University.

Yoon Kim, Carl Denton, Luong Hoang, and Alexander M. Rush. 2017. Structured attention networks. In *International Conference on Learning Representations*.

Yoon Kim, Alexander Rush, Lei Yu, Adhiguna Kuncoro, Chris Dyer, and Gábor Melis. 2019. Unsupervised recurrent neural network grammars. 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 1105–1117, Minneapolis, Minnesota. Association for Computational Linguistics.

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

Dan Klein and Christopher Manning. 2004. Corpus-based induction of syntactic structure: Models of dependency and constituency. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 478–485, Barcelona, Spain.

Dan Klein and Christopher D. Manning. 2003a. A* parsing: Fast exact Viterbi parse selection. In *Proceedings of the 2003 Human Language Technology Conference*

of the North American Chapter of the Association for Computational Linguistics, pages 119–126.

Dan Klein and Christopher D. Manning. 2003b. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan. Association for Computational Linguistics.

Philipp Koehn. 2010. *Statistical Machine Translation*, 1st edition. Cambridge University Press, USA.

Austin C. Kozlowski, Matt Taddy, and James A. Evans. 2019. The geometry of culture: Analyzing the meanings of class through word embeddings. *American Sociological Review*, 84(5):905–949.

Susumo Kuno. 1973. *The Structure of the Japanese Language*. Massachusetts Institute of Technology Press, Cambridge, Massachusetts.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Raymond Lau, Ronald Rosenfeld, and Salim Roukos. 1993. Adaptive language modeling using the maximum entropy principle. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*.

Moontae Lee, Sungjun Cho, David Bindel, and David Mimno. 2019. Practical correlated topic modeling and analysis via the rectified anchor word algorithm. 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)*, pages 4991–5001, Hong Kong, China. Association for Computational Linguistics.

W Lehnert. 1986. *A Conceptual Theory of Question Answering*, pages 651–657. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Wendy G. Lehnert. 1977a. A conceptual theory of question answering. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence - Volume 1*, pages 158–164, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Wendy Grace Lehnert. 1977b. The process of question answering.

Tao Lei, Yu Zhang, Sida I. Wang, Hui Dai, and Yoav Artzi. 2018. Simple recurrent units for highly parallelizable recurrence. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4470–4481, Brussels, Belgium. Association for Computational Linguistics.

Uri Lerner and Slav Petrov. 2013. Source-side classifier preordering for machine translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 513–523, Seattle, Washington, USA. Association for Computational Linguistics.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix

factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Roger Levy. 2008a. Expectation-based syntactic comprehension. *Cognition*, 106(3):1126 – 1177.

Roger Levy. 2008b. A noisy-channel model of human sentence comprehension under uncertain input. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 234–243, Honolulu, Hawaii. Association for Computational Linguistics.

Roger Levy. 2011. Integrating surprisal and uncertain-input models in online sentence comprehension: formal techniques and empirical results. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1055–1065, Portland, Oregon, USA. Association for Computational Linguistics.

David D. Lewis. 1998. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer.

David D. Lewis and William A. Gale. 1994. A sequential algorithm for training text classifiers. In *SIGIRâĂŹ94*, pages 3–12. Springer.

Charles N. Li and Sandra A. Thompson. 1981. *Mandarin Chinese: a Functional Reference Grammar*. University of California Press, Berkeley.

Chi-Ho Li, Minghui Li, Dongdong Zhang, Mu Li, Ming Zhou, and Yi Guan. 2007. A probabilistic approach to syntax-based reordering for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 720–727, Prague, Czech Republic. Association for Computational Linguistics.

Xiaodong Li, Haoran Xie, Li Chen, Jianping Wang, and Xiaotie Deng. 2014. News impact on stock price return via sentiment analysis. *Knowledge-Based Systems*, 69:14 – 23.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Chu-Cheng Lin, Hao Zhu, Matthew R. Gormley, and Jason Eisner. 2019. Neural finite-state transducers: Beyond rational relations. 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 272–283, Minneapolis, Minnesota. Association for Computational Linguistics.

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. 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.

Bing Liu. 2012. *Sentiment Analysis and Opinion Mining*. Morgan Claypool Publishers.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. 2020. Understanding the difficulty of training transformers.

Weiguo Liu and Anthony Vannelli. 1995. Generating lower bounds for the linear arrangement problem. *Discrete applied mathematics*, 59(2):137–151.

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

Barbara Lohse, Barbara A. Hawkins, and Barbara John A Thomas Wasow. 2004. Domain minimization in english verb-particle constructions. *Language*, 80:238 – 261.

Adam Lopez. 2008. Statistical machine translation. *ACM Computing Surveys (CSUR)*, 40(3):1–49.

Hans Peter Luhn. 1957. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 1(4):309–317.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal. Association for Computational Linguistics.

