

March 25,2022

Thesis Proposal
**Continued Relevance of the Classical NLG
Pipeline As A Guide To Motivate
Interventions On End-To-End NLG Models**

Varun Prashant Gangal

March 25, 2022

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15123

Thesis Committee:

Eduard Hovy (Chair)	Carnegie Mellon University
Alan Black	Carnegie Mellon University
David Mortensen	Carnegie Mellon University
Sebastian Gehrmann	Google Research
Joel Tetreault	DataMinR

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

Copyright © 2021 Varun Prashant Gangal

Keywords: Natural Language Generation, Microplanning, Macroplanning, Surface Realization, Style, Narrative

Abstract

Natural Language Generation (NLG) is the field of study which aims to endow agents with the ability to generate language to satisfy any stated communicative goal (CG). Today, NLG systems that converse (e.g., Meena) and co-author (e.g., Gmail SmartCompose) with humans aren't just deployable, but an accepted part of net-equipped societies. Notwithstanding the rapid strides in NLG research, several aspects of model outputs that require further research scrutiny, *inter alia*, remain:

- Enhancing commonsense plausibility of generated outputs, that existing research [110] has found to be deficient.
- Ensuring discourse coherence, both in terms of local discourse structure [149] and wider multi-sentence discourse structure [97].
- Controlling generated text to adhere to given constraints (e.g., lexical choice preferences to control for lexical styles, or avoid swear words [1, 65]).
- Incorporating domain-specific pragmatic or commonsense knowledge for fine-tuning to low resource domains.

To analyze and address these shortcomings, one has to explore a task setting focussing on the aspect, and then formulate an approach to either improve the learning or the evaluation with respect to it. Since NLG is a complex, multipart operation, it can become challenging to disentangle and isolate out individual aspects. We claim in this thesis that a productive approach is to follow the traditional decomposition of NLG into a 3-stage pipeline, as we describe next, since it facilitates separation of aspects by their principal contribution to the process.

Models underlying today's systems e.g., T5 [173], based on neural architectures like Transformers [227], are "end-to-end" in structure and learning. They neither produce intermediate symbolic outputs, nor split into stages learnable independently. In contrast, NLG was traditionally seen as a composite sequence of subtasks, with typical models being pipelines of stages. In 2000, Reiter & Dale presented a consensus stagewise architecture — *macroplanning* (discourse planning), *microplanning* (sentence planning) and *surface realization*, which we call the *Classical NLG pipeline*, or CNP. This pipeline motivates our 3-stage decomposition, which allows focus on a small set of aspects while holding others constant. Such focusing is a powerful methodology for NLG research. Herein, we exemplify how it helps us contribute to each aspect, via four contribution types:

1. New tasks isolating the aspect, denoted by [Task].
2. Improve existing evaluation process for the aspect, denoted by [Evaluation].
3. Use knowledge external to CG to learn models better for the aspect, denoted by [Knowledge].
4. Improve Learning Model/Process. We denote this by [Method].

The culmination of this work is a set of algorithms, approaches, and evaluation techniques that contribute to the aspect-wise evolution of NN-based NLG, leading to richer and more appropriate NLG output.

Contents

1	Introduction	1
1.1	Natural Language Generation	1
1.2	Contributions & Structure	16
1.3	Projected Timeline	22
I	Realization	23
2	Lexico-Phonetic Surface Realization and Portmanteaus	
	(EMNLP 2017)	
	[Method]	25
2.1	Introduction	27
2.2	Related Work	29
2.3	Models	29
2.4	Making Predictions	31
2.5	Dataset	31
2.6	Baseline	32
2.7	Experiments	33
2.8	Conclusion	37
3	Stylistic Surface Transduction To Shakespearize Modern English	
	(EMNLP 2017'WS)	
	[Knowledge][Method]	41
3.1	Introduction	43
3.2	Dataset	45
3.3	Method Overview	47

3.4	Token embeddings	47
3.5	Method Description	49
3.6	Loss functions	52
3.7	Experiments	52
3.8	Results	54
3.9	Related Work	57
3.10	Conclusion	58
4	Tongue Twister Generation	
	(Proposed)	
	[New Task]	61
4.1	Introduction	61
4.2	Background	62
4.3	Representational Primitives	63
4.4	Dataset	64
4.5	Conclusion	64
II	Microplanning	65
5	Improving Realization Level Metric Evaluation of Open Dialog via Microplanning Rollouts for Reference Augmentation	
	(Findings of ACL 2021)	
	[Evaluation][Knowledge]	67
5.1	Introduction	69
5.2	Method	71
5.3	Experiments	74
5.4	Discussion	76
5.5	Related Work	76
5.6	Conclusion	77
5.7	Additional Results	79
5.8	Additional Details on Context Adaptation	80
5.9	Retrieval Similarity Function - Details	82
5.10	Qualitative Examples	83
5.11	Human Evaluation Details	83

5.12 Computing Details	84
6 VisCTG: Improving Plausibility of Microplanning for Concept-To-Text Generation Through Retrieve-Caption-Generate (AAAI 2022)	
[Method][Knowledge]	85
6.1 Introduction	88
6.2 Dataset, Models, and Metrics	90
6.3 Initial Analysis and Motivation	92
6.4 Methodology	93
6.5 Experiments	95
6.6 Results and Analysis	96
6.7 Related Work	101
6.8 Conclusion and Future Work	102
6.9 Appendices	105
6.10 Full Re-implementation versus Reported Model Numbers	105
6.11 Pretrained FC Image Captioning Model Details	106
6.12 BART and T5 Model Training and Generation Details	107
6.13 Human Evaluation Details	108
6.14 Further Qualitative Examples	109
7 Better Microplans For Concept-to-Text Tasks By Self Introspecting Input Expansion (Under Preparation)	
[Method]	111
7.1 Introduction	111
7.2 Proposed Approach	113
7.3 Intended Conclusion	115
III Macroplanning	117
8 Viable Content Selection and Refex Generation Through Pragmatic Backoff For Chess Commentary Generation	

(ACL 2018)	
[Knowledge][New Task]	119
8.1 Introduction	122
8.2 Chess Commentary Dataset	124
8.3 Game Aware Neural Commentary Generations (GAC)	126
8.4 Experiments	130
8.5 Related Work	136
8.6 Conclusions	137
9 Macro-Level Controllable Generation based on Elements of Narrativity: The Narrative Reordering Problem	
(AAAI 2022)	
[New Task][Evaluation]	151
9.1 Introduction	152
9.2 Dataset: NAREORC	156
9.3 Methodology	158
9.4 Experiments	161
9.5 Results and Analysis	162
9.6 Applications of NAREOR	167
9.7 Related Work	168
9.8 Conclusion and Future Work	168
9.9 Discussion: NAREOR as a Hard Instance of Controllable Generation	170
9.10 NAREORC Annotation Details	170
9.11 Further Model Finetuning and Generation Details	171
9.12 Human Evaluation Study Details	173
9.13 Further Qualitative Examples of Rewritten Text	174
Bibliography	181

List of Figures

1.1	An illustration of how the <i>Classical NLG Pipeline</i> or CNP would work in action for an actual generation task and input example. Here, the task is to summarize the given input news article to within 280 characters. In addition to the classical components, we also include an overarching “Rhetorical Goals” layer, shown as a cylinder, which is seen in certain architectures such as that of [78]. The necessity of having such a layer for any reasonably realistic system is explained in §8. Having such a layer becomes a necessity for most real-world NLG tasks, since not all aspects of the communicative goal specifications deal with content (Recall the textual, ideational and interpersonal meta-function categorization from Halliday’s Systemic Functional Theory [72], which we also discuss in §7)	12
1.2	An illustration of how <i>End-to-End Neural NLG Pseudo-Pipeline</i> or E2EN2PP would work in action for an actual generation task and input example. Here, the task is to summarize the given input news article to within 280 characters. Note that this is a <i>Pseudo-Pipeline</i> , since the layers do not correspond to subtasks of NLG; moreover, they cannot be learnt or updated independently.	13
1.3	An illustration of how the <i>End-to-End Neural NLG Pseudo-Pipeline</i> or E2EN2PP fleshed out in Figure 1.2 would work in action for an actual generation task and input example, after incorporating the Intervention in Chapter 2. Here, the task is to summarize the given input news article to within 280 characters. The forward E2EN2PP here merely acts as a candidate generator, with the three new introduced components – Prior Estimator, Backward Model and Reranker producing the final output distribution used to generate the Final Output (by reranking candidates)	14

1.4	An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 8. Here, the task is to summarize the given input news article to within 280 characters. The pragmatic knowledge store here has additional knowledge about what would be apt referring expression preferences which the <i>Pragmatic Interpretation Layer</i> which it then uses to mark out redundant referring expressions which ought to be modified.	15
1.5	An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 6. Here, the task is to summarize the given input news article to within 280 characters. The text marked out in carrot-red in the <i>Final Output</i> , i.e., <i>dedocked</i> is clearly picked up by the model from the caption-expanded portion of the input (also marked in carrot-red)	16
2.1	An illustration of how the <i>End-to-End Neural NLG Pseudo-Pipeline</i> or E2EN2PP fleshed out in Figure 1.2 would work in action for our actual generation task and input example, after incorporating the Intervention described in this Chapter. The forward E2EN2PP here merely acts as a candidate generator, with the three new introduced components — Prior Estimator, Backward Model and Reranker producing the final output distribution used to generate the Final Output (by reranking candidates)	27
2.2	A sketch of our BACKWARD, noisy-channel model. The attentional S2S model with bidirectional encoder gives $P(x y)$ and next-character model gives $P(y)$, where y (<i>spime</i>) is the portmanteau and $x = \text{concat}(x^{(1)}, ";", x^{(2)})$ are the concatenated root words (<i>space</i> and <i>time</i>).	28
2.3	A sketch of the BASELINE FST-based pipeline approach from [38], starting with the input root words <i>jogging</i> and <i>juggling</i> to the left, leading to the final output, <i>joggling</i> at the rightmost end. This approach requires both root words $x^{(1)}$ and $x^{(2)}$ to be present in the CMU Phonetic Dictionary to get the phonetic sequences for the first step, as shown.	33
2.4	Attention matrices while generating <i>slurve</i> from <i>slider;curve</i> , and <i>bennifer</i> from <i>ben;jennifer</i> respectively, using <i>Forward</i> model. ; and . are separator and stop characters. Darker cells are higher-valued	35

3.1	Depiction of our overall architecture (showing decoder step 3). Attention weights are computed using previous decoder hidden state h_2 , encoder representations, and sentinel vector. Attention weights are shared by decoder RNN and pointer models. The final probability distribution over vocabulary comes from both the decoder RNN and the pointer network. Similar formulation is used over all decoder steps	47
3.2	Attention matrices from a <i>Copy</i> (top) and a <i>simple S2S</i> (bottom) model respectively on the input sentence “ <i>Holy Saint Francis, this is a drastic change!</i> ”. < s > and < / s > are start and stop characters. Darker cells are higher-valued.	57
5.1	We devise automatic ways to collect references sans any crowd-sourcing, through two types of knowledge sources: commonsense and retrieved instance knowledge, followed by automated adaptation to make them more fluent in the target contexts.	70
6.1	An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 6. Here, the task is to summarize the given input news article to within 280 characters. The text marked out in carrot-red in the <i>Final Output</i> , i.e <i>dedocked</i> is clearly picked up by the model from the caption-expanded portion of the input (also marked in carrot-red)	88
6.2	Graph displaying the average coverage (out of 100) by the top NTC captions in aggregate per concept set.	93
6.3	BLEU-4, CIDEr, and SPICE on test _{CG} over different values of NTC for BART-base and T5-base.	97
6.4	Snapshots of human evaluation: a) instructions seen by annotator and b) an example with questions.	106
7.1	An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter ???. Here, the task is to summarize the given input news article to within 280 characters. The dotted arrows represent the second pass of inference, after expanding the input with information extracted by the IE module from the Stage 1/ “Pre-Final” output(marked in red)	115

8.1	An illustration of how <i>End-to-End Neural NLG Pseudo-Pipeline</i> would work in action for an actual generation task and input example, after incorporating the Intervention in Chapter 8. Here, the task is to summarize the given input news article to within 280 characters. Note that this is a <i>Pseudo-Pipeline</i> , since the layers do not correspond to sub-tasks of NLG; moreover, they cannot be learnt or updated independently. The specific intervention shown here is the introduction of a <i>Pragmatic Interpretation Layer</i> that takes in the raw board states and featurizes them into a collection of discrete game-pertinent features.	122
8.2	Move commentary generated from our method (Game-aware neural commentary generation (GAC)) and some baseline methods for a sample move.	123
8.3	A multi-move, single commentary example from our data. Here, the sequence of moves Ba4 → b5 → Nd6 → bxa4 → e5 is commented upon.	124
8.4	The figure shows some features extracted using the chess board states before (<i>left</i>) and after (<i>right</i>) a chess move. Our method uses various semantic and pragmatic features of the move, including the location and type of piece being moved, which opposing team pieces attack the piece being moved before as well as after the move, the change in score by <i>Stockfish</i> UCI engine, etc.	127
8.5	The figure shows a model overview. We first extract various semantic and pragmatic features from the previous and current chess board states. We represent features through embedding in a shared space. We observe that feeding in feature conjunctions helps a lot. We consider a selection mechanism for the model to choose salient attributes from the input at every decoder step.	128
8.6	Outputs from various models on a test example from the MoveDesc subset. . .	132
8.7	Example output 1: Move description subset of data.	141
8.8	Example output 2: Move description subset of data.	141
8.9	Example output 3: Move description subset of data.	142
8.10	Example output 4: Move description subset of data.	142
8.11	Example output 5: Move description subset of data.	142
8.12	Example output 6: Move description subset of data.	142
8.13	Example output 7: Move description subset of data.	143
8.14	Example output 8: Move description subset of data.	143
8.15	Example output 1: Move quality subset of data.	144
8.16	Example output 2: Move quality subset of data.	144
8.17	Example output 3: Move quality subset of data.	144

8.18 Example output 4: Move quality subset of data.	144
8.19 Example output 5: Move quality subset of data.	145
8.20 Example output 6: Move quality subset of data.	145
8.21 Example output 7: Move quality subset of data.	145
8.22 Example output 1: Comparative subset of data.	146
8.23 Example output 2: Comparative subset of data.	146
8.24 Example output 3: Comparative subset of data.	146
8.25 AMT (Amazon Mechanical Turk) sample HIT (Human Intelligence Task): Part 1 of 2 : Two chess proficiency questions are asked at beginning of a HIT	147
8.26 AMT (Amazon Mechanical Turk) sample HIT (Human Intelligence Task): Part 2 of 2: 7 sets of questions are asked to judge quality of generated text. Each of the seven texts is output from a different method.	148
8.27 Commentary text: <i>I develop my bishop to the queen</i> . An example instance where output commentary from our method was marked as not valid for the given chess move	149
 9.1 Example of our task and dataset, with original input story S on the left, target narrative order $\pi_{i'}$ on the top, and human rewritten story S' on the right.	153
9.2 Snapshots of a) instructions seen by annotator before writing, b) one of the examples provided, and c) part of the page the annotator interacts with while rewriting.	172
9.3 Snapshots of Plot-Preservation study: a) instructions seen by annotator and b) part of the page annotators interact with while answering the question.	173

March 25,2022

List of Tables

2.1	10-Fold Cross-Validation results, D_{Wiki} . $Attn$, Ens , $Init$ denote attention, ensembling, and initializing character embeddings respectively.	33
2.2	Results on D_{Blind} (1223 Examples). In general, BACKWARD architecture performs better than FORWARD architecture.	34
2.3	Example outputs from different models. Outputs are from best performing configurations of the models. G.TRUTH denotes the ground truth portmanteau. . . .	36
2.4	AMT annotator judgements on whether our system’s proposed portmanteau is better or worse compared to the baseline	37
3.1	Examples from dataset showing modern paraphrases (MODERN) from the learning resource Sparknotes.com of few sentences from Shakespeare’s plays (ORIGINAL). We also show transformation of modern text to Shakespearean text from our models (COPY, SIMPLES2S and STAT).	44
3.2	Dataset Statistics	46
3.3	Test BLEU results. Sh denotes encoder-decoder embedding sharing ($No=\times$, $Yes=\checkmark$) . $Init$ denotes the manner of initializing embedding vectors. The $-Fixed$ or $-Var$ suffix indicates whether embeddings are fixed or trainable. COPY and SIMPLES2S denote presence/absence of <i>Copy</i> component. $+SL$ denotes sentinel loss.	56
5.1	Utterance level Spearman Rank Correlation [219] and Kendall Tau Rank Correlations [88]. (1) SCARCE-SINGLE augments the original single human written response (SINGLE) in DailyDialog dataset [107] using the devised method. It leads to large improvements in correlations across most of the metrics, when compared to SINGLE. (2) SCARCE-MULTI augments the MULTI dataset, again leading to improvements in correlations to human ratings, especially for BLEU and BERT-Prec metrics.	73
5.2	Analyzing impact of various components	75

5.3 Utterance level Spearman Rank Correlation [219] with p-values. (1) SCARCE-SINGLE augments the original single human written response (SINGLE) in DailyDialog dataset [107] using the devised method. It leads to large improvements in correlations across most of the metrics, when compared to SINGLE. (2) SCARCE-MULTI augments the MULTI dataset, again leading to improvements in correlations to human ratings, especially for BLEU and BERT-Prec metrics. Additionally, we note that almost all of the correlation values with SCARCE-MULTI are statistically significant with $p < 0.05$	80
5.4 Templates and rules to transform semi-structured COMET outputs to surface NL forms.	81
5.5 Example context-response pairs from the test split of DailyDialog, showing the automated responses returned by different sub-components of SCARCE. CONTEXTADAPT is shortened to CA for brevity.	84
6.1 Examples of retrieved images, associated captions, baseline and VisCTG (our visually grounded model’s) generations for select concept sets. Note that the images and captions are used as an intermediary to guide the final generation and thus the final generation need not be faithful to them. E.g. there is nobody petting the cat in the image or caption, but since the VisCTG output is conditioned on both the concept set and the caption, it includes <i>being petted</i>	89
6.2 Statistics of CommonGen dataset splits.	91
6.3 Comparing dev_O performance of our re-implemented models to those in Lin et al. [110]. Bold represents where we reach/exceed reported numbers. Results averaged over two seeds for our models. Lin et al. [110] did not report BART-base. See §6.2.3 for metric explanations for comparison of all metrics.	91
6.4 Example generations from our baseline models versus human references.	92
6.5 Examples of augmented inputs and final generations for varying values of NTC.	94
6.6 Automatic eval results for BART on test_{CG} over two seeds. Bold corresponds to best performance on that metric. We include stat sig p-values (from Pitman’s permutation test [163]) for VisCTG compared to the baseline. Insignificant ones ($\alpha = 0.1$) marked with *.	96
6.7 Automatic eval results for T5 on test_{CG} over two seeds. Bold corresponds to best performance on that metric. We include stat sig p-values (from Pitman’s permutation test [163]) for VisCTG compared to the baseline. Insignificant ones ($\alpha = 0.1$) marked with *.	97

6.8	Automatic eval results of VisCTG models on test_O , evaluated by CommonGen authors. We compare to reported baseline numbers in Lin et al. [110] (they did not evaluate BART-base), and models on their leaderboard with publications at time of writing that outperform baselines. Their leaderboard reports BLEU-4, CIDEr, and SPICE. Bold corresponds to best performance (for those three) per model type+size.	98
6.9	Avg. AMT eval results on test_{CG} for <i>overall quality</i> . O1: VisCTG wins, O2: baseline wins, O3: both indistinguishable. Bold corresponds to higher fractional outcome between O1 and O2. All results are statistically significant based on paired two-tailed t-tests and $\alpha = 0.1$. The inter-annotator agreement (IAA) is the average direct fractional agreement (where both annotators choose O1 or O2) over all examples. See §6.5.2 for further details.	98
6.10	Avg. expert linguist eval results on test_{CG} for BART-large. O1: VisCTG wins, O2: baseline wins, O3: both indistinguishable. Bold corresponds to higher fractional outcome between O1 and O2 per aspect. See §6.5.2 for further details.	98
6.11	Qualitative examples for test_{CG} . <i>BL</i> stands for baseline. <i>Concept set</i> refers to the input keywords and <i>Captions</i> refers to the captions (separated by <s>) used by the VisCTG model for that particular example to produce its final generation.	99
6.12	Performance of our re-implemented CommonGen models on dev_O compared to the original numbers reported in Lin et al. [110]. Note that for our models, results are averaged over two seeds, and that the original authors did not experiment with BART-base or report BERTScore. Bold indicates where we match or exceed the corresponding reported baseline metric.	105
6.13	Further qualitative examples for test_{CG} . <i>BL</i> stands for baseline. <i>Concept set</i> refers to the input keywords and <i>Captions</i> refers to the captions (separated by <s>) used by the VisCTG model for that particular example to produce its final generation.	107
8.1	Dataset and Vocabulary Statistics	125
8.2	Commentary texts have a large variety making the problem of content selection an important challenge in our dataset. We classify the commentaries into 6 different categories using a classifier trained on some hand-labelled data, a fraction of which is kept for validation. % data refers to the percentage of commentary sentences in the tagged data belonging to the respective category.	125

8.3	Performance of baselines and our model with different subsets of features as per various quantitative measures. (S = Score, M = Move, T = Threat features;) On all data subsets, our model outperforms the TEMP and NN baselines. Among devised models, GAC performs better than GAC-sparse & RAW in general. For NN, GAC-sparse and GAC methods, we experiment with multiple feature combinations and report only the best as per BLEU scores.	133
8.4	Performance of the GAC model with different feature sets. (S = Score, M = Move, T = Threat features;) Different subset of features work best for different subsets. For instance, <i>Score</i> features seem to help only in the Quality category. Note that the results for Quality are from 5-fold cross-validation, since the number of datapoints in the category is much lesser than the other two.	134
8.5	The COMB approaches show the combined performance of separately trained models on the respective test subsets.	134
8.6	Human study results on <i>MoveDesc</i> data category. Outputs from GAC are in general better than ground truth, NN and GAC-sparse. TEMP outperforms other methods, though as shown earlier, outputs from TEMP lack diversity.	135
8.7	Some commentary texts from each of the six categories. The Categories column lists those into which the example falls. As pointed out earlier, the category labels are not exclusive i.e., a text can belong to multiple categories, though texts with more than one category are few in our dataset. ('Desc' is short for 'Move Description')	140
9.1	Sentence pairs in testSup stories are annotated for 4 linguistic change types common in NAREORC. Sent denotes % of sentence pairs showing that change type. Stor denotes story pairs (S, S') where \geq one sentence pair shows that change type.	158
9.2	Average human evaluation results on testSup (excluding interestingness), rated from 1-5. Bold corresponds to best model performance per metric, and underline second-best model performance.	162
9.3	Average interestingness results on testSup, rated from 1-5 (3 represents equal to original story). Models are 2S versions. Bold corresponds to best performance, and underline second-best.	162
9.4	Average automatic evaluation results on testSup (values multiplied by 100). Bold corresponds to best performance per metric, and underline second-best (excluding the TOF metrics that are mainly for validation).	163

9.5	Pearson and Spearman correlations between automatic and human evaluation metrics, with p-values in brackets. TOF metrics excluded as they are mainly for validation. Bold corresponds to highest correlation per human evaluation metric.	163
9.6	Qualitative examples for testSup. Target perms are in brackets beside original stories. <i>d</i> refers to denoise, and <i>r</i> to reorder (2S models).	164
9.7	Sentence ordering on control vs. challenge sets.	167
9.8	Further qualitative examples for testSup. Target permutations are in brackets beside the original stories.	175
9.9	Further qualitative examples for testSup. Target permutations are in brackets beside the original stories.	176

March 25,2022

Chapter 1

Introduction

The old order changeth yielding place to
new;
And God fulfills himself in many ways,
Lest one good custom should corrupt the
world.

Lord Alfred Tennyson

The old that is strong does not wither,
Deep roots are not reached by the frost.

JRR Tolkien

1.1 Natural Language Generation

1.1.1 Preliminaries & Guiding Principles

1. Grammar, Grammaticality & Fluency:

A *grammar* is a set of rules of the form $\Sigma^* \rightarrow \Sigma^*$, defined over an alphabet of symbols $\Sigma = NT \cup T$, where NT are the non-terminal symbols and T are the terminal symbols.

We call a piece of text *grammatical* or say that it possesses *grammaticality* if it can be generated by the grammar of English, or the language under question. Note, however, that it is an almost impossible task to write a grammar for an entire existing, *sui-generis* language which handles all sentences/phenomena seen in that language, though even the earliest grammarians like Panini [192] have made attempts at this. As a result, when we say *grammatical* what

we mean is that the piece of text would be considered *acceptable* by most native speakers of the language when they are asked so, and is hence also sometimes called *acceptability*. A related but slightly different notion is that of fluency — a text is fluent if it sounds like a natural text you would hear from a native speaker of the language — such texts of course would be largely grammatical , but it would also exclude grammatical sentences which are meaningless i.e., they are so implausible that it is hard to assign them a meaning, even an abstract or imaginative one e.g., Chomsky's famous example *Colorless green ideas sleep furiously*.

2. Finite State Automatons

An automaton is a finite-memory (or one-step memory) accept-reject mechanism that can take in a sequence of symbols from some symbolset Σ as input. Internally, the automaton consists of a i) set of states ii) transition arcs between states which are traversed based on the current input symbol read iii) start and end states ; note that these are from the existing states and may themselves overlap

Symbol sequences which on being read by the automaton take it to one of its stop states are said to be accepted by the automaton. Every automaton has a corresponding CFG associated with it. These automatons are also known as *Finite State Automatons* (FSAs).

A generalization of FSA is that of pushdown automata, which are also provided with a stack of potentially unbounded length.

Finite State Transducers (FSTs) are FSAs which can also emit output symbols during transitions (or after reaching an input state, depending on how one may define it)

In Chapter 2, we will see FSTs being employed by one of the baseline approaches.

3. Language Model:

A language model (LM) defines a probability distribution $P(s)$ over all possible word (or subword/character, depending on modelling choice and task etc.,) sequences $s \in S$, where S is the Kleene closure of the vocabulary V .

Many LM architectures are factored in left-to-right fashion $P(s) = \prod_{i=2}^{i=|s|} P_{next}(w_i | s_1^{i-1})$, where P_{next} is the next-word distribution and s_i^{i-1} is the subsequence of the first $(i - 1)$ elements of s . Note that there also exist other formulations, e.g., whole sentence language models [193].

4. Transducer

A transducer is a model $f_\theta() : V_{in}^* \times X \rightarrow V_{out}^*$ which can accept an input sequence string $s_{inp} \in V_{in}^*$ from an input vocabulary V_{in} , where $*$ is the Kleene closure and θ are the transducer's parameters, along with other potential inputs/parts of the communicative

goal (themselves symbolic or continuous) $x_{in} \in X$, and output a sequence $s_{out} \in V_{out}^*$. The transduction function can be written as $x_{out} = f_\theta(s_{in}, x_{in})$

Any NLG model basically functions as a transducer at inference time/test-time (taking in the communicative goal and returning the output sequence), though the internal representations, learning process and architecture can vary significantly.

When $X = \phi$ and V_{out} and $V_{in} \subset V_{out}$, it becomes possible to use any left-to-right factored language model architecture as a transducer. This is since one can now feed in s_{in} as the first few tokens of a segment or the “prompt” to the language model (this part is “teacher forced” running of the model i.e., since the sequence is predetermined, the language model is only fed the sequence under question) and then predict out s_{out} using the language model using some decoding method (see §5 for more).

Transducer models which have two roughly separable modules can be called Seq2Seq or encoder-decoder models. The first module, or the *encoder* is for representing s_{in} in some symbolic or continuous intermediate form h (Note that h could even be a sequence or set of things, e.g., a set of vectors). The second module uses h to then generate s_{out} — this module is known as the *decoder*. The term Seq2Seq is also often used in a wider sense for any neural transducer and not just the particular form above. Seq2Seq models where the decoder uses some internal form of attention mechanism [9] are also described as *attentional*.

5. Decoding

Decoding refers to the algorithm which finally uses a learnt NLG model to produce the output given an instance of the communicative goal (or “input” instance, if we assume the rest of the communicative goal to remain fixed for the “task”).

6. Infilling

Intuitively, infilling refers to the process of using a learnt model to perform “fill in the blanks” i.e., predicting a masked out (usually using a special character e.g., [MASK]) token given its surrounding context. Depending on the model architecture and training, this might involve the entire left and right contexts or subsets of them (e.g., only the left context for left-to-right language models).

7. Systemic Functional Linguistics

Systemic Functional Linguistics (SFL) was a theory devised by the linguist M.A.K. Halliday in the 1970s [70] SFL categorizes subgoals or subparts of the communicative goal into three metafunctional categories:

- (a) **Ideational Goals:** These subgoals pertain to the author’s state of mind; their knowledge, memory and experience about the various states of the world (factual, physical

etc.,), as well as the motivation and intent underlying the communicative goal (persuasion vs communication?)

- (b) **Interpersonal Goals:** These subgoals pertain to the relationship between the speaker and the listener/addressee. It also subsumes subgoals pertaining to the medium of transmission, or the individual physical / emotional states of the addressee
- (c) **Textual Goals:** These subgoals pertain to choices in terms of the order of presentation of information in the text, the subset of textual surface forms employed, and the internal structure and packaging of the text in terms of its constituent sentences, phrases, words and other elements.

8. Rhetorical Goals

The non-textual subgoals of the wider communicative goal, namely those which can be categorized under the *Ideational* and *Interpersonal* metafunctions are also sometimes referred to as *Rhetorical Goals*.

9. Gricean Maxims

The four Gricean maxims of *Quality*, *Quantity*, *Manner* and *Relation* are four general guidelines relating the pragmatics of the speaker and their actual utterance; implicitly followed by most human speakers (though sometimes violated intentionally, e.g., for humour) They are sometimes together also referred to as The Cooperative Principle.

- (a) **Maxim of Quantity:** Be as informative as is required; but not any more. The phenomenon of implicature is often an outcome of this maxim.
- (b) **Maxim of Quality:** Do not say what you don't believe in; or what you believe in but think the evidence is insufficient.
- (c) **Maxim of Relation:** Be as relevant as possible.
- (d) **Maxim of Manner:** Be as clear, unambiguous, simple as you can while conveying the information you intend to.

1.1.2 Defining a NLG System

Communicative Goal (CG)

The overall goal which the output of the NLG system must satisfy in order for the process of generation, and consequently the model, to be deemed successful. This also includes all the information which the NLG model needs to modify, process and condition on while generating its output.

It is often common to characterize and address certain parts (or subgoals) of the CG as controls, input and style(s) etc., though the choice of these parts is highly subjective in nature – for example, for a movie review, the sentiment is considered part of the “input” when doing formality transfer, but is considered a “control/style” when doing sentiment transfer.

Subtasks

The subtasks of natural language generation are a conceptual decomposition of the activities to be performed to generate a text, given a CG. They may also be thought of as subgoals to be accomplished before the overall CG has been achieved.

1. Content Selection:

Before generating the sentences and words, it is necessary to decide “What all to say?” out of all the potential things which suitably fit the communicative goal. The set of these choices is called content selection.

2. Content Ordering:

Having decided what to say, it is necessary to decide “In what order?” the selected pieces of information from content selection would be presented in. The process of choosing this is content ordering. Collectively with content selection, the two are also referred to as Macroplanning or Sentence Planning. This can be heavily dependent on the rhetorical goals (roughly speaking, extra-textual goals; see §8 for a complete description) e.g., For a Twitter thread, it might be required to place more retweetable and topically high-coverage content earlier on.

3. Sentence Aggregation:

This subtask pertains to the breaking up and packaging of the content to present into sentences, according to the broad order decided in Content Ordering.

4. Lexicalization:

This subtask refers to the choice of which word forms to broadly use in each sentence. Note that some subdecisions maybe left unspecified for the latter stages, especially Surface Realization.

5. Referring Expression Generation:

Referring Expression Generation, a.k.a. *Refex Generation* is the choice of expressions, or refexes to point to various entities, events or other item types while mentioning them at each point in the generation. (this can include pieces of the generated output itself i.e., discourse segments e.g., as in “In our earlier argument, … ”)

6. Surface Realization:

Also referred to simply as *Realization*, this refers to the final, explicit generation of the output text, resolving all the partially specified elements from earlier stages, as well as filling in remaining gaps based on syntactic, co-occurrence based, prosodic and other considerations.

Though there is a natural ordering and sequence to the subtasks based on their typical mutual dependency, and that is the order in which we shall present them, they need not always be performed in that order, though the Classical NLG Pipeline which we shall describe in 1.1.2 makes a best attempt to do so. For instance, for many a CG, Referring Expression Generation might be entirely independent of Lexicalization. For some others, it may be very closely tied to syntax (e.g., in pro-drop languages like Spanish) (and hence would need to be revised) during Surface Realization.

Classical NLG Pipeline (CNP) & Its Stages

1. Macroplanning:

This stage deals with the discourse level, i.e., the level where sentences are elements; also sometimes called the *macrostructure* or the *macro level*. This stage handles the *Content Selection* and *Content Organization* subtasks.

2. Microplanning:

This stage deals with the sentence level, i.e., the level where words/phrases are elements; also sometimes called the *microstructure* or the *micro level*. This stage handles the *Sentence Aggregation*, *Lexicalization* and *Referring Expression Generation* subtasks.

3. Surface Realization:

This stage handles the sole final subtask – i.e., the identically named *Surface Realization*.

4. What about Understanding?

An important detail omitted from most discussion or descriptions of the CNP is understanding the input, whether it be for data-to-text or text-to-text generation tasks.

For simplicity of presentation, we will present and discuss issues pertaining to understanding at a certain level at the corresponding generation level. For instance, issues related to understanding discourse markers and coreference chains would be discussed at the macroplanning level.

5. Need for an Overarching Rhetorical Goals Layer

To satisfy the rhetorical (sub) goals within the wider CG, which are by definition *not textual* in nature, the NLG model needs to potentially factor them in at each of the subtasks in

the *CNP*. However, in its most basic form, the CNP only allows the CG as an input at its topmost stage, i.e., Macroplanning, from whence it can only affect the bottom stages through the content and order choices made at the topmost stage. This condition is naturally too restrictive and limiting. It is for this reason that we introduce an overarching “Rhetorical Goals Layer” (depicted as a vertically oriented cylinder in Figure 1.1) which provides access to all rhetorical goals as well as any of their intermediate states as one traverses down the CNLP.

We borrow the idea for such a layer from [78].

End-to-End Neural NLG Pseudo Pipeline

End-to-End Neural NLG Pseudo Pipeline or E2EN2PP refers to the canonical neural architecture for NLG based on the Seq2Seq paradigm, where one or more encoders first encode the CG, which is initially an input string but later embedded into continuous space through the encoder embedding layers. Next, the encoder representations are aggregated or functionally transformed in various ways. Finally, a decoder network uses any of the encoder representations to compute the probability/loss functions based additionally off the gold output (at training time) or to generate the output text at test-time using some inference/search procedure or sampling method.

Note that though the Seq2Seq paradigm in general, and our characterization of it here in particular, though general enough to include many paradigms of neural architectures do not cover all of them exhaustively – particular exceptions being VAEs, GANs, Energy-Based Models etc. We leave performing a similar study on these models with a generalization of our framework, as we do in this thesis, as a point for future work.

Medium Constraints

These are constraints relating to the medium of transmission between the speaker and the listener, rather than their individual states or intents, or their pairwise relationship. Nonetheless, as per the three-way SFL classification, these would constraints would be classified under interpersonal goals, barring those which are explicitly tied to the text itself (e.g., using a maximum of 280 characters), in which case they would (also) be classified as textual goals

Controls

A *control* is a variable or a property defined over any text, which is specified as a part of the CG to hold necessarily a particular range or subset of values for the target text. The set of all controls specified in the CG are sometimes simply referred to as *controls*. This could include for example, properties like the number of sentences, token length, or even continuous values such as entropy of the word distribution. This can also encompass properties defined by way of functions or models, such as estimators of text simplicity, or perplexity according to some pretrained language model.

Listener

Also referred to variously as the *Addressee* or the *Target* or the *Audience*, this refers to the individual or group of people who will finally read or listen to the generated text.

Control and Transfer tasks

Controllable generation tasks [156, 164] are generation tasks with a 2-part CG.

1. A content-based or textual goal, often simply called the *input* or ‘*content*’. For instance, controllable infobox-to-biography generation, this would simply be the Wikipedia Infobox. For each unique test-time example, the input remains fixed.
2. Ensemble of one or more content-orthogonal/non-content goals (*interpersonal* and *ideational* goals, if one follows the Systemic Functional Linguistics terminology). These goals are also sometimes referred to as *styles*. Example control variables for controllable infobox-to-biography generation could be audience literacy, biography length etc. The control goals can each take on two or more discrete values, and can be dynamically varied by the user at test-time.

Transfer tasks are controllable generation tasks where the *input* is a fully-realized text with an initial or default configuration of control variables. They are sometimes also referred to as *style transfer* tasks.

Concept-To-Text-Generation Tasks

These are tasks where the CG requires generating a pertinent output of one or a few sentences, given a largely unstructured or semi-structured collection of “concepts” as an input. We will also refer to the input in such tasks as *concept set* or *input concept set* Commongen [110] (where

the task is to generate a single sentence describing a situation involving all the given concepts) and WebNLG [64] (where the task is to describe a sequence of SVO triples) are two prominent examples of this family of tasks.

1.1.3 Background and Recent History

Natural Language Generation (NLG) is the field of inquiry which aims to endow agents with the ability to generate natural language to satisfy any stated communicative goal given corresponding input information. NLG is a subfield of Natural Language Processing (NLP), and is often seen as a dual to Natural Language Understanding (NLU). From the inception days of AI, NLG capability became a part of its holy grail, with the ability to converse with human-indistinguishable replies being the crux of Turing’s Imitation Game.¹. Specifically, the imitation game stated that for an AI agent to be deemed sufficiently intelligent, it must be able to converse with a secluded human judge with enough proficiency such that the judge cannot tell apart either of the agent and another control human conversee as being distinctively more human.

Through the decades, NLG research largely followed the waves of broader AI research, though, being dependent on the underlying perceptual and understanding components from vision, NLU and other subfields, it was often the last to which these seeped through. For instance, during the first wave of statistical ML methods of the 90s and 2000s, one of the earliest NLG works to introduce a learning-based component was in the surface realization part of the Nitrogen system [102], which was as late as 1995. Application of these methods to higher granularity subtasks of NLG like content selection started in the post-2000s [43]. This was also the case with the latest wave of “deep”, neural machine learning methods, which began with the commanding performance improvements of deep convolutional neural networks in the ImageNet LSVRC-2010 Challenge. These methods first brought about dramatic improvements in vision , then closely followed by speech recognition (through novel loss functions like the CTC loss) [68], natural language understanding (through a revival of the already proposed continuous distributional representations or *word embeddings*) [35] and machine translation (through improvements to the already-known Elman RNN and the sequence-to-sequence framework) [9].

The Seq2Seq framework from neural machine translation [224] found widespread adoption and adaptation for NLG tasks [91, 134, 145, 206], leading to considerable gains on existing met-

¹Famed, and much mischaracterized, in posterity as *The Turing Test*

rics such as ROUGE². Inspite of these improvements, there remained gaps between human and model performance on most NLP benchmarks, and designing adequate architectures required significant engineering effort and task knowledge to incorporate task-specific inductive biases. With improvements brought about in this fashion saturating – people spoke of NLP not yet having its “Imagenet Moment” ³.