Fillia Makedon and Ivan Hal Sudborough. 1989. On minimizing width in linear layouts. *Discrete Applied Mathematics*, 23(3):243–265.

Chaitanya Malaviya, Pedro Ferreira, and André F. T. Martins. 2018. Sparse and constrained attention for neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 370–376, Melbourne, Australia. Association for Computational Linguistics.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60, Baltimore, Maryland. Association for Computational Linguistics.

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Cambridge University Press, USA.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA.

Andre Martins and Ramon Astudillo. 2016. From softmax to sparsemax: A sparse model of attention and multi-label classification. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1614–1623, New York, New York, USA. PMLR.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Lan-*

guage Processing, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Andrew McCallum, Dayne Freitag, and Fernando C. N. Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, ICML 2000, pages 591–598, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification. In *AAAI 1998*.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305.

James L. McClelland, Felix Hill, Maja Rudolph, Jason Baldridge, and Hinrich Schütze. 2019. Extending machine language models toward human-level language understanding.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Ryan McDonald and Fernando Pereira. 2006. Online learning of approximate dependency parsing algorithms. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005a. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 523–530, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Pratik Mehta, Anoop Kunchukuttan, and Pushpak Bhattacharyya. 2015. Investigating the potential of post-ordering SMT output to improve translation quality. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 351–356, Trivandrum, India. NLP Association of India.

Gábor Melis, Tomáš Kočiský, and Phil Blunsom. 2020. Mogrifier lstm. In *International Conference on Learning Representations*.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. 2018. Learning latent permutations with gumbel-sinkhorn networks. In *International Conference on Learning Representations*.

Arthur Mensch and Mathieu Blondel. 2018. Differentiable dynamic programming for structured prediction and attention. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3462–3471, Stockholmsmässan, Stockholm Sweden. PMLR.

William Merrill. 2019. Sequential neural networks as automata. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 1–13, Florence. Association for Computational Linguistics.

William Merrill, Gail Weiss, Yoav Goldberg, Roy Schwartz, Noah A. Smith, and Eran Yahav. 2020. A formal hierarchy of rnn architectures. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

George A Miller. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological review*, 63(2):81.

George Armitage Miller. 1951. Language and communication.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Thomas M. Mitchell. 1997. *Machine Learning*, 1 edition. McGraw-Hill, Inc., USA.

Graeme Mitchison and Richard Durbin. 1986. Optimal numberings of an n*n array. *SIAM Journal on Algebraic Discrete Methods*, 7(4):571–582.

Ruslan Mitkov and Aravind K. Joshi. 2012. Tree-adjoining grammars.

Burkhard Monien. 1986. The bandwidth minimization problem for caterpillars with hair length 3 is np-complete. *SIAM Journal on Algebraic Discrete Methods*, 7(4):505–512.

Gereon Müller. 2002. Free word order, morphological case, and sympathy theory. *Resolving Conflicts in Grammars: Optimality Theory in Syntax, Morphology, and Phonology*.

Petra Mutzel. 1995. A polyhedral approach to planar augmentation and related problems. In *European Symposium on Algorithms*, pages 494–507. Springer.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gùlçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Roberto Navigli and Simone Paolo Ponzetto. 2012. Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Pandu Nayak. 2019. Understanding searches better than ever before. Technical report, Google.

Graham Neubig, Taro Watanabe, and Shinsuke Mori. 2012. Inducing a discriminative parser to optimize machine translation reordering. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computa-*

tional Natural Language Learning, pages 843–853, Jeju Island, Korea. Association for Computational Linguistics.

David J. Newman and Sharon Block. 2006. Probabilistic topic decomposition of an eighteenth-century american newspaper. *Journal of the American Society for Information Science and Technology*, 57(6):753–767.

Vlad Niculae. 2018. *Learning Deep Models with Linguistically-Inspired Structure*. Ph.D. thesis, Cornell University.

Vlad Niculae and Mathieu Blondel. 2017. A regularized framework for sparse and structured neural attention. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3338–3348. Curran Associates, Inc.

Joakim Nivre. 2005. Dependency grammar and dependency parsing.

Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, and Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666, Portorož, Slovenia. European Language Resources Association (ELRA).

János Pach, Farhad Shahrokhi, and Mario Szegedy. 1996. Applications of the crossing number. *Algorithmica*, 16(1):111–117.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The

pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278, Barcelona, Spain.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan. Association for Computational Linguistics.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1–2):1–135.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 79–86. Association for Computational Linguistics.

Christos H Papadimitriou. 1976. The np-completeness of the bandwidth minimization problem. *Computing*, 16(3):263–270.

Fuchun Peng, Fangfang Feng, and Andrew McCallum. 2004. Chinese segmentation and new word detection using conditional random fields. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 562–568, Geneva, Switzerland. COLING.