This aforementioned hurdle, of the absence of a generally good, underlying representation, was finally crossed on the back of two developments i) Introduction of transformers [227], which could handle longer contexts and also trained faster by an order of complexity, opening doors to training on much larger corpora ii) The concept of pretrain-and-then-finetune (PTFM), where a large model is first pretrained on terabyte-scale corpora, typically using self-supervision based objectives such as masked likelihood and then lightly finetuned with few additional layers for each specific task. ElMo [158] and BERT [39] were early successes of this paradigm, matching human performance on NLU benchmarks like GLUE [232]. This success finally made its way to NLG with OpenAI’s GPT [169] and GPT-2 [170], both of which were autoregressive language models. The latter garnered widespread attention for reaching hitherto unseen fluency at completing and extrapolating from arbitrary “prompts” of short to medium length, besides being finetuneable for specific tasks like story generation [207] and dialog [22]. Finally, this paradigm was also extended to sequence-to-sequence architectures, such as in BART [104] or T5 [173], which exhibited even stronger performance on NLG tasks due to stronger conditioning on, and better representation of input information. A good fraction of successive work to these milestone methods has been based on what [116] dub *objective engineering*, namely, devising better objectives either for pretraining [118, 252] or for task-specific finetuning [237].

Traditionally, before the 2010s, NLG was traditionally seen as a composite sequence of sub-tasks, with typical models being pipelines of stages. [180, 183, 184]⁴. For instance, McKeown’s TEXT architecture [130] for multi-sentence text generation, proposed in 1985, had

1. A “strategic component” stage to decide discourse-level structure and inter-sentence organization
2. A “tactical component” stage with a dictionary and grammar to realize individual sentences and words given a plan from the former

²ROUGE-2 scores on most datasets doubled from 7-10 range for non-neural baselines to 15-20 in [206]

³A reference to the pervasive practice in modern computer vision of having the same underlying base layers trained from ImageNet image classification, with task specific layers starting from this representation

⁴Note, though, that there were dissenters such as Appelt’s KAMP [6] who opposed this view from the outset.

This division into subtasks had its roots in multiple, but chiefly two considerations. The first was how the field of linguistics was historically subcategorized hierarchically into phonetics, morphology, semantics, discourse etc. The second, more engineering one was the relative ease offered by such a modular division to hand-designing both grammars/rules and the discrete representation/structures they manipulated, such as trees [84], frames [54], scripts [203] and schema [130], which were then the dominant paradigm underlining most AI systems for language.

By the mid-90s, a near convergence had been reached on what the stages of a NLG pipeline would be, with Reiter [180] discussing the psycholinguistic plausibility of a “consensus” 5-stage architecture:- *content determination* → *sentence planning* → *surface generation* → *morphology* → *formatting*. In 1997 [183], they reworked this to a 6-stage one:- *content determination* → *discourse planning* → *sentence aggregation* → *lexicalization* → *referring expression generation* → *surface realization*. Finally, writing in 2000, the same authors [184] presented a more concise 3-stage architecture, with “*macroplanning*” encompassing *content determination* and *discourse planning*, and “*microplanning*” encompassing *sentence aggregation*, *lexicalization*, *referring expression generation*. Specifically, their pipelined architecture was: *macroplanning* → *microplanning* → *surface realization*. We henceforth refer to this as the *Classical NLG pipeline*, or more tersely, CNP (see §1.1.2 for a longer explanation).

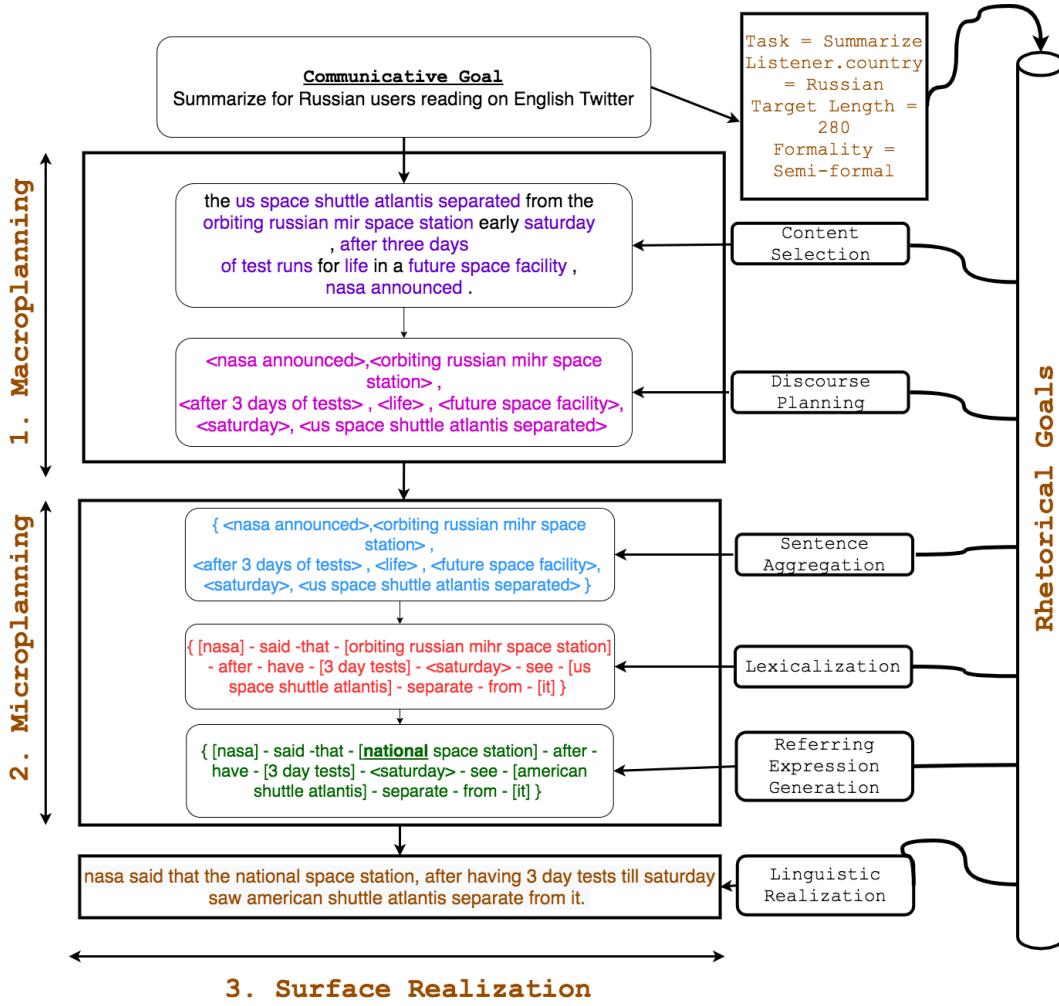


Figure 1.1: An illustration of how the *Classical NLG Pipeline* or CNP would work in action for an actual generation task and input example. Here, the task is to summarize the given input news article to within 280 characters. In addition to the classical components, we also include an overarching “Rhetorical Goals” layer, shown as a cylinder, which is seen in certain architectures such as that of [78]. The necessity of having such a layer for any reasonably realistic system is explained in §8. Having such a layer becomes a necessity for most real-world NLG tasks, since not all aspects of the communicative goal specifications deal with content (Recall the textual, ideational and interpersonal meta-function categorization from Halliday’s Systemic Functional Theory [72], which we also discuss in §7)

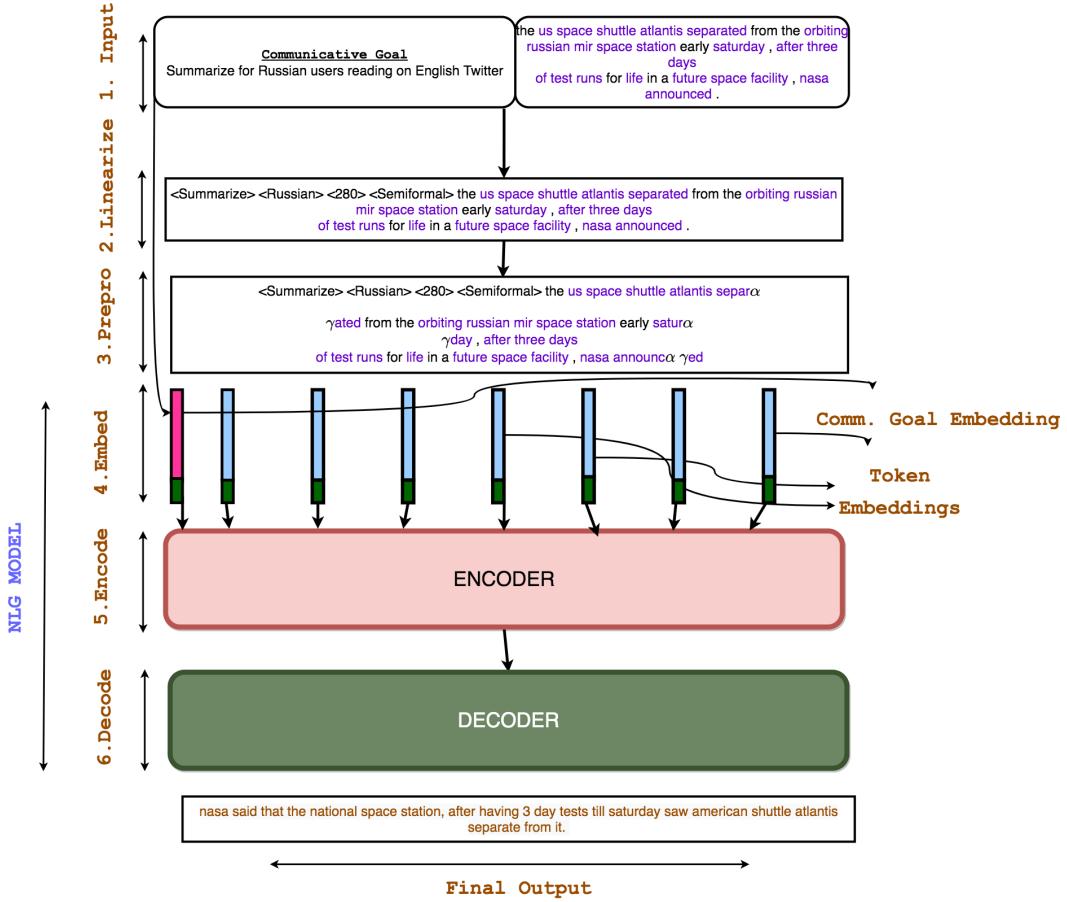


Figure 1.2: An illustration of how *End-to-End Neural NLG Pseudo-Pipeline* or E2EN2PP would work in action for an actual generation task and input example. Here, the task is to summarize the given input news article to within 280 characters. Note that this is a *Pseudo-Pipeline*, since the layers do not correspond to subtasks of NLG; moreover, they cannot be learnt or updated independently.

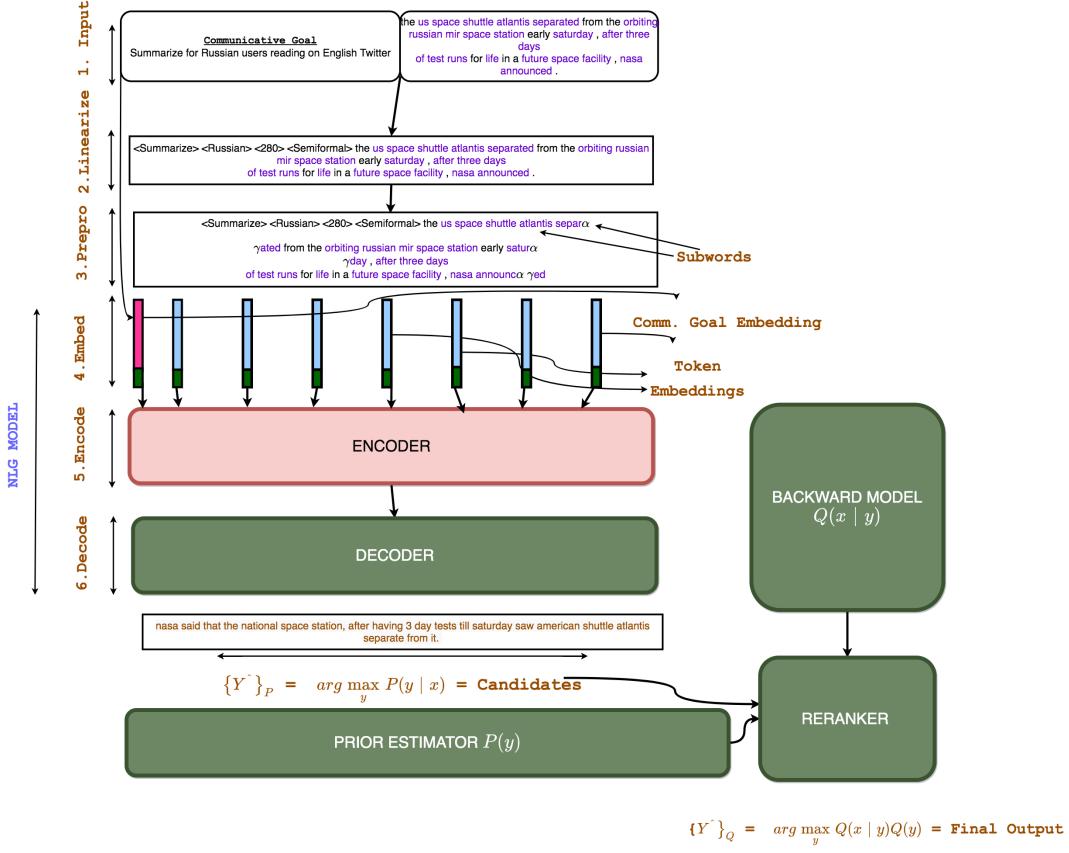


Figure 1.3: An illustration of how the *End-to-End Neural NLG Pseudo-Pipeline* or E2EN2PP fleshed out in Figure 1.2 would work in action for an actual generation task and input example, after incorporating the Intervention in Chapter 2. Here, the task is to summarize the given input news article to within 280 characters. The forward E2EN2PP here merely acts as a candidate generator, with the three new introduced components – Prior Estimator, Backward Model and Reranker producing the final output distribution used to generate the Final Output (by reranking candidates)

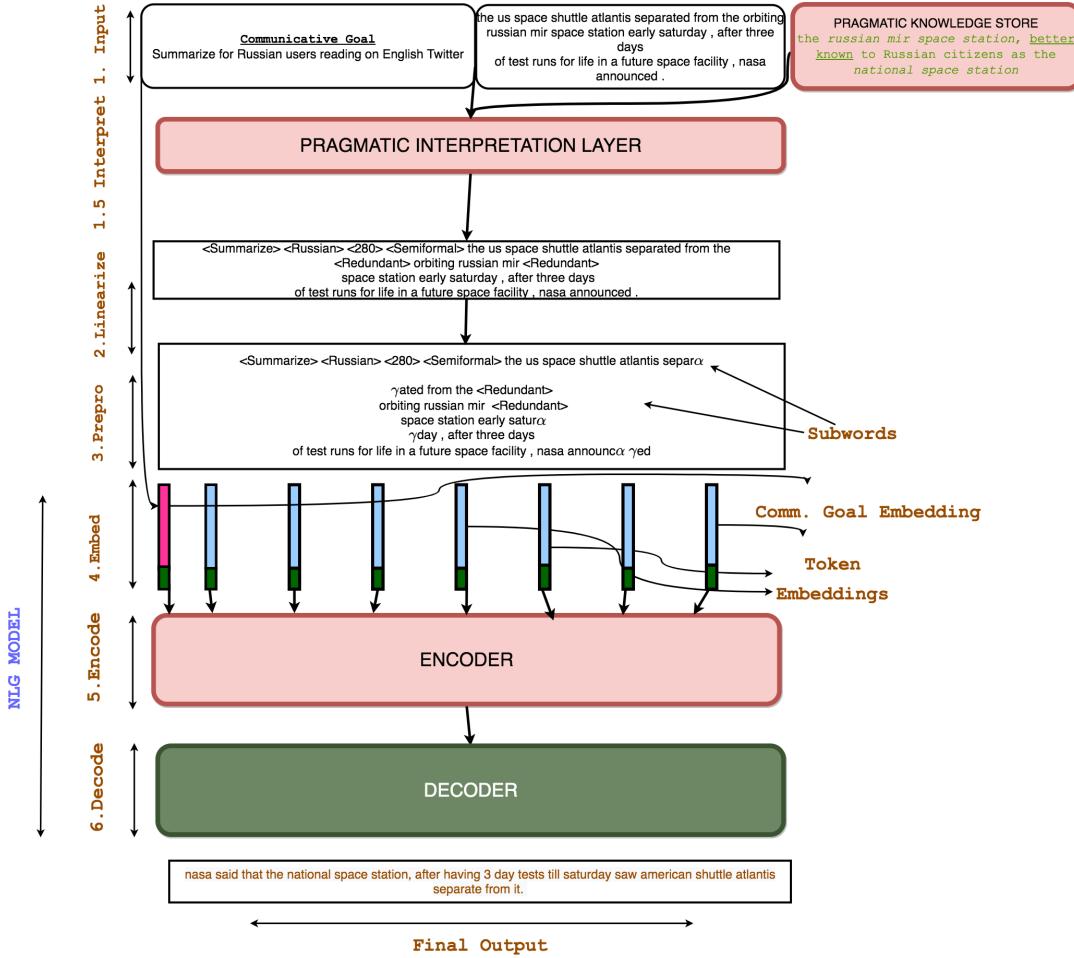


Figure 1.4: An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 8. Here, the task is to summarize the given input news article to within 280 characters. The pragmatic knowledge store here has additional knowledge about what would be apt referring expression preferences which the *Pragmatic Interpretation Layer* which it then uses to mark out redundant referring expressions which ought to be modified.

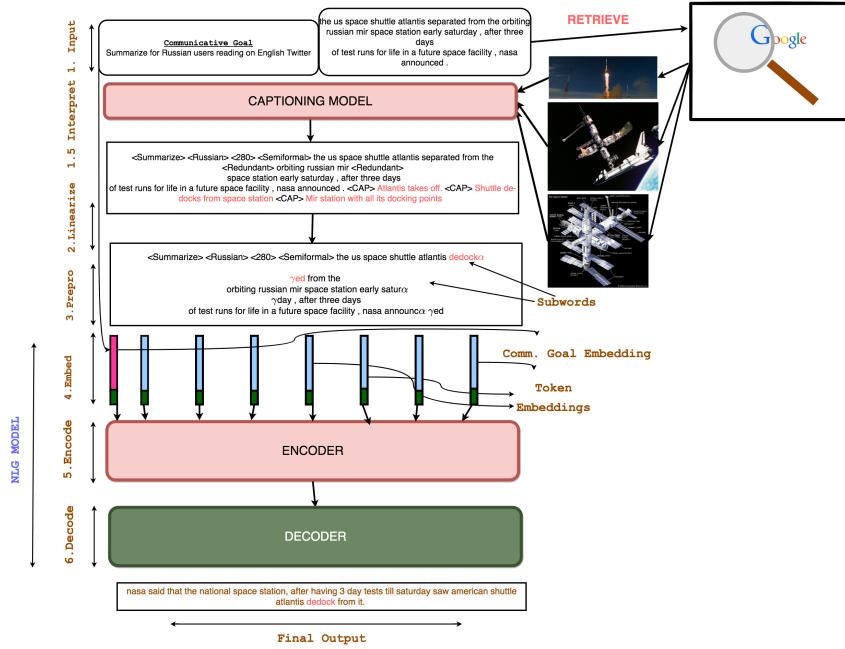


Figure 1.5: An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 6. Here, the task is to summarize the given input news article to within 280 characters. The text marked out in carrot-red in the *Final Output*, i.e., *docked* is clearly picked up by the model from the caption-expanded portion of the input (also marked in carrot-red)

1.2 Contributions & Structure

1.2.1 Contribution Types

Each contribution we make in this thesis addresses a particular aspect of NLG output, such as, for e.g., commonsense plausibility, content ordering, etc. Each such contribution falls under one or more of the contribution types below.

1. New tasks isolating the aspect, denoted by [Task].
2. Improve existing evaluation process for the aspect, denoted by [Evaluation].
3. Use knowledge external to CG to learn models better for the aspect, denoted by [Knowledge].
4. Improve Learning Model/Process. We denote this by [Method].

1.2.2 Thesis Contributions & Outline

The rest of this thesis is split into three parts – Part I (Surface Realization), Part II (Microplanning) and Part III (Macroplanning). The parts are further subdivided into chapters, each of which makes a focussed contribution touching one or more of the contribution types in §1.2.1. Furthermore, each chapter contains a “Broader Takeaways” subsection within its Conclusion to further contextualize and assimilate the concluded chapter within the evolving narrative of the thesis.

Part I: Surface Realization

This part of our thesis contains three chapters.

- **Chapter 2: Lexico-Phonetic Surface Realization and Portmanteaus**
- **Chapter 3: Stylistic Surface Transduction To Shakespeare Modern English**
- **Chapter 4: Tongue Twister Generation (Proposed)**

First, in **Chapter 2**, we address the aspect of *creativity in generation* through exploring a creative NLG task at the *lexico-phonetic level*, namely, the task of creating a suitably interesting as well as lexically and phonetically appropriate blend, a.k.a. portmanteau given two root words. We devise character-level neural sequence-to-sequence (S2S) methods for the task of portmanteau generation that are end-to-end-trainable, language independent, and do not explicitly use additional phonetic information.

A common property of many creative generation tasks is paucity of training data, arising from the low number of widely known creative artifacts produced for a particular task. Overcoming this paucity requires modifying the typical E2EN2PP pipeline by incorporating additional constraints drawn from underlying theory and properties that define the creative phenomenon. To enforce the property that portmanteaus should sound “word-like”, we devise a noisy-channel-style model that allows incorporation of unsupervised word lists into the learning process, improving performance over a standard source-to-target model that directly learns a distribution of the form $P(y|x)$ without refactoring it in Bayesian terms. Specifically, the intervention to the E2E NLG pipeline required here, is fleshed out in Figure 1.3. Overall, this chapter represents a contribution of type [Method]. This work on its completion was accepted for publication as a short paper at EMNLP 2017. The respective publication is [59].

Next, in **Chapter 3**, we address the aspect of *controlling diachronic register*, which is expressed primarily through *lexico-phrasal means*. Specifically, we explore the task of transferring

the style of a given sentence authored in contemporary English, such as e.g. *I am in a rush*, to the style of William Shakespeare, who wrote in the Early Modern English prevalent in Elizabethan times, such as e.g., *I stand on sudden haste*. The three methodical enhancements we propose, each motivated from a property of style transfer at the lexical and phrase level, lead to three major changes in the typical E2EN2PP pipeline. The first leverages the property of a shared language between source and target sides. The second exploits the property of considerable content invariance, specifically in terms of lexical choice, arising from meaning preservation. The third devises a mechanism to retrospectively adjust lexical representations to incorporate pairwise lexical source word → target word correspondences from additional knowledge available from a dictionary-like source. Besides exploiting the task properties, these inductive biases also serve the role of making the learning process less dependent on parallel training data, which is typically lesser for style transfer tasks compared to other typical transduction tasks, e.g., Machine Translation.

This chapter represents a contribution along both the [Method] and [Knowledge] types. This work on its completion was accepted as a long paper at the EMNLP 2017 Workshop on Stylistic Variation. The respective publication is [81].

Finally, concluding Part I, in **Chapter 4** we propose further exploring the aspect of *creativity* in generation which we already broached in Chapter 2, but this time through the prism of a more challenging setting that requires adhering to subgoals spanning two task granularities at once. Specifically, we propose the task of tongue twister generation. Tongue twisters are sentences that are *fluent* besides being *difficult to pronounce* e.g., "*She sells seashells on the seashore.*". Generating a tongue twister requires:

1. Maintaining difficulty of pronunciation, which is a token and phrase-level subgoal related to the lexico-phonetic level of surface realization.
2. Maintaining fluency, that is a subgoal at the phrase and sentence levels of surface realization.

We foresee a challenge arising in this task not just from having to satisfy two subgoals simultaneously, but additionally from them being expressible only through two distinct representational primitives of phoneme sequences and lexeme (word/wordpiece) sequences simultaneously.

This chapter aims to represent a contribution along the [New Task] types.

Part II: Microplanning

This part is comprised of three chapters

- **Chapter 5: Improving Realization Level Metric Evaluation of Open Dialog via Microplanning Rollouts for Reference Augmentation**
- **Chapter 6: VisCTG: Improving Plausibility of Microplanning for Concept-To-Text Generation Through Retrieve-Caption-Generate**
- **Chapter 7: Better Microplans For Concept-to-Text Tasks By Self Introspecting Input Expansion (Proposed)**

First, in **Chapter 5** we address the aspect of *microplanning-sensitive evaluation*, particularly in the context of evaluating NLG outputs against references using automatic evaluation metrics such as BLEU that, by definition, perform matching only at the level of surface realization. These metrics are essential for NLG model development since constant, per-configuration human evaluation is infeasible. However, their restriction to surface-level matching is problematic for tasks like dialog, where there are many adequate output responses for a given input dialog context, since it necessitates collection of multiple references to cover the range of output responses. This is expensive, time-consuming and not scalable. We devise a novel suite of techniques for automatically expanding a reference set to a larger set of pseudo-references, sans any added annotation. Our formulation treats the dialog context as a microplan-in-progress and projects out multiple pseudo-responses from it, with the help of both commonsense and retrieved instance-based knowledge.

This chapter makes a contribution along both the [Evaluation] and [Knowledge] types. A work based on this chapter was accepted as a long paper at Findings of ACL 2021 [62].

Next, in **Chapter 6** we address the aspect of *commonsense plausibility* of NLG outputs and how to improve the same. We explore this aspect through the lens of devising a model agnostic enhancement to SOTA pretrained generator models to bolster their microplanning, and consequently, their output plausibility, while doing concept-to-text generation tasks (see §1.1.2) such as CommonGen [110], where the communicative goal is to generate a sentence constructing a plausible situation from a given set of input concepts.

We first identify several critical issues in baseline model generated outputs for this task, like poor plausibility, inadequate lexical relationships, and incomplete arguments etc. We posit that properties specific to the textual modality, such as the Gricean maxim of Quantity and the Zipfian nature of concept occurrence, could indeed have a marked negative downstream effect on the NLG model’s learning for CommonGen. We devise an enhancement that augments the

input concepts by drawing information from the visual modality to help dampen this negative effect. Comprehensive evaluation and analysis demonstrate that this enhancement noticeably improves base model performance while successfully addressing the issues earlier noticed in the baseline output.

Specifically, the intervention we devise to the E2EN2PP is the addition of an Input Expansion Layer between the Input and the Embedding Layer. Before passing the input string to the Embedding Layer, the Input Expansion Layer symbolically augments it with the captions of retrieved relevant images. Figure 1.5 illustrates our intervention.

This chapter makes a contribution along both the [Method] and [Knowledge] types. A work based on this chapter was accepted as a long paper at AAAI 2022 [53].

Finally, through Chapter 7, we continue to address the aspect of *commonsense plausibility*. Specifically, we continue in the Commongen task setting just like Chapter 6. We aim to enhance the method to improve the commonsense plausibility of generated outputs, albeit using a proposed approach which, unlike Chapter 6 does not require information from another modality.

Our approach is based on the observation that large, pretrained generator models, by virtue of having been trained to predict masked out words given their contexts on large corpora such as BookCorpus, also tend to acquire a measure of factual and commonsense knowledge. We aim to devise an overarching model architecture based on breaking the aggregate process of generation into two passes through the base model , each of which can distinctly leverage one of the two abilities of these NLG models, in order, i.e., i) as a concept augmentation/expansion mechanism ii) as a text sequence to text sequence transducer. Such a formulation makes increased sense in particular for concept-to-text generation tasks, especially CommonGen, since the communicative goal is to construct a sentence describing a sufficiently complete, common-sense plausible situation involving all the given input concepts.

This chapter aims to represent a contribution along the [Method] types.

Part III: Macroplanning

This part is comprised of two chapters, addressing the content selection and content ordering subtasks respectively.

- **Chapter 8: Viable Content Selection and Refex Generation Through Pragmatic Backoff For Chess Commentary Generation**
- **Chapter 9: Macro-Level Controllable Generation based on Elements of Narrativ-**

ity: The Narrative Reordering Problem

First, in **Chapter 8**, we address the aspect of *content selection* and *referring expression generation*, particularly in settings which requires an understanding of domain pragmatics. Specifically, we present the novel NLG task of generating short, interesting natural language commentary for each chess game move during gameplay. We show how S2S models simply based on a E2EN2PP suffer from the common response problem [41] and fail to produce commentary that is even at the level of a template based baseline. We posit that this arises from the inability to acquire tabula rasa the pragmatic knowledge necessary to understand the input game state. We devise an alternative model that includes an additional Pragmatic Interpretation Layer to discretely featurize the board states using a game library, essentially backing off to pragmatic game knowledge to viably declutter the input states, thereby simplifying the understanding and overcoming the microplanning and macroplanning issues observed. The devised intervention in the E2EN2PP that needs to be done can be seen in Figure 1.4.

This chapter makes a contribution along the [[Knowledge](#)] and [[New Task](#)] types. A work based on this chapter was accepted as a long paper at ACL 2018 [82].

Second, in **Chapter 9**, we address the macroplanning related aspects of *maintaining multi-sentence coherence* and *preserving underlying plot*, particularly in the context of a change in the configuration of *content order*. In this chapter, we define and investigate the task of Narrative Reordering (NAREOR) where the communicative goal involves rewriting a given story in a different narrative order while preserving its plot. A NLG system that can rewrite the story in a different narrative order such that it retains its coherence will also ensure adequate interpretation and understanding by the reader. We present a dataset, NAREORC, with human rewritings of stories within ROCStories in non-linear orders. Simply reordering sentences is far from sufficient, as rewritten text must be adjusted to handle coreference, tense, time expressions *inter alia*.

This chapter makes a contribution along both the [[Evaluation](#)] and [[New Task](#)] types. A paper based on this chapter was accepted as a long paper at AAAI 2022 [61].

1.3 Projected Timeline

- Feb. 2022 Thesis Proposal
- Mar. 2022 – Apr. 2022 Finish the work in progress in [chapter 7](#)
- Apr. 2022 – May. 2022 Finish the proposed work in [chapter 4](#)
- May. 2022 – Jun. 2022 Job Search
- July. 2022 Thesis Defense

March 25,2022

Part I

Realization

Chapter 2

Lexico-Phonetic Surface Realization and Portmanteaus (EMNLP 2017) [Method]

For instance, take the two words "fuming" and "furious". Make up your mind that you will say both words, but leave it unsettled which you will say first ...if you have the rarest of gifts, a perfectly balanced mind, you will say "frumious".

Lewis Carroll, *Hunting Of The Snark*

A central feature of NLG is its creative aspect. Unlike NLU tasks such as dependency parsing [101], the output of a NLG system is far less circumscribed by its input. Whereas a dependency parser performs a transformation which results in information loss, generators have to perform transformations which require considerable addition of information beyond that provided in the input. For example, dependency parsing only has to flesh out the syntactic pairwise relation information, in contrast to a generator which has to produce an output text with information on multiple aspects — syntax, semantics, pragmatics, *inter alia*. Often, a generator has to make assumptions or develop its own set of arbitrary preferences with regard to some of these choice

points, since they are unstated in the input. In that sense, NLG is a creative process. Note that this does not mean that the subspace of acceptable outcomes (NLG outputs) is undefined or unconstrained, but that this can nevertheless be quite large with little further discrimination possible based on the input information alone. Good generators can innovatively and creatively employ the lexical, syntactic and other means at their disposal in creative ways to express the nuances of what is being communicated as per the communicative goal. This creativity starts at even the lowest granularity, lexical level, with this level providing an ideal platform to begin exploring issues concerning the same. In this chapter, we shall explore the dynamic generation of a class of lexical artifacts named *portmanteaus*.

Portmanteaus are a creative word formation phenomenon where two words blend to form a new word, with a meaning derived from but distinct to their original meanings e.g *wiki + etiquette* → *wikiquette*, *fashion + fascism* → *fashism*. Portmanteaus are a form of neologism. Portmanteau generation [38] can find potential use as a lower-level submodule in creative generation tasks. We devise character-level neural sequence-to-sequence (S2S) methods for the task of portmanteau generation that are end-to-end-trainable, language independent, and do not explicitly use additional phonetic information.

In this chapter, we focus on the *contribution type* of [Method]. We do not specify the actual overarching communicative intent of the system (*Why* it should create the portmanteau word - which would in a real world NLG application be a submodule occasionally invoked), but assume that its given. Instead we focus on *How* it can do so.

We devise a noisy-channel-style model that allows for the incorporation of unsupervised word lists, improving performance over a standard source-to-target model that directly learns a distribution of the form $P(y|x)$ without refactoring it in Bayesian terms. This model is made possible by an exhaustive candidate generation strategy specifically enabled by the features of the portmanteau task. Besides the forward factored model, our experiments also find our approach to be superior to a state-of-the-art FST-based baseline with respect to ground truth accuracy and human evaluation.

Specifically, the intervention to the E2E NLG pipeline required here is fleshed out in Figure 2.1. Importantly, our work also highlights the need for special sensitivity and carefully designed approaches towards long-tailed phenomena [11, 144], where otherwise neglected aspects (here phonetics) and paucity of data to learn from become important considerations. These phenomena typically manifest themselves at the realization level. It is critical to handle them adequately in order to maintain good worst case performance, which is an important consideration for making NLG systems deployable.

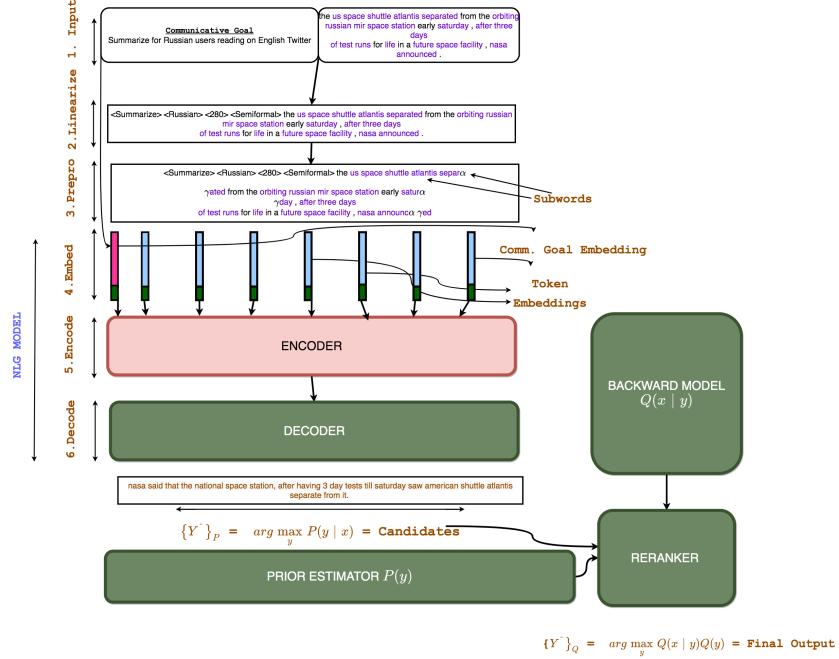


Figure 2.1: An illustration of how the *End-to-End Neural NLG Pseudo-Pipeline* or E2EN2PP fleshed out in Figure 1.2 would work in action for our actual generation task and input example, after incorporating the Intervention described in this Chapter. The forward E2EN2PP here merely acts as a candidate generator, with the three new introduced components – Prior Estimator, Backward Model and Reranker producing the final output distribution used to generate the Final Output (by reranking candidates)

2.1 Introduction

Portmanteaus (or lexical blends [3]) are novel words formed from parts of multiple root words in order to refer to a new concept that can't otherwise be expressed concisely. Portmanteaus have become frequent in modern-day social media, news reports and advertising, one popular example being *Brexit* (Britain + Exit). [159]. These are found not only in English but many other languages such as Bahasa Indonesia [36], Modern Hebrew [14, 15] and Spanish [161]. Their short length makes them ideal for headlines and brandnames. [57].

Some languages such as Japanese also have portmanteau-like structures, albeit with fairly regular rules of formation, taking away the need for a machine learning based approach. However, for English, unlike better-defined morphological phenomena such as inflection and derivation, portmanteau generation is not a typical regular phenomena with a well-agreed upon set

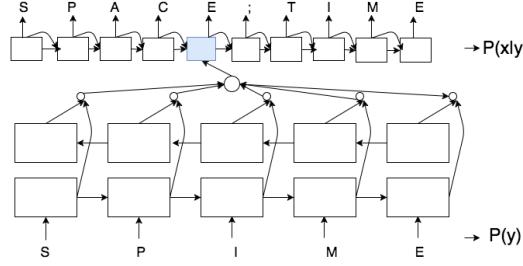


Figure 2.2: A sketch of our BACKWARD, noisy-channel model. The attentional S2S model with bidirectional encoder gives $P(x|y)$ and next-character model gives $P(y)$, where y (*spime*) is the portmanteau and $x = \text{concat}(x^{(1)}, ";", x^{(2)})$ are the concatenated root words (*space* and *time*).

of rules¹. For instance, [212] state that the composition of the portmanteau from its root words depends on several factors, two important ones being maintaining prosody and retaining character segments from the root words, especially the head. An existing work by [38] aims to solve the problem of predicting portmanteaus using a multi-tape FST model, which is data-driven, unlike prior approaches. Their methods rely on a grapheme to phoneme converter, which takes into account the phonetic features of the language, but may not be available or accurate for non-dictionary words, or low resource languages.

Prior works, such as [49], have demonstrated the efficacy of neural approaches for morphological tasks such as inflection. We hypothesize that such neural methods can (1) provide a simpler and more integrated end-to-end framework than multiple FSTs used in the previous work, and (2) automatically capture features such as phonetic similarity through the use of character embeddings, removing the need for explicit grapheme-to-phoneme prediction. To test these hypotheses, in this chapter, we devise a neural S2S model to predict portmanteaus given the two root words, specifically making 3 major contributions:

- We devise an S2S model that attends to the two input words to generate portmanteaus, and an additional improvement that leverages noisy-channel-style modelling to incorporate a language model over the vocabulary of words (§9.3.1).
- Instead of using the model to directly predict output character-by-character, we use the features of portmanteaus to exhaustively generate candidates, making scoring using the noisy channel model possible (§2.4).
- We curate and share a new and larger dataset of 1624 portmanteaus (§9.2).

¹This does not imply that one cannot come up with a set of rules by observing a sufficient number of portmanteau examples - but one is more likely to see a larger number of violations of these rules for newer examples than one would if portmanteaus were a regular phenomenon in the English language e.g pluralization.

In Experiments (§9.4), our model performs better than the baseline [38] on both objective and subjective measures, demonstrating that such methods can be used effectively in a morphological task.

2.2 Related Work

[Özbal and Strapparava](#) [148] generate new words to describe a product given its category and properties. However, their method is limited to hand-crafted rules as compared to our data driven approach. Also, their focus is on brand names. [Hiranandani et al.](#) have devised an approach to recommend brand names based on brand/product description. However, they consider only a limited number of features like memorability and readability. [Smith et al.](#) [217] devise an approach to generate portmanteaus, which requires user-defined weights for attributes like *sounding good*. Generating a portmanteau from two root words can be viewed as a S2S problem. Recently, neural approaches have been used for S2S problems [224] such as MT. [Ling et al.](#) [115] and [Chung et al.](#) [31] have shown that character-level neural sequence models work as well as word-level ones for language modelling and MT. [Zoph and Knight](#) [258] devise S2S models for multi-source MT, which have multi-sequence inputs, similar to our case.

2.3 Models

This section describes our neural models.

2.3.1 Forward Architecture

Under our first devised architecture, the input sequence $\mathbf{x} = \text{concat}(\mathbf{x}^{(1)}, ";", \mathbf{x}^{(2)})$, while the output sequence is the portmanteau \mathbf{y} . The model learns the distribution $P(\mathbf{y}|\mathbf{x})$.