Hao Peng, Roy Schwartz, Sam Thomson, and Noah A. Smith. 2018. Rational recurrences. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1214, Brussels, Belgium. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Fernando C. N. Pereira and David H. D. Warren. 1983. Parsing as deduction. In *21st Annual Meeting of the Association for Computational Linguistics*, pages 137–144, Cambridge, Massachusetts, USA. Association for Computational Linguistics.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2018a. Interpretable structure induction via sparse attention. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 365–367, Brussels, Belgium. Association for Computational Linguistics.

Ben Peters, Vlad Niculae, and André F. T. Martins. 2019a. Sparse sequence-to-sequence models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1504–1519, Florence, Italy. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018b. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the*

Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019b. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14, Florence, Italy. Association for Computational Linguistics.

Jordi Petit and Jordi Girona Salgado. 1998. *Approximation heuristics and benchmarkings for the MinLA problem*.

Steven Pinker. 2003. *The language instinct: How the mind creates language*. Penguin UK.

Steven Pinker. 2007. *The stuff of thought: Language as a window into human nature*. Penguin.

Steven Pinker and Paul Bloom. 1990. Natural language and natural selection. *Behavioral and Brain Sciences*, 13(4):707–727.

Steven Pinker and Ray Jackendoff. 2005. The faculty of language: what's special about it? *Cognition*, 95(2):201 – 236.

David Pinto, Andrew McCallum, Xing Wei, and W. Bruce Croft. 2003. Table extraction using conditional random fields. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR 2003, pages 235–242, New York, NY, USA. Association for Computing Machinery.

Jakub Piskorski and Roman Yangarber. 2013. *Information Extraction: Past, Present and Future*, pages 23–49. Springer Berlin Heidelberg, Berlin, Heidelberg.

Grusha Prasad, Marten van Schijndel, and Tal Linzen. 2019. Using priming to uncover the organization of syntactic representations in neural language models. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 66–76, Hong Kong, China. Association for Computational Linguistics.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. 2020. Stanza: A python natural language processing toolkit for many human languages. *ArXiv*, abs/2003.07082.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer.

Mohammad Sadegh Rasooli and Michael Collins. 2019. Low-resource syntactic transfer with unsupervised source reordering. 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 3845–3856, Minneapolis, Minnesota. Association for Computational Linguistics.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Conference on Empirical Methods in Natural Language Processing*.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Shauli Ravfogel, Yoav Goldberg, and Tal Linzen. 2019. Studying the inductive biases of RNNs with synthetic variations of natural languages. 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 3532–3542, Minneapolis, Minnesota. Association for Computational Linguistics.

Kumar Ravi and Vadlamani Ravi. 2015. A survey on opinion mining and sentiment analysis: Tasks, approaches and applications. *Knowledge-Based Systems*, 89:14 – 46.

Ramamurthy Ravi, Ajit Agrawal, and Philip Klein. 1991. Ordering problems approximated: single-processor scheduling and interval graph completion. In *International Colloquium on Automata, Languages, and Programming*, pages 751–762. Springer.

Jeffrey C. Reynar and Adwait Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Fifth Conference on Applied Natural Language Processing*, pages 16–19, Washington, DC, USA. Association for Computational Linguistics.

Jan Rijkhoff. 1990. Explaining word order in the noun phrase. *Linguistics*, 28(1):5 – 42.

Brian Roark, Murat Saraclar, Michael Collins, and Mark Johnson. 2004. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 47–54, Barcelona, Spain.

Joseph Rocchio. 1971. Relevance feedback in information retrieval. *The Smart retrieval system-experiments in automatic document processing*, pages 313–323.

Sebastian Ruder. 2019. *Neural Transfer Learning for Natural Language Processing*. Ph.D. thesis, National University of Ireland, Galway.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 379–389, Lisbon, Portugal. Association for Computational Linguistics.

Yousef Saad. 2003. *Iterative Methods for Sparse Linear Systems*, second edition. Society for Industrial and Applied Mathematics.

I.A. Sag, T. Wasow, and E.M. Bender. 2003. *Syntactic Theory: A Formal Introduction*. CSLI lecture notes. Center for the Study of Language and Information.

Gerald Salton. 1971. *The SMART Retrieval System — Experiments in Automatic Document Processing*. Prentice-Hall, Inc., USA.

Gerald Salton, A. Wong, and C. S. Yang. 1975. A vector space model for automatic indexing. *Commun. ACM*, 18(11):613–620.

Gerard Salton. 1968. *Automatic Information Organization and Retrieval.* McGraw Hill Text.

Gerard Salton. 1975. *Theory of Indexing*. Society for Industrial and Applied Mathematics, USA.

Gerard Salton. 1991. Developments in automatic text retrieval. *science*, 253(5023):974–980.

Gerard Salton and Michael J McGill. 1986. Introduction to modern information retrieval.