The network architecture we use is an attentional S2S model [9]. We use a bidirectional encoder, which is known to work well for S2S problems with similar token order, which is true in our case. Let \overrightarrow{LSTM} and \overleftarrow{LSTM} represent the forward and reverse encoder; $e_{enc}()$ and $e_{dec}()$ represent the character embedding functions used by encoder and decoder. The following

equations describe the model:

$$\begin{aligned}
h_0^{\overrightarrow{enc}} &= \vec{0}, h_{|x|}^{\overleftarrow{enc}} = \vec{0} \\
h_t^{\overrightarrow{enc}} &= \overrightarrow{LSTM}(h_{t-1}^{enc}, e_{enc}(x_t)) \\
h_t^{\overleftarrow{enc}} &= \overleftarrow{LSTM}(h_{t+1}^{enc}, e_{enc}(x_t)) \\
h_t^{enc} &= h_t^{\overrightarrow{enc}} + h_t^{\overleftarrow{enc}} \\
h_0^{dec} &= h_{|x|}^{enc} \\
h_t^{dec} &= LSTM(h_{t-1}^{dec}, [\text{concat}(e_{dec}(y_{t-1}), c_{t-1})]) \\
p_t &= \text{softmax}(W_{hs}[\text{concat}(h_t^{dec}, c_t)] + b_s)
\end{aligned}$$

The context vector c_t is computed using dot-product attention over encoder states. We choose dot-product attention because it doesn't add extra parameters, which is important in a low-data scenario such as portmanteau generation.

$$\begin{aligned}
a_i^t &= \text{dot}(h_t^{dec}, h_i^{enc}), \alpha^t = \text{softmax}(a^t) \\
c_t &= \sum_{i=1}^{|x|} \alpha_i^t h_i^{enc}
\end{aligned}$$

In addition to capturing the fact that portmanteaus of two English words typically sound *English-like*, and to compensate for the fact that available portmanteau data will be small, we pretrain the character embeddings on English language words. We use character embeddings learnt using an LSTM language model over words in an English dictionary,² where each word is a sequence of characters, and the model will predict next character in sequence conditioned on previous characters in the sequence.

2.3.2 Backward Architecture

The second devised model uses Bayes's rule to reverse the probabilities $P(\mathbf{y}|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{y})P(\mathbf{y})}{P(\mathbf{x})}$ to get $\text{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \text{argmax}_{\mathbf{y}} P(\mathbf{x}|\mathbf{y})P(\mathbf{y})$. Thus, we have a reverse model of the probability $P(\mathbf{x}|\mathbf{y})$ that the given root words were generated from the portmanteau and a character language model model $P(\mathbf{y})$. This is a probability distribution over all character sequences

²Specifically in our experiments, 134K words from the CMU Pronouncing dictionary [236].

$y \in A^*$, where A is the alphabet of the language. This way of factorizing the probability is also known as a *noisy channel model*, which has recently also been shown to be effective for neural MT ([75], [248]). Such a model offers two advantages

1. The reverse direction model (or alignment model) gives higher probability to those portmanteaus from which one can discern the root words easily, which is one feature of good portmanteaus.
2. The character language model $P(\mathbf{y})$ can be trained on a large vocabulary of words in the language. The likelihood of a word y is factorized as $P(y) = \prod_{i=1}^{i=|y|} P(y_i | y_1^{i-1})$, where $y_j^i = y_i, y_{i+1} \dots y_j$, and we train a LSTM to maximize this likelihood.

2.4 Making Predictions

Given these models, we must make predictions, which we do by two methods

Greedy Decoding: In most neural sequence-to-sequence models, we perform auto-regressive greedy decoding, selecting the next character greedily based on the probability distribution for the next character at current time step. We refer to this decoding strategy as GREEDY.

Exhaustive Generation: Many portmanteaus were observed to be concatenation of a prefix of the first word and a suffix of the second. We therefore generate all candidate outputs which follow this rule. Thereafter we score these candidates with the decoder and output the one with the maximum score. We refer to this decoding strategy as SCORE.

Given that our training data is small in size, we expect ensembling [20] to help reduce model variance and improve performance. In this chapter, we ensemble our models wherever mentioned by training multiple models on 80% subsamples of the training data, and averaging log probability scores across the ensemble at test-time.

2.5 Dataset

The existing dataset by [38] contains 401 portmanteau examples from Wikipedia. We refer to this dataset as D_{Wiki} . Besides being small for detailed evaluation, D_{Wiki} is biased by being from just one source. We manually collect D_{Large} , a dataset of 1624 distinct English portmanteaus from following sources:

- Urban Dictionary³
- Wikipedia
- Wiktionary
- BCU’s Neologism Lists from ’94 to ’12.

Naturally, $D_{Wiki} \subset D_{Large}$. We define $D_{Blind} = D_{Large} - D_{Wiki}$ as the dataset of 1223 examples not from Wikipedia. We observed that 84.7% of the words in D_{Large} can be generated by concatenating prefix of first word with a suffix of the second.

2.6 Baseline

In this section, we shall concisely discuss the FST-based approach for our task, devised by [38]. We refer the reader to the original paper for a more detailed exposition.

The baseline approach from [38], illustrated in Figure 2.3, defines a pipeline of FSTs to progressively transform the root words $x^{(1)}$ and $x^{(2)}$ to the output y . It also requires the root words to have corresponding phoneme sequences based on the CMU Pronouncing Dictionary. The approach proceeds through the following steps.

1. Get the phoneme sequences of root words $x^{(1)}$ and $x^{(2)}$ from CMU Pronouncing Dictionary
2. FST A pretransforms the individual phoneme sequences before they are merged.
3. FST B , which has two input tapes (one for each root word) generates a merged phoneme sequence on its “output tape”, denoted by PM_{pron} .
4. FST C reads off this phoneme sequence and reconverts it to the space of graphemes/letters, with the output grapheme sequence being denoted PM' .
5. Finally, FST D and FST $E_{1,2}$ each do a round of successive “post-processing” of sorts, converting $PM' \rightarrow PM''$ and $PM' \rightarrow PM''$ respectively. FST $E_{1,2}$ has three input tapes, since it also reads in the two root words as inputs in addition to the current portmanteau sequence PM'' .
6. The final, output portmanteau produced by FST is denoted PM''' .

The first disadvantage of this approach is its dependence on converting root words to their phoneme sequences explicitly, which requires their presence in the CMU Pronouncing Dictionary. Though this is not an issue for root words in the D_{Wiki} , which are all covered by the dictionary, we find 6.36% of the root words in D_{Blind} missing from CMU Phonetic Dictionary.

³Not all neologisms are portmanteaus, so we manually choose those which are for our dataset.

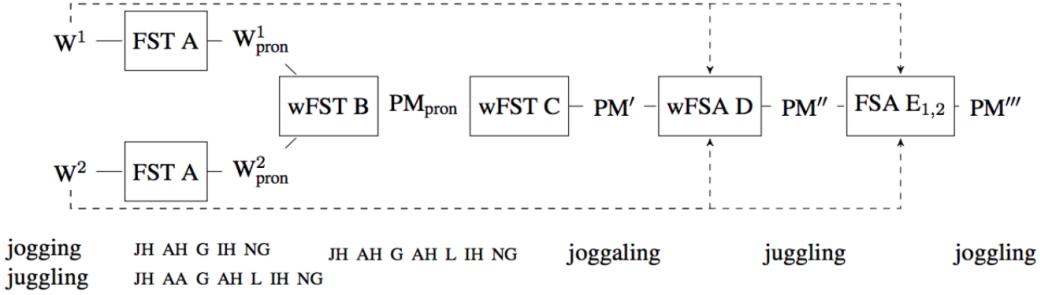


Figure 2.3: A sketch of the BASELINE FST-based pipeline approach from [38], starting with the input root words *jogging* and *juggling* to the left, leading to the final output, *joggaling* at the rightmost end. This approach requires both root words $x^{(1)}$ and $x^{(2)}$ to be present in the CMU Phonetic Dictionary to get the phonetic sequences for the first step, as shown.

Model	Attn	Ens	Init	Search	Matches	Distance
BASELINE	-	-	-	-	45.39%	1.59
FORWARD	✓	✗	✗	GREEDY	22.00%	1.98
	✓	✗	✓	GREEDY	28.00%	1.90
	✓	✗	✗	BEAM	13.25%	2.47
	✓	✗	✓	BEAM	15.25%	2.37
	✓	✗	✗	SCORE	30.25%	1.64
	✓	✗	✓	SCORE	32.88%	1.53
	✓	✓	✓	SCORE	42.25%	1.33
	✓	✓	✗	SCORE	41.25%	1.34
	✗	✗	✓	SCORE	6.75%	3.78
	✗	✗	✗	SCORE	6.50%	3.76
BACKWARD	✓	✗	✗	SCORE	37.00%	1.53
	✓	✗	✓	SCORE	42.25%	1.35
	✓	✓	✓	SCORE	48.75%	1.12
	✓	✓	✗	SCORE	46.50%	1.24
	✗	✗	✓	SCORE	5.00%	3.95

Table 2.1: 10-Fold Cross-Validation results, D_{Wiki} . *Attn*, *Ens*, *Init* denote attention, ensembling, and initializing character embeddings respectively.

2.7 Experiments

In this section, we show results comparing various configurations of our model to the baseline FST model of [38] (BASELINE). Models are evaluated using exact-matches (*Matches*) and average Levenshtein edit-distance (*Distance*) w.r.t ground truth.

2.7.1 Objective Evaluation Results

In *Experiment 1*, we follow the same setup as [38]. D_{Wiki} is split into 10 folds. Each fold model uses 8 folds for training, 1 for validation, and 1 for test. The average (10 fold cross-validation style approach) performance metrics on the test fold are then evaluated. *Table 3.3* shows the results of *Experiment 1* for various model configurations. We get the BASELINE numbers from [38]. Our best model obtains 48.75% *Matches* and 1.12 *Distance*, compared to 45.39% *Matches*

Model	Attn	Ens	Init	Search	Matches	Distance
BASELINE	-	-	-	-	31.56%	2.32
FORWARD	✓	✗	✓	SCORE	25.26%	2.13
	✓	✗	✗	SCORE	24.93%	2.32
	✓	✓	✗	SCORE	31.23%	1.98
	✓	✓	✓	SCORE	28.94%	2.04
BACKWARD	✓	✗	✓	SCORE	25.75%	2.14
	✓	✗	✗	SCORE	25.26%	2.17
	✓	✓	✗	SCORE	31.72%	1.96
	✓	✓	✓	SCORE	32.78%	1.96

Table 2.2: Results on D_{Blind} (1223 Examples). In general, BACKWARD architecture performs better than FORWARD architecture.

and 1.59 *Distance* using BASELINE.

For *Experiment 2*, we seek to compare our best approaches from *Experiment 1* to the BASELINE on a large, held-out dataset. Each model is trained on D_{Wiki} and tested on D_{Blind} . BASELINE was similarly trained only on D_{Wiki} , making it a fair comparison. Table 2.2 shows the results⁴. Our best model gets *Distance* of 1.96 as compared to 2.32 from BASELINE.

We observe that the *Backward* architecture performs better than *Forward* architecture, confirming our hypothesis in §2.3.2. In addition, ablation results confirm the importance of attention, and initializing the word embeddings. We believe this is because portmanteaus have high fidelity towards their root word characters and its critical that the model can observe all root sequence characters, which attention manages to do as shown in Fig. 3.2.

Performance on Uncovered Examples

The set of candidates generated before scoring in the approximate SCORE decoding approach sometimes do not include the ground truth. Some such uncovered examples are *precise+exactly* → *prezactly* and *puke+extravaganza* → *pukestravaganza*. This holds true for 229 out of 1223 examples in D_{Blind} . We compare the FORWARD approach along with a GREEDY decoding strategy to the BASELINE approach for these examples.

Both FORWARD+GREEDY and the BASELINE get 0 *Matches* on these examples. The *Distance* for these examples is 4.52 for BASELINE and 4.09 for FORWARD+GREEDY. Further, a spot checked inspection for a randomly chosen subsample also confirms example outputs from both the approaches to be of comparable quality. Hence, we see that one of our approaches

⁴For BASELINE [38], we use the model from <http://leps.isi.edu/fst/step-all.php>



Figure 2.4: Attention matrices while generating *slurve* from *slider;curve*, and *bennifer* from *ben;jennifer* respectively, using *Forward* model. ; and . are separator and stop characters. Darker cells are higher-valued

(FORWARD+GREEDY) stands at par with the BASELINE even for these examples.

2.7.2 Significance Tests

Since our dataset is still small relatively small (1223 examples), it is essential to verify whether BACKWARD is indeed statistically significantly better than BASELINE in terms of *Matches*.

In order to do this, we use a paired bootstrap⁵ comparison [98] between BACKWARD and BASELINE in terms of *Matches*. BACKWARD is found to be better (gets more *Matches*) than BASELINE in 99.9% ($p = 0.999$) of the subsets.

Similarly, BACKWARD has a lower *Distance* than BASELINE by a margin of 0.2 in 99.5% ($p = 0.995$) of the subsets.

2.7.3 Subjective Evaluation and Analysis

On inspecting outputs, we observed that often output from our system seemed good in spite of high edit distance from ground truth. Such aspect of an output *seeming good* is not captured satisfactorily by measures like edit distance. To compare the errors made by our model to the baseline, we designed and conducted a human evaluation task on AMT.⁶ In the survey, we show human annotators outputs from our system and that of the baseline. We ask them to judge which alternative is *better* overall based on following criteria: 1. It is a good shorthand for two

⁵We average across $M = 1000$ randomly chosen subsets of D_{Blind} , each of size $N = 611$ ($\approx 1223/2$)

⁶We avoid ground truth comparison because annotators can be biased to ground truth due to its existing popularity.

Input	FORWARD	BACKWARD	BASELINE	G.TRUTH
shopping;marathon	shopparathon	shoathon	shon	shopathon
fashion;fascism	fashism	fashism	fashism	fashism
wiki;etiquette	wikiquette	wiquette	wiquette	wikiquette
clown;president	clowident	clownsident	clownt	clownsident
car;hijack	carjack	carjack	cack	carjack
dialectical;materialism	dialerrialism	dialerrialism	dialism	dialerrialism
tinder;disaster	tinter	tindersaster	tindisaster	tindisaster
data;broadcasting	datasting	doadcasting	dating	datacasting
back;acronym	backronym	bacronym	bacronym	backronym
britain;regret	bregret	brigret	bregret	bregret
social;entertainer	socialtainer	sociartainer	sentertainer	socialtainer
chopstick;fork	chopstork	chopfork	chork	chork
happy;harmonius	happonius	happonius	hharmonius	happymonius
flexible;vegetarian	flexarian	flexetarian	fegetarian	flexitarian
laughter;orgasm	lauggasm	laughtergasm	lasm	laughgasm
frequency;recency	frecenty	frecenty	frecenty	frecenty
tender;enterpreneur	tenpreneur	tendereneur	tenterpreneur	tenderpreneur
fall;halloween	falloween	falloween	falloween	falloween
frisbee;golf	frolf	frisbolf	frolf	frolf
hitler;hillary	hitlary	hitlery	hitlery	hitlery
trump;economics	trumpics	trumponomics	trumics	trumponomics
flirtation;relationship	flirtionship	flirtationship	flirtationship	flirtationship
next;yesterday	nexterday	nesterday	nexterday	nexterday
lobster;monstrosity	lobstrosity	lonstrosity	lobstrosity	lobstrosity
global;english	glonglish	globlish	glish	globlish
puke;extravaganza	pukaganza	pukaganza	pueextravaganza	pukestravaganza
beverage;avalanche	bevalanche	beveranche	bavalanche	bevelanche
excited;dimmer	excimmer	excimmer	excitedimmer	excimmer
phone;amnesia	phonesia	phonesia	phonesia	phonesia
camera;phone	came	camphone	camphone	camphone
bored;ordinary	bordinary	bordinary	bordinary	bordinary
precise;exactly	prexactly	prexactly	prexactly	prezactly

Table 2.3: Example outputs from different models. Outputs are from best performing configurations of the models. G.TRUTH denotes the ground truth portmanteau.

original words 2. It sounds better. We requested annotation on a scale of 1-4. To avoid ordering bias, we shuffled the order of two portmanteau between our system and that of baseline. We restrict annotators to be from Anglophone countries, have HIT Approval Rate > 80% and pay 0.40\$ per HIT (5 Questions per HIT). We had 2 annotations per question and considered each annotation as a separate judgement, without doing per-example aggregation. Nevertheless, the two annotations showed a reasonably high Pearson correlation of 0.6191.

As seen in Table 2.4, output from our system was labelled better by humans as compared to the baseline 58.12% of the time. Table 2.3 shows outputs from different models for a few examples.

Judgement	Percentage of total
Much Better (1)	29.06
Better (2)	29.06
Worse (3)	25.11
Much Worse (4)	16.74

Table 2.4: AMT annotator judgements on whether our system’s proposed portmanteau is better or worse compared to the baseline

2.8 Conclusion

In this chapter, we explored a problem involving NLG as a creative process at the lexical level, within the larger *Realization* stage of the CNP. Our contribution was primarily along the *contribution type* [Method], and our devised approach necessitated Intervention to the typical E2EN2PP as illustrated in Figure 2.1.

Specifically, we devised an end-to-end neural system to model portmanteau generation. Our experiments show the efficacy of devised system in predicting portmanteaus given the root words. Our methods can be learnt without using external phonetic resources, and learn from existing list of portmanteaus and word types.

We conclude that pretraining character embeddings on the English vocabulary helps the model. When we additionally incorporate a character-level next-character prediction module pretrained on word types through a prior, we are able to outperform earlier state-of-the-art models based on explicit phonetic knowledge from [38].

This shows that the Intervention performed to the E2EN2PP as shown in Figure 2.1, leading to a departure from its end-to-end nature, was indeed justified and benefited end-task performance.

Through human evaluation we show that our model’s predictions are superior to the baseline. We have also released our dataset and code⁷ to encourage further research on the phenomenon of portmanteaus. An obvious extension to our work is to try similar models on multiple languages.

Broader Takeaways

First, let us consider how an architecture similar to ours can be devised for another NLG task involving a creative aspect at the lexico-phonetic level. Note that we will only provide a high-level sketch of the task herein.

⁷<https://github.com/vgtomahawk/Charmanteau-CamReady>

Consider the task of generating quasi-punny brand names given a seed brand name ngram x and a topic z . This task was one of the subtypes of creative brand naming studied in [148]. As an example, consider a seed brand name $x = \text{Thanks a lot!}$ and the topic $z = \text{defense manufacturing}$. A potential punny brandname $y = \hat{y}$ is Tanks a lot! . Note that the output brand name y need not be restricted to be a word-substituted variant of x . For instance, if we had topic $z = \text{movie tickets}$, a possible output would have been $y = \hat{y} = \text{Thanks allot!}$. From this example, we can see that we need to model x , y and z as sequences of characters, just like the architectures we used for portmanteau generation. Furthermore, a character-level factoring of models allows the consideration of phonetics implicitly to a certain extent, which is not possible with a word or subword-level factoring. This is a property even we exploit for our portmanteau generation task.

Finally, let us consider how we can differently refactor the distribution $P(y|x, z)$ to open up the possibility of pretraining certain distribution components using unsupervised data sources, beyond the small number of completely supervised gold triples $D_{gold} = \{(x_i, z_i, y_i^*)\}_{i=1}^{i=|D_{gold}|}$ available for training, just like we used a Bayes Rule based refactoring to allow pretraining using all word types in the English vocabulary for portmanteau generation earlier in this chapter.

$$\begin{aligned}\operatorname{argmax}_y P(y|x, z) &= \operatorname{argmax}_y \frac{P(x, z, y)}{P(x, z)} \\ &= \operatorname{argmax}_y P(x, z, y) \\ &= \operatorname{argmax}_y P(x|z, y)P(z|y)P(y)\end{aligned}$$

Since x is conditionally independent of z given y ,

$$= \operatorname{argmax}_y P(x|y)P(z|y)P(y)$$

We'll now describe how each of the three distributional components can be pretrained using respective unsupervised data sources based on their availability.

1. $P(x|y)$ can be pretrained using a list of homophonic word pairs or multiword expression pairs.
2. $P(z|y)$ can be pretrained using any list of keyword-sentence or keyword-expression pairs. It is possible to construct such lists using off-the-shelf keyword extraction algorithms such as YAKE [23] and applying them to any collection of English sentences/phrases.
3. $P(y)$ can be pretrained using any list of ngrams/phrases.

Note that these possibilities for pretraining get opened up only as a result of the refactoring.

Having pretrained these components, we can finally finetune all of them jointly on the fully supervised training data D_{gold} .

Second, what are the other larger takeways from our chapter for NLG tasks involving a creative aspect in general?

One clear takeaway we can see is that for creative NLG tasks in general, it is always meaningful to consider alternative refactoring schemes for the probability distribution, rather than jumping directly to explicitly modelling $P(y|x)$ using a source-to-target sequence-to-sequence model, where the output y is the output text (whose generation involves creative aspects) and x is the input. The reason underlying this is that since creative artifacts are rarely produced, the number of full, paired examples $\{x^i, y^i\}$ tends to be small e.g there are atmost about 2000 unique portmanteaus and 1000 unique tongue twisters present in English text on the Web. The explicit source-to-target formulation is locked in to be learnt from such paired data exclusively, making it sensitive to the aforementioned shortage. Training solely on such limited data can also make further amplify the learnt model's tendency to replicate portions of the training data at test-time [128], which is especially a disadvantage for tasks with a creative aspect to them.

Alternative refactoring schemes can decompose $P(y|x)$ into two or more components, some of which can be potentially pretrained on unsupervised or distantly supervised data, depending on the specifics of the refactoring as well as the task under consideration. For instance, in this chapter, we use Bayes Rule to refactor $P(y|x)$ as $\frac{P(x|y)P(y)}{P(x)}$. As a result, we were able to pretrain $P(y)$ using all word types in the English vocabulary.

This was of course, only one specific refactoring scheme - a variety of other candidate refactoring schemes can be considered based on the generation task at hand. For instance, introducing an intermediate hidden variable z based on properties of the task to refactor $P(y|x) = \sum_z P(y|z)P(z|x)$, where an approximate set of "silver" z values is derivable from x using some task-specific heuristic.

We will now sketch an example of another creative NLG task and a possibly beneficial alternative refactoring scheme for it. Consider the task of abductively generating intermediate hypotheses given a precondition x and a post-condition z ⁸ e.g $x = \text{"Stephen left his home in a rush after waking up late, and forgot to close the windows."}$, $z = \text{"On returning home at night, Stephen found the house in a meddled state with his belongings strewn around and water and mud spilt over the floor"}$. Several plausible intermediate hypotheses can be generated, such as $y_1 = \text{"A mid afternoon burst of wind and torrential rain got into his home, throwing unlocked cabinets"}$

⁸This can also be viewed as three-sentence story generation given a pre-prompt and post-prompt

open and blowing around his clothes and accessories", $y_2 = \text{"A pack of monkeys got in through the window at 3pm, raiding and messing up every corner of the house while Stephen was away"}$ etc. We have very limited examples with gold intermediate hypotheses i.e $D_{gold} = \{(x_i, z_i, y_i)\}_{i=1}^{i=|D_{gold}|}$, although we do have access to a large number of approximate pre-condition/post-condition pairs derived from consecutive sentences of the form $D_{silver} = (x_{approx} = a_i, y_{approx} = b_i)$ ⁹.

The most obvious, source-to-target formulation for this would be to learn $P(y|x, z)$. However, application of Bayes rule can lead us to a potentially better refactoring scheme for the distribution. Using Bayes rule, we first get $P(y|x, z) = \frac{P(x, y, z)}{P(x, z)}$. Further refactoring the numerator, we get $P(y|x, z) = \frac{P(z|y, x)P(y|x)P(x)}{P(z|x)P(x)}$. Since we are only interested in the most likely hypothesis $\hat{y} = \text{argmax } P(y|x, z)$, it would be sufficient to learn $P(z|y, x)P(y|x)P(x)$. Now, we can see that one of these components can be pretrained using examples from D_{silver} , in the same way we pretrained one of our model components in this chapter using word types from the English vocabulary. Specifically, we can pretrain $P(y|x)$ using $y = y_{approx} = b_i$ and $x = x_{approx} = a_i$. We can also pretrain $P(x)$ using $x = x_{approx} = a_i$.

To conclude, in this subsection, we saw how our learning architecture can be adopted and/or provide takeaways/lessons for a) Specifically, for lexico-phonetic generation tasks with a creative aspect b) More generally, for any NLG task with a creative aspect to it.

Acknowledgements

We thank Dongyeop Kang, David Mortensen, Qinlan Shen and anonymous reviewers for their valuable comments. This research was supported in part by DARPA grant FA8750-12-2-0342 funded under the DEFT program.

⁹Since we are only sketching out this task here, we do not delve too much into the specifics of the sources such approximate unsupervised data can be derived. Another source can be external textual entailment datasets which are more readily available, from where we can use the premise and hypothesis as x_{approx} and y_{approx} respectively.

Chapter 3

Stylistic Surface Transduction To Shakespearize Modern English (EMNLP 2017'WS)

[Knowledge][Method]

Most of us in speaking and writing English use only one pronoun of address; we say ‘you’ to many persons and ‘you’ to one person. The pronoun ‘thou’ is reserved, nowadays, to prayer and naive poetry, but in the past it was the form of familiar address to a single person. At that time ‘you’ was the singular of reverence and of polite distance, and also the invariable plural.

R.Brown & A. Gilman, ‘*The Pronouns of Power & Solidarity*’, 1960

As users get ever more habituated to using, interacting and even co-authoring with NLG systems, there is increasing expectation on them to exhibit *consistent personality* [222] and also be *accommodative* [216] towards *user preferences* and *situation of use*. Together, one can think of these as aspects of *target style*. Hence, NLG systems should be able to transfer their content

to match aspect values for each aspect of target style. From the perspective of Halliday's SFL, style transfer can be seen as having the changing of *extra-textual aspects* as its communicative goal, while keeping the textual aspect constant - these aspects could be either *interpersonal* or *ideational* in nature. *Diachronic register* of language is one example of an *interpersonal* aspect, being determined by the author of a text as well as the historical period in which the author resides. Specifically, the term *diachronic* refers to the perspective of language as a changing variable that evolves through *chronos* i.e time. The diachronic register of a language is simply denotes its state at a given period in history. For instance, consider the two variants *I stand on sudden haste* and *I am in a rush*. Though a native English speaker would likely understand both, and understand both of these to mean the same thing in ideational terms, they would certainly find the first one a strange way of conveying the same idea, and rightly so, for it is how Shakespeare originally said it, with the latter one being a Modern English paraphrase from Sparknotes.com

Though traditionally often viewed as the least challenging level in terms of reasoning and knowledge, surface level language abstractions can nonetheless prove a rich source to motivate task-specific knowledge and inductive biases. Moreover, by explicitly incorporating such biases and easing the learning of surface realization level transformations, we make it easier for the model to learn higher-order sentential (microplanning-level) and extra-sentential (macroplanning-level) biases

Though traditionally often viewed as the least challenging level in terms of reasoning and knowledge, surface level language abstractions can nonetheless prove a rich source to motivate task-specific knowledge and inductive biases. Moreover, by explicitly incorporating such biases and easing the learning of surface realization level transformations, we make it easier for the model to learn higher-order sentential (microplanning-level) and extra-sentential (macroplanning-level) biases

Through this chapter, we present an example of how *knowledge* and *inductive bias* emanating from the surface realization level can aid neural S2S models for diachronic style transfer.

- i) We explicitly provide source → target surface realization correspondences through incorporating dictionary knowledge into word representations. The "word representation" here is the shared, jointly pretrained space in which both source and target words are embedded; we also empirically show that having a shared embedding space is better than having separate ones helps, as also that joint pretraining is beneficial.
- ii) We furnish the model with the ability to copy over words besides generating them tabula rasa, based on the inductive bias

Specifically, we explore automated methods to transform text from modern English to Shakespearean English using an end to end trainable neural model with pointers to enable copy action. To ameliorate the deficient learning of embeddings due to a limited amount of parallel data, we pretrain embeddings of words by leveraging external dictionaries mapping Shakespearean words to modern English words as well as additional text. Our methods are able to get a BLEU score of 31+, an improvement of ≈ 6 points over the strongest baseline. We publicly release our code to foster further research in this area.¹

3.1 Introduction

Given a source text, human speakers/authors/content moderators often morph it further using a variety of lexical and grammatical transformations, adjusting the degree of formality, usage of catchy phrases, and other such stylistic changes; with non-textual subgoals in mind e.g to make it more appealing. For instance, assuming a “make more appealing” subgoal, different text styles appeal to different target user segments [196] [96] [205]. Millennials may find text using social media slangs to be more appealing, while middle-aged New Yorkers may find the inclusion of Yiddish and Italian slang to be so. Thus there is a need to effectively adapt text to different target styles. However, manually transforming text to a desired style can be a tedious process.

There have been increased efforts towards machine assisted text content creation and editing through automated methods for summarization [194], brand naming [74], text expansion [221], etc. However, there is a dearth of automated solutions for adapting text quickly to different styles. We consider the problem of transforming text written in modern English text to Shakespearean style English. For the sake of brevity and clarity of exposition, we henceforth refer to the *Shakespearean* sentences/side as *Original* and the modern English paraphrases as *Modern*.

Unlike more traditional domain or style transfer settings e.g., formality, our task is distinguished by the fact that the two styles employ diachronically disparate registers of English - one style uses the contemporary language while the other uses *Early Modern English*² from the *Elizabethan Era* (1558-1603). Although *Early Modern English* is not classified as a different language (unlike *Old English* and *Middle English*), it does have novel words (*acknow* and *belike*), novel grammatical constructions (two *second person* forms - *thou* (informal) and *you* (formal)

¹<https://github.com/harsh19/Shakespearizing-Modern-English>

²https://en.wikipedia.org/wiki/Early_Modern_English

No	Type	Text
1	MODERN	Oh my, my bones ache so much
	ORIGINAL	Fie, how my bones ache !
	COPY	fie, how my bones ache !
	SIMPLES2S	you'll be, sir, what the bones are tired .
	STAT	Oh my, my bones ache so much .
2	MODERN	I am in a rush .
	ORIGINAL	I stand on sudden haste .
	COPY	i stand on sudden haste .
	SIMPLES2S	i'm stand right here .
	STAT	I am in a Fly
3	MODERN	Give my compliments to your lady
	ORIGINAL	Commend me to thy lady
	COPY	commend me to your lady
	SIMPLES2S	give my regards to your lady
	STAT	give my praises to your lady
4	MODERN	Showing mercy by pardoning killers only causes more murders .
	ORIGINAL	Mercy but murders, pardoning those that kill .
	COPY	mercy but murders, those those who kill us .
	SIMPLES2S	but except the murderers to those murderers to kill you .
	STAT	of mercy by pardoning killers causes more dire.
5	MODERN	Holy Saint Francis, this is a drastic change !
	ORIGINAL	Holy Saint Francis, what a change is here !
	COPY	holy saint francis, what a change is here !
	SIMPLES2S	it's the holy flute, what's the changed !
	STAT	Holy Saint Francis, this is a drastic change !
6	MODERN	was that my father who left here in such a hurry ?
	ORIGINAL	Was that my father that went hence so fast ?
	COPY	was that my father that went went so fast ?
	SIMPLES2S	was that my father was so that ?
	STAT	was that my father that left here in such a haste ?
7	MODERN	Give me one kiss and I'll go down .
	ORIGINAL	One kiss, and I'll descend .
	COPY	one kiss me, and I'll descend .
	SIMPLES2S	one kiss, and I come down .
	STAT	Give me a kiss, and I'll go down .
8	MODERN	then the window lets day in, and life goes out the window .
	ORIGINAL	Then, window, let day in and life out .
	COPY	then, window out, and day life .
	SIMPLES2S	then she is just a life of life, let me life out of life .
	STAT	then the window will let day in, and life out .

Table 3.1: Examples from dataset showing modern paraphrases (MODERN) from the learning resource Sparknotes.com of few sentences from Shakespeare’s plays (ORIGINAL). We also show transformation of modern text to Shakespearean text from our models (COPY, SIMPLES2S and STAT).

[21]), semantically drifted senses (e.g. *fetches* is a synonym of *excuses*) and non-standard orthography [177]. Additionally, there is a domain difference since the Shakespearean play sentences are from a dramatic screenplay whereas the *parallel* modern English sentences are meant to be simplified explanation for high-school students.

Prior works in this field leverage a language model for the target style, achieving transformation either using phrase tables [246], or by inserting relevant adjectives and adverbs [196]. Such works have limited scope in the type of transformations that can be achieved. Firstly, it is difficult for human curated canned phrase table resources to cover the combinatorially increasing number of phrase pairs. Secondly, phrase tables cannot take context into account when doing phrase → phrase replacements. Moreover, statistical and rule MT based systems do not provide a direct mechanism to a) share word representation information between source and target sides b) incorporating constraints between words into word representations in end-to-end fashion. Neural sequence-to-sequence models, on the other hand, provide such flexibility. They provide direct mechanisms to handle all of these — Sharing source and target embeddings to share word-representation information, pretraining to leverage external information, and adding constraints to word representations using [48].

Our main contributions are as follows:

- We use a sentence level sequence to sequence neural model with a pointer network component to enable direct copying of words from input. We demonstrate that this method performs much better than prior phrase translation based approaches for transforming *Modern English* text to *Shakespearean English*.
- We leverage a dictionary providing mapping between Shakespearean words and modern English words to retrofit pre-trained word embeddings. Incorporating this extra information enables our model to perform well in spite of small size of parallel data.

The rest of the chapter is organized as follows. We first provide a brief analysis of our dataset in (§3.2). We then elaborate on details of our methods in (§3.3, §3.4, §3.5, §3.6). We then discuss experimental setup and baselines in (§3.7). Thereafter, we discuss the results and observations in (§3.8). We conclude with discussions on related work (§3.9) and future directions (§3.10).

3.2 Dataset

Our dataset is a collection of line-by-line modern paraphrases for 16 of Shakespeare’s 36 plays (*Antony & Cleopatra*, *As You Like It*, *Comedy of Errors*, *Hamlet*, *Henry V* etc) from the educational

	<i>Original</i>	<i>Modern</i>
# Word Tokens	217K	200K
# Word Types	12.39K	10.05K
Average Sentence Length	11.81	10.91
Entropy (Type.Dist)	6.15	6.06
\cap Word Types	6.33K	

Table 3.2: Dataset Statistics

site *Sparknotes*³. This dataset was compiled by Xu et al. [245, 246] and is freely available on github.⁴ 14 plays covering 18,395 sentences form the training data split. We kept 1218 sentences from the play *Twelfth Night* as validation data set. The last play, *Romeo and Juliet*, comprising of 1462 sentences, forms the test set.

3.2.1 Examples

Table 3.1 shows some parallel pairs from the test split of our data, along with the corresponding target outputs from some of our models. *Copy* and *SimpleS2S* refer to our best performing attentional S2S models with and without a *Copy* component respectively. *Stat* refers to the best statistical machine translation baseline using off-the-shelf GIZA++ aligner and MOSES. We can see through many of the examples how direct copying from the source side helps the *Copy* generates better outputs than the *SimpleS2S*. The approaches are described in greater detail in (§3.3) and (§3.7).

3.2.2 Analysis

Table 3.2 shows some statistics from the training split of the dataset. In general, the *Original* side has longer sentences and a larger vocabulary. The slightly higher entropy of the *Original* side’s frequency distribution indicates that the frequencies are more spread out over words. Intuitively, the large number of shared word types indicates that sharing the representation between *Original* and *Modern* sides could provide some benefit.

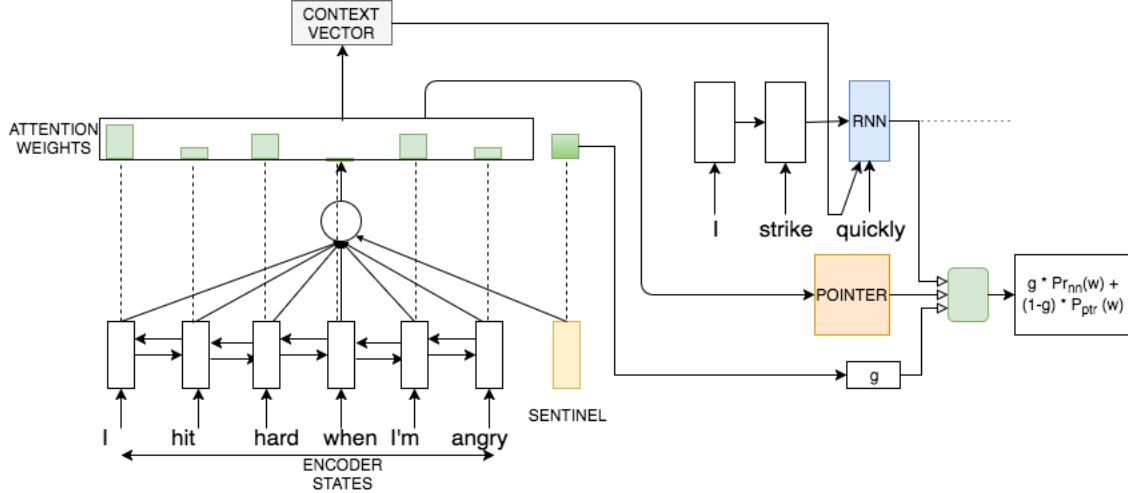


Figure 3.1: Depiction of our overall architecture (showing decoder step 3). Attention weights are computed using previous decoder hidden state h_2 , encoder representations, and sentinel vector. Attention weights are shared by decoder RNN and pointer models. The final probability distribution over vocabulary comes from both the decoder RNN and the pointer network. Similar formulation is used over all decoder steps

3.3 Method Overview

The Overall architecture of our system is shown in Figure 3.1. We use a bidirectional LSTM to encode the input modern English sentence. Our decoder side model is a mixture model of RNN module and pointer network module. The two individual modules share the attention weights over encoder states, although it is not necessary to do so. The decoder RNN predicts probability distribution of next word over the vocabulary, while pointer model predicts probability distribution over words in input. The two probabilities undergo a weighted addition, the weights themselves computed based on previous decoder hidden state and the encoder outputs.

Let x, y be the some input - output sentence pair in the dataset. Both input x as well as output y are sequence of tokens. $x = x_1 x_2 \dots x_{T_{enc}}$, where T_{enc} represents the length of the input sequence x . Similarly, $y = y_1 y_2 \dots y_{T_{dec}}$. Each of x_i, y_j is a token from the vocabulary.

3.4 Token embeddings

Each token in vocabulary is represented by a M dimensional embedding vector. Let vocabulary V be the union of modern English and Shakespearean vocabularies i.e. $V = V_{shakespeare} \cup V_{modern}$.

³www.sparknotes.com

⁴<http://tinyurl.com/ycdd3v6h>

E_{enc} and E_{dec} represent the embedding matrices used by encoder and decoder respectively ($E_{enc}, E_{dec} \in \mathbb{R}^{|V| \times M}$). We consider union of the vocabularies for both input and output embeddings because many of the tokens are common in two vocabularies, and in the best performing setting we share embeddings between encoder and decoder models. Let $E_{enc}(t)$, represent encoder side embeddings of some token t . For some input sequence \mathbf{x} , $E_{enc}(\mathbf{x})$ is given as $(E_{enc}(\mathbf{x}_1), E_{enc}(\mathbf{x}_2), \dots)$.

3.4.1 Pretraining of embeddings

Learning token embeddings from scratch in an end-to-end fashion along with the model greatly increases the number of parameters. To mitigate this, we consider pretraining of the token embeddings. We pretrain our embeddings on all training sentences. We also experiment with adding additional data from PTB [127] for better learning of embeddings. Additionally we leverage a dictionary mapping tokens from Shakespearean English to modern English.