Rodrigo Santa Cruz, Basura Fernando, Anoop Cherian, and Stephen Gould. 2019. Visual permutation learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3100–3114.

Aaron Schein, John Paisley, David M. Blei, and Hanna Wallach. 2015. Bayesian poisson tensor factorization for inferring multilateral relations from sparse dyadic event counts. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1045–1054.

Aaron Schein, Zhiwei Steven Wu, Alexandra Schofield, Mingyuan Zhou, and Hanna Wallach. 2019. Locally private Bayesian inference for count models. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5638–5648, Long Beach, California, USA. PMLR.

Marten van Schijndel and Tal Linzen. 2018. A neural model of adaptation in reading. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4704–4710, Brussels, Belgium. Association for Computational Linguistics.

Marten van Schijndel and Tal Linzen. 2019. Can entropy explain successor surprisal effects in reading? In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 1–7.

Marten van Schijndel, Aaron Mueller, and Tal Linzen. 2019. Quantity doesn't buy quality syntax with neural language 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)*, pages 5831–5837, Hong Kong, China. Association for Computational Linguistics.

Alexandra Schofield. 2019. *Text Processing for the Effective Application of Latent Dirichlet Allocation*. Ph.D. thesis, Cornell University.

Mike Schuster and Kuldip K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681.

Roy Schwartz, Sam Thomson, and Noah A. Smith. 2018. Bridging CNNs, RNNs, and weighted finite-state machines. In *Proceedings of the 56th Annual Meeting of the*

Association for Computational Linguistics (Volume 1: Long Papers), pages 295–305, Melbourne, Australia. Association for Computational Linguistics.

Sofia Serrano and Noah A. Smith. 2019. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy. Association for Computational Linguistics.

Fei Sha and Fernando Pereira. 2003. Shallow parsing with conditional random fields. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 213–220.

Claude E. Shannon. 1948. A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423.

Claude E. Shannon and Warren Weaver. 1963. *The Mathematical Theory of Communication*. pt. 11. University of Illinois Press.

Vatsal Sharan, Sham M. Kakade, Percy S. Liang, and Gregory Valiant. 2017. Learning overcomplete hmms. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 940–949. Curran Associates, Inc.

Vatsal Sharan, Sham M. Kakade, Percy S. Liang, and Gregory Valiant. 2018. Prediction with a short memory. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 1074–1087, New York, NY, USA. Association for Computing Machinery.

Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. 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 464–468, New Orleans, Louisiana. Association for Computational Linguistics.

Tianze Shi, Liang Huang, and Lillian Lee. 2017a. Fast(er) exact decoding and global training for transition-based dependency parsing via a minimal feature set. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 12–23, Copenhagen, Denmark. Association for Computational Linguistics.

Tianze Shi and Lillian Lee. 2018. Valency-augmented dependency parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1277–1291, Brussels, Belgium. Association for Computational Linguistics.

Tianze Shi and Lillian Lee. 2020. Extracting headless mwes from dependency parse trees: Parsing, tagging, and joint modeling approaches. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, Washington. Association for Computational Linguistics.

Tianze Shi, Felix G. Wu, Xilun Chen, and Yao Cheng. 2017b. Combining global models for parsing universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 31–39, Vancouver, Canada. Association for Computational Linguistics.

Yossi Shiloach. 1979. A minimum linear arrangement algorithm for undirected trees. *SIAM Journal on Computing*, 8(1):15–32.

Amit Singhal. 2005. Challenges in running a commercial search engine. In *SIGIR*, page 432.

Amit Singhal, Steve Abney, Michiel Bacchiani, Michael Collins, Donald Hindle, and Fernando Pereira. 2000. At&t at trec-8.

Richard Sinkhorn. 1964. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879.

Richard Sinkhorn and Paul Knopp. 1967. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348.

Nathaniel J. Smith and Roger Levy. 2013. The effect of word predictability on reading time is logarithmic. *Cognition*, 128(3):302 – 319.

Noah A. Smith and Jason Eisner. 2006. Annealing structural bias in multilingual weighted grammar induction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 569–576, Sydney, Australia. Association for Computational Linguistics.

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

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013b. Recursive deep models for semantic

compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 60(5):493–502.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Mark Steedman. 1987. Combinatory grammars and parasitic gaps. *Natural language and linguistic theory*.

Mark Steedman. 2004. The syntactic process. In *Language, speech, and communication*.

Mark Steedman. 2008. On becoming a discipline. *Computational Linguistics*, 34(1):137–144.

Julia Strout, Ye Zhang, and Raymond Mooney. 2019. Do human rationales improve machine explanations? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–62, Florence, Italy. Association for Computational Linguistics.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Florence, Italy. Association for Computational Linguistics.

Katsuhito Sudoh, Xianchao Wu, Kevin Duh, Hajime Tsukada, and Masaaki Nagata. 2011. Post-ordering in statistical machine translation. In *Proceedings of the Machine Translation Summit*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Charles Sutton and Andrew McCallum. 2012. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, 4(4):267–373.

Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. 2007. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *The journal of machine learning research*, 8:693–723.

Mirac Suzgun, Yonatan Belinkov, Stuart Shieber, and Sebastian Gehrmann. 2019a. LSTM networks can perform dynamic counting. In *Proceedings of the Workshop on Deep Learning and Formal Languages: Building Bridges*, pages 44–54, Florence. Association for Computational Linguistics.

Mirac Suzgun, Yonatan Belinkov, and Stuart M. Shieber. 2019b. On evaluating the generalization of LSTM models in formal languages. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 277–286.

Michael K Tanenhaus and John C Trueswell. 1995. Sentence comprehension.

Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Christopher D. Manning. 2004. Max-margin parsing. In *Proceedings of the 2004 Conference on Empirical*

Methods in Natural Language Processing, pages 1–8, Barcelona, Spain. Association for Computational Linguistics.

David Temperley and Daniel Gildea. 2018. Minimizing syntactic dependency lengths: Typological/cognitive universal? *Annual Review of Linguistics*, 4(1):67–80.

Lucien Tesnière. 1959. Les éléments de syntaxe structurale.

Kristina Toutanova, H. Tolga Ilhan, and Christopher Manning. 2002. Extentions to HMM-based statistical word alignment models. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 87–94. Association for Computational Linguistics.

Kristina Toutanova and Christopher D. Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, pages 63–70, Hong Kong, China. Association for Computational Linguistics.

Roy Tromble and Jason Eisner. 2009. Learning linear ordering problems for better translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1007–1016, Singapore. Association for Computational Linguistics.

Peter Turney. 2002. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 417–424, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

William E. Underwood. 2012. What can topic models of pmla teach us about the history of literary scholarship? *Journal of Digital Humanities*, 2(1).

Clara Vania, Yova Kementchedjhieva, Anders Søgaard, and Adam Lopez. 2019. A systematic comparison of methods for low-resource dependency parsing on genuinely low-resource languages. 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)*, pages 1105–1116, Hong Kong, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Noortje J. Venhuizen, Matthew W. Crocker, and Harm Brouwer. 2019. Expectation-based comprehension: Modeling the interaction of world knowledge and linguistic experience. *Discourse Processes*, 56(3):229–255.

Oriol Vinyals, Samy Bengio, and Manjunath Kudlur. 2016. Order matters: Sequence to sequence for sets. In *International Conference on Learning Representations*.

Karthik Visweswariah, Rajakrishnan Rajkumar, Ankur Gandhe, Ananthakrishnan Ramanathan, and Jiri Navratil. 2011. A word reordering model for improved machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 486–496, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Ellen M. Voorhees and Donna K. Harman. 2000a. The eighth text retrieval conference (trec-8). Technical report.

Ellen M. Voorhees and Donna K. Harman. 2000b. The ninth text retrieval conference (trec-9). Technical report.

Ellen M. Voorhees and Donna K. Harman. 2001. The tenth text retrieval conference (trec-10). Technical report.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, pages 3261–3275.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019b. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Benyou Wang, Donghao Zhao, Christina Lioma, Qiuchi Li, Peng Zhang, and Jakob Grue Simonsen. 2020. Encoding word order in complex embeddings. In *International Conference on Learning Representations*.

Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C.-C. Jay Kuo. 2019c. Evaluating word embedding models: methods and experimental results. *APSIPA Transactions on Signal and Information Processing*, 8:e19.

Chao Wang, Michael Collins, and Philipp Koehn. 2007. Chinese syntactic reordering for statistical machine translation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language*

Learning (EMNLP-CoNLL), pages 737–745, Prague, Czech Republic. Association for Computational Linguistics.

Dingquan Wang and Jason Eisner. 2016. The galactic dependencies treebanks: Getting more data by synthesizing new languages. *Transactions of the Association for Computational Linguistics*, 4:491–505.

Dingquan Wang and Jason Eisner. 2018. Synthetic data made to order: The case of parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1325–1337, Brussels, Belgium. Association for Computational Linguistics.

Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc V. Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. 2017. Tacotron: Towards end-to-end speech synthesis. In *INTERSPEECH*.

Warren Weaver. 1949/55. Translation. In William N. Locke and A. Donald Boothe, editors, *Machine Translation of Languages*, pages 15–23. MIT Press, Cambridge, MA. Reprinted from a memorandum written by Weaver in 1949.

E. M. Helen Weir. 1994. Nadëb. In Peter Kahrel and RenÃľ van den Berg, editors, *Typological Studies in Negation*, pages 291–323. John Benjamins, Amsterdam.