We consider four distinct strategies to train the embeddings. In the cases where we use external text data, we first train the embeddings using both the external data and training data, and then for the same number of iterations on training data alone, to ensure adaptation. Note that we do not directly use off-the-shelf pretrained embeddings such as *Glove* [157] and *Word2Vec* [137] since we need to learn embeddings for novel word forms (and also different word senses for extant word forms) on the *Original* side.

Plain

This method is the simplest pre-training method. Here, we do not use any additional data, and train word embeddings are trained on the union of *Modern* and *Original* sentences.

PlainExt

In this method, we add all the sentences from the external text source (*PTB*) in addition to sentences in training split of our data.

Retro

We leverage a dictionary L of approximate *Original* \rightarrow *Modern* word pairs [245, 246], crawled from shakespeare-words.com, a source distinct from Sparknotes. We explicitly add the two *2nd persons* and their corresponding forms (thy, thou, thyself etc) which are very frequent

but not present in L . The final dictionary we use has 1524 pairs. Faruqui et al [48] proposed a *retrofitting* method to update a set of word embeddings to incorporate pairwise similarity constraints. Given a set of embeddings $p_i \in P$, a vocabulary V , and a set C of pairwise constraints (i, j) between words, retrofitting tries to learn a new set of embeddings $q_i \in Q$ to minimize the following objective:

$$f(Q) = \delta \sum_{i=1}^{|V|} (p_i - q_i)^2 + \omega \sum_{(i,j) \in C} (q_i - q_j)^2 \quad (3.1)$$

We use their off-the-shelf implementation ⁵ to encode the dictionary constraints into our pre-trained embeddings, setting $C = L$ and using suggested default hyperparameters for δ , ω and number of iterations.

RetroExt

This method is similar to *Retro*, except that we use sentences from the external data (*PTB*) in addition to training sentences.

We use **None** to represent the settings where we do not pretrain the embeddings.

3.4.2 Fixed embeddings

Fine-tuning pre-trained embeddings for a given task may lead to *overfitting*, especially in scenarios with small amount of supervised data for the task [123]. This is because embeddings for only a fraction of vocabulary items get updated, leaving the embeddings unchanged for many vocabulary items. To avoid this, we consider fixed embeddings pretrained as per procedures described earlier. While reporting results in Section (§3.8), we separately report results for fixed (*FIXED*) and trainable (*VAR*) embeddings, and observe that keeping embeddings fixed leads to better performance.

3.5 Method Description

In this section we give details of the various modules in the devised neural model.

⁵github.com/mfaruqui/retrofitting

3.5.1 Encoder model

Let $\overrightarrow{LSTM}_{enc}$ and $\overleftarrow{LSTM}_{enc}$ represent the forward and reverse encoder. $\mathbf{h}_t^{\overrightarrow{enc}}$ represent hidden state of encoder model at step t ($\mathbf{h}_t^{\overrightarrow{enc}} \in \mathbb{R}^H$). The following equations describe the model:

$$\mathbf{h}_0^{\overrightarrow{enc}} = \overrightarrow{0}, \mathbf{h}_{|x|}^{\overleftarrow{enc}} = \overrightarrow{0} \quad (3.2)$$

$$\mathbf{h}_t^{\overrightarrow{enc}} = \overrightarrow{LSTM}_{enc}(\mathbf{h}_{t-1}^{enc}, E_{enc}(\mathbf{x}_t)) \quad (3.3)$$

$$\mathbf{h}_t^{\overleftarrow{enc}} = \overleftarrow{LSTM}_{enc}(\mathbf{h}_{t+1}^{enc}, E_{enc}(x_t)) \quad (3.4)$$

$$\mathbf{h}_t^{enc} = \mathbf{h}_t^{\overrightarrow{enc}} + \mathbf{h}_t^{\overleftarrow{enc}} \quad (3.5)$$

We use addition to combine the forward and backward encoder states, rather than concatenation that is standardly used, since it doesn't add extra parameters, which is important in a low-data scenario such as ours.

3.5.2 Attention

Let \mathbf{h}_t^{dec} represent the hidden state of the decoder LSTM at step t . Let $E_{dec}(\mathbf{y}_{t-1})$ represent the decoder side embeddings of previous step output. We use special *START* symbol at $t = 1$.

We first compute a query vector, that is a linear transformation of \mathbf{h}_{t-1}^{dec} . A sentinel vector $\mathbf{s} \in \mathbb{R}^H$ is concatenated with the encoder states to create $F_{att} \in \mathbb{R}^{(T_{enc}+1) \times H}$, where T_{enc} represents the number of tokens in encoder input sequence \mathbf{x} . A normalized attention weight vector $\boldsymbol{\alpha}^{norm}$ is computed. The value g , which corresponds to attention weight over sentinel vector, represents the weight given to the decoder RNN module while computing output probabilities.

$$\mathbf{q} = \mathbf{h}_{t-1}^{dec} W_q \quad W_q \in \mathbb{R}^{H \times H} \quad (3.6)$$

$$F_{att} = concat(\mathbf{h}_{1..T_{enc}}^{enc}, \mathbf{s}) \quad F_{att} \in \mathbb{R}^{(T_{enc}+1) \times H} \quad (3.7)$$

$$\boldsymbol{\alpha}_i = \sum_{j=1}^H (\tanh(F_{att}^{(ij)} \mathbf{q}_j)) + \mathbf{b}_i \quad \boldsymbol{\alpha}_i, \mathbf{b}_i \in \mathbb{R} \quad (3.8)$$

$$\boldsymbol{\alpha}^{norm} = softmax(\boldsymbol{\alpha}) \quad \boldsymbol{\alpha}^{norm} \in \mathbb{R}^{T_{enc}+1} \quad (3.9)$$

$$\boldsymbol{\beta} = \boldsymbol{\alpha}_{1,2,\dots,T_{enc}}^{norm} \quad \boldsymbol{\beta} \in \mathbb{R}^{T_{enc}} \quad (3.10)$$

$$g = \boldsymbol{\alpha}_{T_{enc}+1}^{norm} \quad g \in \mathbb{R} \quad (3.11)$$

3.5.3 Pointer model

As pointed out earlier, a pair of corresponding *Original* and *Modern* sentences have significant vocabulary overlap. Moreover, there are lot of proper nouns and rare words that might not be predicted by a sequence to sequence model. To rectify this, pointer networks have been used to enable copying of tokens from input directly [135]. The pointer module provides location based attention, and output probability distribution due to pointer network module can be expressed as follows:

$$P_t^{PTR}(w) = \sum_{\mathbf{x}_j=w} (\beta_j) \quad (3.12)$$

3.5.4 Decoder RNN

Summation of encoder states weighed by corresponding attention weights yields context vector. Output probabilities over vocabulary as per the decoder LSTM module are computed as follows:

$$\mathbf{c}_t = \sum_{i=1}^{T_{enc}} \beta_i \mathbf{h}_i^{enc} \quad (3.13)$$

$$\mathbf{h}_t^{dec} = LSTM(\mathbf{h}_{t-1}^{dec}, [\text{concat}(E_{dec}(\mathbf{y}_{t-1}), \mathbf{c}_t)]) \quad (3.14)$$

$$P_t^{LSTM} = \text{softmax}(W_{out}[\text{concat}(\mathbf{h}_t^{dec}, \mathbf{c}_t)] + \mathbf{b}^{out}) \quad (3.15)$$

During training, we feed the ground truth for \mathbf{y}_{t-1} , whereas while making predictions on test data, predicted output from previous step is used instead.

3.5.5 Output prediction

Output probability of a token w at step t is a weighted sum of probabilities from decoder LSTM model and pointer model given as follows:

$$P_t(w) = g \times P_t^{LSTM}(w) + (1 - g) \times P_t^{PTR}(w) \quad (3.16)$$

$P_t^{PTR}(w)$ takes a non-zero value only if w occurs in input sequence, otherwise it is 0. Forcing $g = 0$ would correspond to not having a *Copy* component, reducing the model to a plain attentional S2S model, that we refer to as a *SimpleS2S* model.

3.6 Loss functions

Cross entropy loss is used to train the model. For a data point $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}$ and predicted probability distributions $P_t(w)$ over the different words $w \in \mathbf{V}$ for each time step $t \in \{1, \dots, T_{dec}\}$, the loss is given by

$$-\sum_{t=1}^{T_{dec}} \log p(P_t(\mathbf{y}_t)) \quad (3.17)$$

Sentinel Loss (SL): Following from work by [135], we consider additional sentinel loss. This loss function can be considered as a form of *supervised attention*. Sentinel loss is given as follows:

$$-\sum_{t=1}^{T_{dec}} \log(g^{(t)} + \sum_{x_j=y_t} (\beta_j^{(t)})) \quad (3.18)$$

We report the results demonstrating the impact of including the sentinel loss function (+SL).

3.7 Experiments

In this section we describe the experimental setup and evaluation criteria used.

3.7.1 Preprocessing

We lowercase sentences and then use NLTK’s PUNKT tokenizer [95] to tokenize all sentences into their constituent words (tokens). The *Original* side has certain characters like æ that are not extant in today’s English language. We map these characters to the closest equivalent character(s) used today (e.g æ → ae)

3.7.2 Baseline Methods

As-it-is

Since both source and target side are English, just replicating the input on the target side is a valid and competitive baseline, with a BLEU of 21+.

Dictionary

Xu et al. [246] provide a dictionary mapping between large number of Shakespearean and modern English words. We augment this dictionary with pairs corresponding to the 2nd person thou (*thou*, *thy*, *thyself*) since these common tokens were not present.

Directly using this dictionary to perform word-by-word replacement is another admissible baseline. As was noted by Xu et al. [246], this baseline actually performs worse than *As-it-is*. This could be due to its performing aggressive replacement without regard for word context. Moreover, a dictionary cannot easily capture one-to-many mappings as well as long-range dependencies.⁶

Off-the-shelf SMT

To train statistical machine translation (*SMT*) baselines, we use publicly available open-source toolkit MOSES [99], along with the GIZA++ word aligner [147], as was done in [246]. For training the target-side LM component, we use the *lmpz* toolkit within MOSES to train a 4-gram LM. We also use *MERT* [147], available as part of MOSES, to tune on the validation set.

For fairness of comparison, it is necessary to use the pairwise dictionary and *PTB* while training the SMT models as well - the most obvious way for this is to use the dictionary and *PTB* as additional training data for the alignment component and the target-side LM respectively. We experiment with several SMT models, ablating for the use of both *PTB* and dictionary. In 3.8, we only report the performance of the best of these approaches.

3.7.3 Evaluation

Our primary evaluation metric is *BLEU* [151]. We compute *BLEU* using the freely available and very widely used perl script⁷ from the MOSES decoder.

We also report *PINC* [28], a metric which originates from paraphrase evaluation literature and evaluates how much the target side paraphrases resemble the source side. Given a source sentence s and a target side paraphrase c generated by the system, $PINC(s,c)$ is defined as

$$PINC(s,c) = 1 - \frac{1}{N} \sum_{n=1}^{n=N} \frac{|Ngram(c,n) \cap Ngram(s,n)|}{|Ngram(c,n)|}$$

⁶thou-thyself and you-yourself

⁷<http://tinyurl.com/yben45gm>

where $N\text{gram}(x, n)$ denotes the set of n-grams of length n in sentence x , and N is the maximum length of ngram considered. We set $N = 4$. Higher the PINC, greater the novelty of paraphrases generated by the system. Note, however, that PINC *does not measure fluency* of generated paraphrases. Moreover, it cannot be used to compare against references. Hence, it rewards all changes similarly irrespective of their fluency as well as adequacy, merely proportional to the extent of ngrams edited. As a result, it can merely be used as an auxiliary metric.

3.7.4 Training and Parameters

We use a minibatch-size of 32 and the ADAM optimizer [93] with learning rate 0.001, momentum parameters 0.9 and 0.999, and $\epsilon = 10^{-8}$. All our implementations are written in Python using Tensorflow 1.1.0 framework.

For every model, we experimented with two configurations of embedding and LSTM size - S (128-128), ME (192-192) and L (256-256). Across models, we find that the ME configuration performs better in terms of highest validation BLEU. We also find that larger configurations (384-384 & 512-512) fail to converge or perform very poorly.⁸ Here, we report results only for the ME configuration for all the models. For all our models, we picked the best saved model over 15 epochs that has the highest validation BLEU.

3.7.5 Decoding

At test-time we use greedy decoding to find the most likely target sentence.⁹ We also experiment with a post-processing strategy that replaces *UNKs* in the target output with the highest aligned (maximum attention) source word. We find that this gives a small jump in *BLEU* of about 0.1-0.2 for all neural models.¹⁰ Our best model, for instance, gets a jump of 0.14 to reach a BLEU of **31.26** from 31.12.

3.8 Results

The results in Table 3.3 confirm most of our hypotheses about the right architecture for this task.

⁸This is expected given the small parallel data

⁹Empirically, we observed that beam search does not give improvements for our task

¹⁰Since effect is small and uniform, we report BLEU before post-processing in Table 3.3

- **Copy component:** We can observe from Table 3.3 that the various *Copy* models each outperform their *SimpleS2S* counterparts by atleast 7-8 BLEU points.
- **Retrofitting dictionary constraints:** The *Retro* configurations generally outperform their corresponding *Plain* configurations. For instance, our best configuration *Copy.Yes.RetroExtFixed* gets a better BLEU than *Copy.Yes.PlainExtFixed* by a margin of atleast 11.
- **Sharing Embeddings:** Sharing source and target side embeddings benefits all the *Retro* configurations, although it slightly deteriorates performance (about 1 BLEU point) for some of the *Plain* configurations.
- **Fixing Embeddings:** *Fixed* configurations always perform better than corresponding *Var* ones (save some exceptions). For instance, *Copy.Yes.RetroExtFixed* get a BLEU of 31.12 compared to 20.95 for *Copy.Yes.RetroExtVar*. Due to fixing embeddings, the former has just half as many parameters as the latter (5.25M vs 9.40M)
- **Effect of External Data:** Pretraining with external data *Ext* works well along with retrofitting *Retro*. For instance, *Copy.Yes.RetroExtFixed* gets a BLEU improvement of 2+ points over *Copy.Yes.RetroFixed*
- **Effect of Pretraining:** For the *SimpleS2S* models, pre-training adversely affects BLEU. However, for the *Copy* models, pre-training leads to improvement in BLEU. The simplest pretrained *Copy* model, *Copy.No.PlainVar* has a BLEU score 1.8 higher than *Copy.No.NoneVar*.
- **PINC scores:** All the neural models have higher PINC scores than the statistical and dictionary approaches, which indicate that the target sentences produced differ more from the source sentences than those produced by these approaches.
- **Sentinel Loss:** Adding the sentinel loss does not have any significant effect, and ends up reducing BLEU by a point or two, as seen with the *Copy+SL* configurations.

3.8.1 Qualitative Analysis

Figure 3.2 shows the attention matrices from our best *Copy* model (*Copy.Yes.RetroExtFixed*) and our best *SimpleS2S* model (*SimpleS2S.Yes.Retrofixed*) respectively for the same input test sentence. Without an explicit *Copy* component, the *SimpleS2S* model cannot predict the words *saint* and *francis*, and drifts off after predicting incorrect word *flute*.

Table 3.1 presents model outputs for some test examples. In general, the *Copy* model outputs resemble the ground truth more closely compared to *SimpleS2S* and *Stat*. In some cases, it faces issues with repetition (Examples 4 and 6) and fluency (Example 8).

Model	Sh	Init	BLEU	PINC
AS-IT-IS	-	-	21.13	0.0
DICTIONARY	-	-	17.00	26.64
STAT	-	-	24.39	32.30
SIMPLES2S	✗	<i>NoneVar</i>	11.66	85.61
	✗	<i>PlainVar</i>	9.27	86.52
	✗	<i>PlainExtVar</i>	8.73	87.17
	✗	<i>RetroVar</i>	10.57	85.06
	✗	<i>RetroExtVar</i>	10.26	83.83
	✓	<i>NoneVar</i>	11.17	84.91
	✓	<i>PlainVar</i>	8.78	85.57
	✓	<i>PlainFixed</i>	8.73	89.19
	✓	<i>PlainExtVar</i>	8.59	86.04
	✓	<i>PlainExtFixed</i>	8.59	89.16
	✓	<i>RetroVar</i>	10.86	85.58
	✓	<i>RetroFixed</i>	11.36	85.07
	✓	<i>RetroExtVar</i>	11.25	83.56
	✓	<i>RetroExtFixed</i>	10.86	88.80
COPY	✗	<i>NoneVar</i>	18.44	83.68
	✗	<i>PlainVar</i>	20.26	81.54
	✗	<i>PlainExtVar</i>	20.20	83.38
	✗	<i>RetroVar</i>	21.25	81.18
	✗	<i>RetroExtVar</i>	21.57	82.89
	✓	<i>NoneVar</i>	22.70	81.51
	✓	<i>PlainVar</i>	19.27	83.87
	✓	<i>PlainFixed</i>	21.20	81.61
	✓	<i>PlainExtVar</i>	20.76	83.17
	✓	<i>PlainExtFixed</i>	19.32	82.38
	✓	<i>RetroVar</i>	22.71	81.12
	✓	<i>RetroFixed</i>	28.86	80.53
	✓	<i>RetroExtVar</i>	20.95	81.94
	✓	<i>RetroExtFixed</i>	31.12	79.63
COPY+SL	✗	<i>NoneVar</i>	17.88	83.70
	✗	<i>PlainVar</i>	20.22	81.52
	✗	<i>PlainExtVar</i>	20.14	83.46
	✗	<i>RetroVar</i>	21.30	81.22
	✗	<i>RetroExtVar</i>	21.52	82.86
	✓	<i>NoneVar</i>	22.72	81.41
	✓	<i>PlainVar</i>	21.46	81.39
	✓	<i>PlainFixed</i>	23.76	81.68
	✓	<i>PlainExtVar</i>	20.68	83.18
	✓	<i>PlainExtFixed</i>	22.23	81.71
	✓	<i>RetroVar</i>	22.62	81.15
	✓	<i>RetroFixed</i>	27.66	81.35
	✓	<i>RetroExtVar</i>	24.11	79.92
	✓	<i>RetroExtFixed</i>	27.81	84.67

Table 3.3: Test BLEU results. *Sh* denotes encoder-decoder embedding sharing (*No*=✗, *Yes*=✓) . *Init* denotes the manner of initializing embedding vectors. The -*Fixed* or -*Var* suffix indicates whether embeddings are fixed or trainable. COPY and SIMPLES2S denote presence/absence of *Copy* component. +SL denotes sentinel loss.

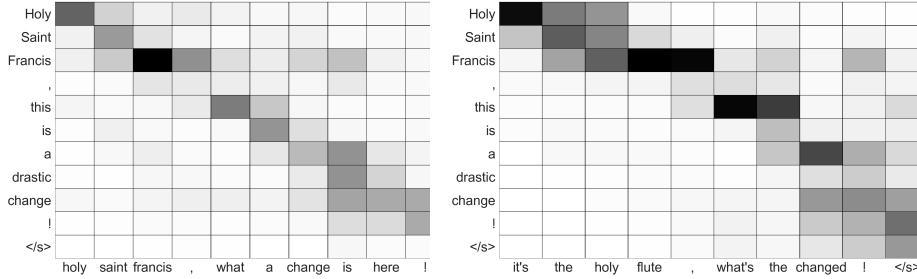


Figure 3.2: Attention matrices from a *Copy* (top) and a *simple S2S* (bottom) model respectively on the input sentence “*Holy Saint Francis, this is a drastic change!*”. $< s >$ and $< /s >$ are start and stop characters. Darker cells are higher-valued.

3.9 Related Work

There have been some prior work on style adaptation. Xu et al. [246] use phrase table based statistical machine translation to transform text to target style. On the other hand our method is an end-to-end trainable neural network. Saha Roy et al [196] leverage different language models based on geolocation and occupation to align a text to specific style. However, their work is limited to addition of adjectives and adverbs. Our method can handle more generic transformations including addition and deletion of words.