Gail Weiss, Yoav Goldberg, and Eran Yahav. 2018. On the practical computational power of finite precision RNNs for language recognition. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short*

Papers), pages 740–745, Melbourne, Australia. Association for Computational Linguistics.

Sarah Wiegreffe and Yuval Pinter. 2019. Attention is not not explanation. 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)*, pages 11–20, Hong Kong, China. Association for Computational Linguistics.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *International Conference on Learning Representations*.

Ethan Wilcox, Roger Levy, and Richard Futrell. 2019. Hierarchical representation in neural language models: Suppression and recovery of expectations. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 181–190, Florence, Italy. Association for Computational Linguistics.

Ethan Wilcox, Roger Levy, Takashi Morita, and Richard Futrell. 2018. What do RNN language models learn about filler–gap dependencies? In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 211–221, Brussels, Belgium. Association for Computational Linguistics.

Yorick Wilks. 1997. Information extraction as a core language technology. In *Information Extraction A Multidisciplinary Approach to an Emerging Information Technology*, pages 1–9, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv*, abs/1609.08144.

Fei Xia and Michael McCord. 2004. Improving a statistical MT system with automatically learned rewrite patterns. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 508–514, Geneva, Switzerland. COLING.

Frank Z. Xing, Erik Cambria, and Roy E. Welsch. 2017. Natural language based financial forecasting: a survey. *Artificial Intelligence Review*, 50:49–73.

Peng Xu, Jaeho Kang, Michael Ringgaard, and Franz Och. 2009. Using a dependency parser to improve SMT for subject-object-verb languages. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 245–253, Boulder, Colorado. Association for Computational Linguistics.

Yilun Xu, Shengjia Zhao, Jiaming Song, Russell Stewart, and Stefano Ermon. 2020. A theory of usable information under computational constraints. In *International Conference on Learning Representations*.

Hiroko Yamashita and Franklin Chang. 2001. âĂIJlong before shortâĂİ preference in the production of a head-final language. *Cognition*, 81(2):B45 – B55.

Yiming Yang. 1999. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1-2):69–90.

Yiming Yang and Xin Liu. 1999. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49.

Yiming Yang and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 33*. Curran Associates, Inc.

Mihalis Yannakakis. 1985. A polynomial algorithm for the min-cut linear arrangement of trees. *Journal ACM*, 32(4):950–988.

Qiang Ye, Ziqiong Zhang, and Rob Law. 2009. Sentiment classification of online reviews to travel destinations by supervised machine learning approaches. *Expert Systems with Applications*, 36(3, Part 2):6527 – 6535.

Dong Yu and Li Deng. 2014. *Automatic Speech Recognition: A Deep Learning Approach*. Springer Publishing Company, Incorporated.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. 2017. Position-aware attention and supervised data improve slot filling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 35–45, Copenhagen, Denmark. Association for Computational Linguistics.

Vitalii Zhelezniak, Aleksandar Savkov, April Shen, Francesco Moramarco, Jack Flann, and Nils Y. Hammerla. 2019. Don't settle for average, go for the max: Fuzzy sets and max-pooled word vectors. In *International Conference on Learning Representations*.

APPENDIX A

REPRODUCIBILITY

In this appendix, we provide further details required to exactly reproduce this work. Particularly significant is that we release the code (§A.2) for running all experiments and generating all tables/figures/visualizations used in this work. We adhere to the guidelines presented in Dodge et al. (2019), which were further extended in the EMNLP 2020 reproducibility guidelines[1], to provide strong and rigorous guarantees on the reproducibility of our work.

A.1 Additional Experimental Details

We use Python 3.6.9 throughout this work along with PyTorch 1.5.0.

Tokenization and Dependency Parsing. Tokenization is done using the English `en_core_web_lg` model released in `spaCy` version 2.2.4. Dependency parsing is done using the same English `en_core_web_lg` model released in `spaCy` version 2.2.4. The model is 789MB and is trained on OntoNotes 5 using a multi-task[2] convolutional neural network-based model with pretrained word embeddings initialized using GloVe.[3]

[1] https://2020.emnlp.org/call-for-papers
[2] The other tasks are part-of-speech tagging and named entity recognition.
[3] The GloVe embeddings are trained on Common Crawl data.

Data Preprocessing. Beyond tokenizing the data, we do no further pre-processing except removing ill-formed examples in any datasets (where there is an input and no label or vice versa). We find that there are 4 such examples in the `CR` dataset and none in any of the other four datasets.

Pretrained Representations. We use pretrained ELMo representations that are are obtained by using data tokenized using `spaCy` as described previously. The exact pretrained ELMo encoders are available here[4] and concatenate the representations from each of the two layers (yielding 2048-dimensional vectors).

Randomness. We fix the Python and PyTorch random seeds to be random seed 0.

Reverse Cuthill-McKee. We use the implementation of the algorithm provided in `SciPy` 1.4.1.

[4] https://s3-us-west-2.amazonaws.com/allennlp/models/elmo/2x4096_512_2048cnn_2xhighway/elmo_2x4096_512_2048cnn_2xhighway_weights.hdf5; file name describes model parameters under the AllenNLP naming conventions.

A.2 Code Release

All code for this work is made publicly available. The code is hosted at https://github.com/rishibommasani/MastersThesis. We note that we provide documentation for most core functionality and clarifiications can be provided upon request.

A.3 Data Access

All data used in this work is publicly available. The copies of the datasets we use are available at https://github.com/harvardnlp/sent-conv-torch/tree/master/data via the Harvard NLP group. The data can also be accessed from the corresponding websites for each of the datasets:

- CR — https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#datasets
  Hosted by Bing Liu.
- SUBJ — https://www.cs.cornell.edu/people/pabo/movie-review-data/
  Hosted by Lillian Lee.
- SST-2 — https://nlp.stanford.edu/sentiment/
  Hosted by Stanford NLP group.

- SST-5 — https://nlp.stanford.edu/sentiment/

  Hosted by Stanford NLP group.

- TREC — https://cogcomp.seas.upenn.edu/Data/QA/QC/

  Hosted by Dan Roth and UPenn CogComp group.

A.4 Contact Information

Questions, concerns, and errata should be directed to the thesis author at any of:

- nlprishi@stanford.edu
- rb724@cornell.edu
- rishibommasani@gmail.com

Any and all remaining errors in this thesis are strictly due to the author.

APPENDIX B

ADDITIONAL RESULTS

In this appendix, we provide further results that were not included in the main body of the thesis. We first provide the results for all hyperparameters with the stopping condition we used in the main body of thesis: stopping after the fixed threshold of 12 epochs. These results appear in Tables B.1–B.6. For further completeness, we provide the results for all hyperparameters with the stopping condition after which we never saw any improvements (for all models, datasets, and orders): stopping after the fixed threshold of 15 epochs. These results appear in Tables B.7–B.12.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.833	0.95	0.87	0.464	0.95
r_r	0.823	0.947	0.87	0.442	0.944
r_b	0.839	0.941	0.867	0.452	0.962
r_c	0.852	0.95	0.873	0.453	0.952
r_m	0.836	0.946	0.862	0.456	0.948
$\tilde{r}_b$	0.839	0.945	0.879	0.478	0.954
$\tilde{r}_c$	0.833	0.952	0.875	0.464	0.952
$\tilde{r}_m$	0.841	0.958	0.876	0.448	0.954

Table B.1: Full classification results for $h = 32, p = 0.0$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.833	0.955	0.882	0.481	0.956
r_r	0.840	0.945	0.864	0.458	0.954
r_b	0.854	0.948	0.865	0.481	0.958
r_c	0.831	0.942	0.871	0.478	0.95
r_m	0.831	0.95	0.864	0.464	0.958
$\tilde{r}_b$	0.839	0.946	0.866	0.46	0.952
$\tilde{r}_c$	0.849	0.956	0.873	0.46	0.958
$\tilde{r}_m$	0.831	0.949	0.868	0.457	0.962

Table B.2: Full classification results for $h = 64, p = 0.02$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.783	0.95	0.868	0.477	0.956
r_r	0.823	0.94	0.876	0.446	0.952
r_b	0.786	0.937	0.869	0.471	0.962
r_c	0.852	0.946	0.861	0.474	0.948
r_m	0.836	0.918	0.868	0.456	0.95
$\tilde{r}_b$	0.841	0.947	0.864	0.478	0.956
$\tilde{r}_c$	0.844	0.946	0.87	0.47	0.952
$\tilde{r}_m$	0.825	0.948	0.872	0.461	0.96

Table B.3: Full classification results for $h = 64, p = 0.2$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.852	0.951	0.884	0.485	0.946
r_r	0.840	0.95	0.877	0.476	0.952
r_b	0.841	0.95	0.873	0.481	0.956
r_c	0.86	0.953	0.874	0.481	0.954
r_m	0.836	0.951	0.874	0.482	0.962
$\tilde{r}_b$	0.847	0.947	0.882	0.469	0.95
$\tilde{r}_c$	0.847	0.953	0.871	0.494	0.962
$\tilde{r}_m$	0.828	0.951	0.876	0.467	0.962

Table B.4: Full classification results for $h = 128, p = 0.02$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.847	0.945	0.874	0.455	0.952
r_r	0.839	0.942	0.851	0.419	0.952
r_b	0.825	0.948	0.865	0.445	0.95
r_c	0.852	0.944	0.867	0.469	0.954
r_m	0.817	0.942	0.87	0.441	0.954
$\tilde{r}_b$	0.831	0.943	0.874	0.468	0.948
$\tilde{r}_c$	0.849	0.948	0.873	0.462	0.952
$\tilde{r}_m$	0.844	0.933	0.863	0.473	0.958

Table B.5: Full classification results for $h = 128, p = 0.2$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.836	0.951	0.896	0.476	0.962
r_r	0.842	0.934	0.864	0.451	0.952
r_b	0.825	0.952	0.87	0.462	0.966
r_c	0.836	0.946	0.864	0.462	0.958
r_m	0.841	0.951	0.859	0.461	0.96
$\tilde{r}_b$	0.852	0.949	0.874	0.461	0.956
$\tilde{r}_c$	0.825	0.932	0.874	0.467	0.968
$\tilde{r}_m$	0.841	0.941	0.86	0.476	0.958

Table B.6: Full classification results for $h = 256, p = 0.2$. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.841	0.948	0.867	0.462	0.944
r_r	0.825	0.945	0.87	0.443	0.942
r_b	0.841	0.94	0.864	0.45	0.932
r_c	0.844	0.948	0.87	0.462	0.948
r_m	0.836	0.946	0.858	0.457	0.954
$\tilde{r}_b$	0.833	0.941	0.877	0.469	0.952
$\tilde{r}_c$	0.828	0.951	0.873	0.466	0.95
$\tilde{r}_m$	0.844	0.953	0.877	0.456	0.948

Table B.7: Full classification results for $h = 32, p = 0.0$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.833	0.954	0.882	0.43	0.956
r_r	0.841	0.945	0.863	0.459	0.95
r_b	0.852	0.95	0.871	0.439	0.956
r_c	0.836	0.944	0.871	0.468	0.932
r_m	0.831	0.95	0.862	0.463	0.958
$\tilde{r}_b$	0.847	0.948	0.864	0.459	0.95
$\tilde{r}_c$	0.847	0.955	0.873	0.462	0.958
$\tilde{r}_m$	0.833	0.949	0.865	0.461	0.96

Table B.8: Full classification results for $h = 64, p = 0.02$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.823	0.951	0.874	0.458	0.956
r_r	0.82	0.945	0.861	0.455	0.954
r_b	0.844	0.947	0.87	0.455	0.948
r_c	0.847	0.947	0.871	0.465	0.964
r_m	0.839	0.938	0.867	0.461	0.96
$\tilde{r}_b$	0.849	0.946	0.855	0.47	0.958
$\tilde{r}_c$	0.836	0.957	0.855	0.47	0.948
$\tilde{r}_m$	0.825	0.952	0.874	0.465	0.958

Table B.9: Full classification results for $h = 64, p = 0.2$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.852	0.949	0.883	0.49	0.966
r_r	0.833	0.949	0.876	0.466	0.946
r_b	0.844	0.948	0.873	0.477	0.944
r_c	0.854	0.953	0.874	0.474	0.958
r_m	0.839	0.95	0.87	0.452	0.916
$\tilde{r}_b$	0.847	0.947	0.878	0.422	0.95
$\tilde{r}_c$	0.849	0.951	0.873	0.471	0.96
$\tilde{r}_m$	0.831	0.952	0.875	0.451	0.958

Table B.10: Full classification results for $h = 128, p = 0.02$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.844	0.944	0.868	0.467	0.96
r_r	0.852	0.941	0.869	0.435	0.948
r_b	0.828	0.945	0.864	0.448	0.954
r_c	0.857	0.917	0.87	0.462	0.96
r_m	0.817	0.938	0.865	0.461	0.954
$\tilde{r}_b$	0.82	0.942	0.871	0.466	0.944
$\tilde{r}_c$	0.849	0.953	0.845	0.482	0.956
$\tilde{r}_m$	0.847	0.951	0.868	0.45	0.958

Table B.11: Full classification results for $h = 128, p = 0.2$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.

	CR	SUBJ	SST-2	SST-5	TREC
r_I	0.841	0.953	0.891	0.471	0.968
r_r	0.844	0.945	0.873	0.462	0.948
r_b	0.825	0.955	0.841	0.455	0.96
r_c	0.839	0.947	0.871	0.475	0.952
r_m	0.839	0.953	0.868	0.483	0.956
$\tilde{r}_b$	0.849	0.95	0.861	0.466	0.956
$\tilde{r}_c$	0.823	0.947	0.87	0.474	0.968
$\tilde{r}_m$	0.839	0.947	0.87	0.471	0.956

Table B.12: Full classification results for $h = 256, p = 0.2$. Results are reported for models after they were trained for 15 epochs. Results use `pretrain-permute-finetune` framework with the order specified in each row. All other hyperparameters are set as described previously. The top part of the table refers to baselines. The middle part of the table refers to orders derived from pure optimization algorithms. The bottom part of the table refers to orders derived from heuristic algorithms we introduce using `Transposition Monte Carlo`.