Pointer networks [231] allow the use of input-side words directly as output in a neural S2S model, and have been used for tasks like extractive summarization [206] [251] and question answering [234]. However, pointer networks cannot generate words not present in the input. A mixture model of recurrent neural network and pointer network has been shown to achieve good performance on language modeling task [135].

S2S neural models, first devised by [224], and enhanced with a attention mechanism by [9], have yielded state-of-the-art results for machine translation (MT), , summarization [194], etc. In the context of MT, various settings such as multi-source MT [258] and MT with external information [209] have been explored. Distinct from all of these, our chapter attempts to solve a Modern English → Shakespearean English style transformation task. Although closely related to both paraphrasing and MT, our task has some differentiating characteristics such as considerable source-target overlap in vocabulary and grammar (unlike MT), and different source and target language (unlike paraphrasing). [60] have devised a neural sequence-to-sequence solution for generating a portmanteau given two English root-words. Though their task also

involves large overlap in target and input, they do not employ any special copying mechanism. Unlike text simplification and summarization, our task does not involve shortening content length.

3.10 Conclusion

We have presented here a study of diachronic style transfer to Shakespearean style English. This is an example where the communicative goal was to alter the particular interpersonal aspect/facet of the *author’s diachronic register* and change its value. Naturally, a great many number of variations of the style transfer task exist based on the specific interpersonal and ideational aspects one wishes to alter, and their desired target configuration. Nevertheless, the observations we make here do generalize to other *style transfer* tasks as well as *control* tasks in general, as we shall discuss in §3.10.1

Our contribution was primarily along the *contribution type* [Method] and [Knowledge]. Specifically, in this chapter, our recommended approach leads to three major changes in the typical E2EN2PP pipeline. First, we recommend using a mixture model of pointer network and LSTM to transform Modern English text to Shakespearean style English. This exploits the property of a *shared language* between source and target sides, a property likely to be shared by a large number of style transfer tasks. Second, we recommend having a shared representation (embedding) space for source and target words initially, and pretrain this using a combined corpus of sentences from either of the sides. Third, we devise a mechanism based on retrofitting [48] to incorporate pairwise lexical source word → target word constraints from a dictionary, into this initial shared representation. This third step represents incorporation of *knowledge* from a resource external to the communicative goal.

We demonstrate the effectiveness of our devised approach over the baselines. Our experiments reveal the utility of incorporating input-copying mechanism, and using dictionary constraints for problems with shared (but non-identical) source-target sides and sparse parallel data. We release our code publicly to foster further research on stylistic transformations on text.¹¹ The work has since then been positively received by the community with over 130 citations and 46 publications explicitly using this as a baseline.

Sparknotes also provides similar translations to Modern English for the Anglo-Saxon Age epic *Beowulf*¹² (Old English) and the famous monastic devotional chronicle *Canterbury Tales*

¹¹<https://github.com/harsh19/Shakespearizing-Modern-English>

¹²tinyurl.com/d5ntme7

(Middle English). A possible future direction is to test the model for styles other than Shakespearean English would be develop models to translate to these styles/languages. An additional extension would be to develop a single unified encoder-decoder model for translation to any diachronic English style, in the manner of [83], who developed an any-source language to any-target language unified translation model.

3.10.1 Broader Takeaways

The property of a *shared language* and the resultant two enhancements we devise are likely applicable to almost any style transfer task. Specifically, these two enhancements are:

1. Using a pointer component in the generating distribution to ease the process of learning to copy over the style-invariant, textual aspect correlated portions of the input. Even the *most disparate* source and target styles are likely to contain *some textual overlap* by sheer virtue of them being *meaning invariant* and sharing the same *script* and *language*. This textual overlap would stem from *named entities* (e.g the KGB, Banquo, Macbeth, Benjamin Disraeli, Donald Trump etc), *strongly topical words* (treaty, detente, fiefdom, shield, bill, election etc) and other such word classes strongly related to the core meaning of the utterance being transformed.
2. Using a jointly pretrained initial shared representation for both source (input) and target (output) sides. Sharing representations is a simple modelling change, while joint pretraining is always going to be feasible given that we don't necessarily need paired source-target data for this, all that is needed is unpaired text from both directions, that is likely to be available in atleast some reasonable quantity given that the task is being posed as a reasonably learnable one.

The above two correspond to purely methodological improvements that can be made, and are not dependent on any further knowledge sources external to the communicative goal.

The third enhancement we devised was, as the reader would recollect, based on incorporating pairwise lexical source word → target word constraints from a dictionary, into the initial, post-pretraining shared word representation via retrofitting [48]. This third step requires incorporation of knowledge from a resource external to the communicative goal, namely a dictionary-like resource that provides some (though not necessarily exhaustive) strong pairwise correspondences between some source and target word types. This enhancement is not as widely generalizable as the first two, since

1. Not all (source style, target style) pairs i.e style transfer settings would necessarily exhibit

strong pairwise lexical correspondences. In this case, even if one were try to construct a collection of pairwise constraints, it would either be completely empty or of negligible size.

2. A resource of the nature required may not be available, even though the style transfer setting does exhibit strong pairwise lexical correspondences. How to *approximately extract such constraints* in an automatic fashion is an interesting direction to explore for future work. Possible intuitions to explore include

- Using analogies to find facet correspondences (e.g gender, wealth etc) rather than (word,word) correspondences. E.g the gender analogy vector can be found by averaging over seed pairs such as (king,queen), (man,woman) etc in both source and target spaces (This procedure will have to be careful in choosing only those seed pairs that are invariant to the style change, if at all they exist). Now, one can try to align the gender analogy vectors of the source and target spaces to point in the same direction, using the same retrofitting procedure.
- Using statistical properties such as PMI or mutual information to automatically mine out such (source,target) word form correspondences from the paired training data.

Nevertheless, some strong lexical correspondences are indeed found in many popular and practically style transfer settings e.g *formality* and *politeness*, and this enhancement would be useful in such cases. For instance, a Prof. X Y in a polite setting would always get replaced by a X in the impolite setting. Likewise, the word forms *thanks* and *its* (in its non-possessive form) in an informal sentence would be typically converted to *thank you* and *it is* in a formal setting.

To conclude, we have discussed here how two of our three enhacements generalize to any style transfer setting, while the third one generalizes to a specific, though significant subset. Note that we defined style transfer very broadly — as a collection of non-textual subgoals that are achieved by means of modifying a given source text/input to match a certain target style where those subgoals are optimal. Hence, our findings relevance encompasses many popular and widely applicable NLP tasks, such as text normalization [8], domain adaptation, text simplification [26], expertise style transfer [24] amongst others.

Chapter 4

Tongue Twister Generation (Proposed) [New Task]

4.1 Introduction

Tongue twisters are sentences which are *fluent* besides being *difficult to pronounce* e.g., "*She sells seashells on the seashore.*", "*What's a synonym for cinnamon.*" etc.

Through our chapter on portmanteau generation (Chap. 2), we already made a foray into creative NLG tasks whose communicative goal significantly focusses on the lexico-phonetic aspect of surface realization . However, since the task setting of portmanteau generation required constructing the portmanteau given two root word types, rather than tokens in specific sentential contexts, it did not require consideration of higher-granularity aspects such as fluency and semantic meaningfulness. Generating a tongue twister, however, requires

1. Maintaining difficulty of pronunciation, which is a token and phrase-level subgoal related to the lexico-phonetic subskill of the wider skill of surface realization. Note that even the lexico-phonetic subgoal alone is more challenging to fulfill than the one we had to in portmanteau generation during Chapter 2. In that chapter, the phonetic subgoal was restricted to a single word type i.e., the generated portmanteau had to sound word-like and reminiscent of its two root word types. However, maintaining a high difficulty of pronouncing the sentence at most points throughout, requires generating a sequence of word forms that are each hard to pronounce in-context. This requires managing the

lexico-phonetic aspect not just at the word level but also at the level of phrases/local word contexts.

2. Maintaining fluency, that is a subgoal at both the phrase and sentence levels. A sentence is fluent if it sounds natural to a native speaker of that language. Fluent sentences must be both grammatical as well as meaningful and can't merely satisfy just one of the two conditions i.e, "*Colorless green ideas sleep furiously.*" and "*Milk I fetch but grow hot milk cold.*" are both not fluent.

Hence, the importance of tongue twister generation from the purview of our thesis and its objectives is that it is a creative NLG task with the rare property that it requires jointly satisfying a blend of higher-granularity, phrase and utterance-level communicative subgoal (i.e., semantic meaningfulness and fluency) and a lower-granularity, lexico-phonetic communicative subgoal (i.e., difficulty of pronunciation).

4.2 Background

Historically, along with riddles, rhymes, fables and other such creative artifacts, tongue twisters have often been employed as a vehicle for early transmission of native language diction, grammar and vocabulary to children, through e.g., parent-child interactions, playtime activities and kindergarten instruction [2, 129]. Tongue twisters have also been employed as experimental aides for research studies of speech production in cognitive science and related disciplines, both amongst healthy speakers and those with speech and auditory disorders such as dysarthria [87]. They are also used as pedagogic aides in speech therapy and treatment of speech disorders as well as psychological disorders relating to public speaking and elocution [190]. Lastly, they find use as pedagogic aides both in teaching English pronunciation and diction in EFL (English as a Foreign Language) instructional settings [167].

Tongue twisters are not merely a phenomenon limited to English, and are found across most major languages and cultures e.g., French ("Cinq chiens chassent six chats."), Hindi ("Chandu ke chacha ne Chandu ki chachi ko chandi ke chamche se chutney chatai.") inter alia.

Despite their global footprint, the set of well-known and accepted tongue twisters in each language tend to be rather small in number, numbering in the few thousands, and novel tongue twisters are either seldom coined or often fail to achieve sufficiently widespread circulation amongst the corresponding population of language speakers. Based on our examination of English resources online, there do not exist more than 2–3K tongue twisters ever coined and

widely circulated amongst English speakers. Though one can certainly construct a good number of these with some amount of individual skill and effort, then getting these to be widely accepted is likely to be an uphill and non-trivial task.

4.2.1 Distinction from poetry

Tongue twisters are distinct from poetry, in that the phonetic subgoal tongue twisters must satisfy is of a markedly different nature than the ones poems have to, in the ways laid out below

1. **Amorphously Defined Subgoal:** Poetry (or atleast its traditional, non-freestyle form) is defined through a set of explicit, well-defined subgoals on rhyme and rhythm e.g., rhyming schemes such as ABAB specify how each sequence of 4 lines should align in terms of their last words. This also makes it easy to evaluate how much a generated poem adheres to the constraints. In contrast, "being difficult to pronounce" is an implicit subgoal that is much harder to evaluate.
2. **Negative Subgoal:** Subgoals such as being fluent, or satisfying a 4-line rhyming scheme can be characterized as "positive" in the sense that they require scoring highly as per some estimator, be it a language model for fluency, or the fraction of couplets for which the scheme is satisfied for rhyming scheme. On the other hand, "difficult to pronounce" is a negative notion that can be characterized in terms of *scoring lower* as per some model-based estimator which scores "typical" or expected pronunciation.

4.3 Representational Primitives

Devising a model architecture which fulfills both the subgoals (difficulty of pronunciation and fluency) with disparate granularities requires two submodels - each operating in the phonetic and word/subword level representation space. For the submodel of difficulty of pronunciation, we employ a representation space P^* , where P is a set of phonetic symbols and $*$ represents the Kleene closure. Likewise, for the submodel concerned with fluency, we employ a representation space W^* , where W is the set of words/subwords in the English language as estimated from a sufficiently large corpus.

Specifically, we choose the IPA (International Phonetic Alphabet) [7] for this purpose.

4.4 Dataset

We curate a [dataset¹](#) of ≈ 650 tongue twisters from a wide range of [sources](#), both historical and recent. Through manual examination, we exclude the small number of tongue twisters based on meaningless repetition of 2-3 words e.g., “*red blood, blue blood*”, retaining only the proper, full-sentence tongue twisters.

4.5 Conclusion

¹https://github.com/vgtomahawk/TT_NLP/blob/master/Data/dataset.txt

Part II

Microplanning

Chapter 5

**Improving Realization Level Metric
Evaluation of Open Dialog via
Microplanning Rollouts for Reference
Augmentation
(Findings of ACL 2021)**

[Evaluation][Knowledge]

There has been increasing recognition that context is of overwhelming importance in the interpretation of text. Implicit real world knowledge is often applied by the understander, and this knowledge can be very highly structured. The appropriate ingredients for extracting the meaning of a sentence, therefore, are nowhere to be found in the sentence itself.

R.Schank and P.Abelson, *Scripts, Plans, Goals and Understanding: An Inquiry Into Human Knowledge* (1972)

Multiple different responses are often plausible for a given open domain dialog context. Prior work has shown the importance of having multiple valid reference responses for meaningful and robust automated dialog evaluation. In such cases, the common practice has been to collect more human written references. However, such collection can be expensive, time consuming, and not easily scalable. In this chapter, we show an entirely alternative route to improve automated evaluation and make it conform closer to human evaluation, sans any additional annotation. We devise SCARCE, a novel suite of techniques for *automatically* expanding a small set of human reference and its dialog context to a larger set of pseudo-references.

A dialog context, which consists of a few finite turns of conversation before the model response or either of the references, maybe thought of a *partial microplan*, or a *microplan-in-action* in the process of generating an entire dialog.

Automatic evaluation metrics, such as BLEU [151], METEOR [12] and BERTScore [253] compare the response to each of the references merely at the surface level, using some measure of symbolic n-gram overlap, or pairwise vector similarities. Hence, they are agnostic to the available dialog context information and cannot leverage it while evaluating. However, if we had access to a knowledge source that can project/predict out another turn of multiple, explicit, plausible responses from the dialog context, we can use these explicit responses as “pseudo-

references". Note that these responses need not be fully fleshed out, and could even simply be terse inferences about the situation or the world state. We merely need a small amount of heuristic, adaptive post-processing (see §5.2.2, for instance) to convert them to conversation-like responses. For example, if the context consists of John saying "*I went to meet my mother*", a plausible inference returned by a commonsense knowledge base such as COMET [18] could be "*Mother feels happy*". This can be easily adapted using a rule-based heuristic to more conversational response like surface forms e.g., "*Did your mother feel happy?*" , "*Was your mother feeling happy?*" etc. The process of fetching these pseudo-references can be thought of as multiple rollouts or forward samples of the partial microplan, represented here by the dialog context, with the "rolling out" / "sampling" being done implicitly by our knowledge source.

More specifically, we fetch plausible "pseudo-references" from knowledge sources, and adapt them so that they are more fluent in context of the dialog instance in question. In terms of knowledge sources, we use (1) a commonsense knowledge base to elicit a large number of plausible reactions given the dialog history (2) relevant instances retrieved from dialog corpus, using similar past as well as future contexts. We demonstrate that our automatically expanded reference sets lead to large improvements in correlations of automated metrics with human ratings of system outputs for dialog contexts from the DailyDialog dataset [107].

5.1 Introduction

Evaluation by human annotators perhaps give the best insights into quality of machine generated natural language outputs. However, such evaluation can be expensive and time consuming. This makes it impractical to use this during model development, where one may be experimenting with and comparing 100s of hyperparameter configurations. This is because using evaluation by human annotators at this stage would entail hundred-fold the cost of just doing this evaluation at test time, both in terms of time and money. Even before getting to the hyperparameter tuning stage, one needs some form of evaluation to sanity check implementations of new model variants as well as replicated native implementations of existing methods. Depending on the complexity of the model architectures and the developer's skills, this can take tens to hundreds of iterations depending on the bugs encountered along the way. Again, we cannot rely on human annotator evaluation for this purpose on account of the prohibitive cost.

Much focus has therefore been on automated evaluation methods that correlate with human evaluations. Automated metrics such as BLEU [151] work well for tasks such as machine trans-

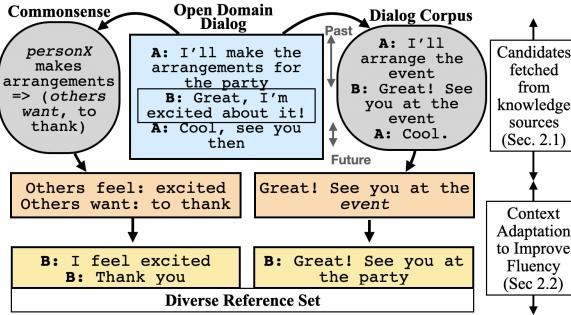


Figure 5.1: We devise automatic ways to collect references sans any crowd-sourcing, through two types of knowledge sources: commonsense and retrieved instance knowledge, followed by automated adaptation to make them more fluent in the target contexts.

lation, but often correlate poorly with human ratings in tasks such as open domain dialog that admit a wide variety of valid response for given context, often due to small number of human written references [199, 256]. Prior work [69, 223] has demonstrated that having multiple valid references for the same context leads to automated metrics being better correlated to human judgements for appropriateness. However, collecting human written responses is difficult to scale, can be costly, and it can be difficult to cover a large variety of correct responses [25].

In this chapter, we automatically extract a large number of diverse references to be used with such reference-based metrics, without resorting to expensive crowd-sourcing. Intuitively, since open-domain dialog pertains to everyday life, its utterance text tends to re-instantiate from a large but limited pool of situations [203] e.g., friends debating politics etc, with variation only on some details e.g. country discussed. Hence, knowledge encapsulating a wide scope of situations can serve as one starting point to automatically seed a set of diverse references. We first fetch plausible candidates from two types of knowledge sources (Figure 5.1). Such knowledge sources provide ready and easy access to a large number of potentially appropriate and diverse references. However, all retrieved instances may not be directly useful. As such, to achieve more fluent references, we devise techniques to adapt the candidate references based on the context (e.g. change country being discussed). Note that since we are interested in creating references for only evaluating appropriateness of system outputs, our techniques can rely on broader data sources compared to dialog models. For example, we use future context and human written reference for retrieval, while a dialog model cannot.

Our contributions are as follows: (1) We devise a method for automated reference set augmentation for automated dialog evaluation. Compared to collecting more human-written responses, our approach is inexpensive and scalable, and fetches a diverse set of references. (2) We observe high correlations of various automated metrics with human ratings when devised

reference augmentation is applied to the test split of DailyDialog dataset [107]. We additionally observe that paraphrasing, a popular data augmentation technique, performs much worse. (3) We employ novel use of commonsense knowledge and dialog corpus instances, and unsupervised techniques for adapting retrieved references into more fluent forms.

5.2 Method

Figure 5.1 shows an overview of our devised methodology. We first fetch plausible candidates from two types of knowledge sources. Thereafter, the retrieved candidate references are adapted so that they are fluent in the target context. We refer to our devised method as **SCARCE** (**S**Calable **A**utomated **R**eference **C**onstruction for **E**valuation).

5.2.1 Knowledge Sources

Pre-trained Commonsense Model Much open domain dialog is based on everyday matters. We posit that extracting inferences about a situation using a commonsense knowledge base could be useful in identifying a wide variety of plausible reactions for a given dialog context. For example, a person making arrangements for an event might receive thanks from others (Figure 5.1). We utilize COMET [18] an off-the-shelf commonsense knowledge model built on ATOMIC [200] or ConceptNet [220] corpus. It can be used to elicit commonsense inferences.

Traditional commonsense knowledge bases are labelled, directed graphs consisting of nodes which are short phrases describing the associated entities or concepts, e.g. President of the U.S.A, gamekeeper, shock infantry, blitzkrieg etc. Consider a pair of nodes u and v . A labelled, directed edge $\{u, v, r\}$ from u to v indicates that v is a possible attribute value for node u given a relation r . For example, $\{\text{gamekeeper}, \text{IsA}, \text{profession}\}$ and $\{\text{gamekeeper}, \text{InteractWith}, \text{animals}\}$ would be labelled, directed edges, indicating that *gamekeeper* is a type of profession and that a *gamekeeper* has to typically interact with animals. A problem faced by traditional commonsense knowledge bases is the low coverage in terms of being able to handle variation in phrasing of query concepts and relations e.g. on being queried with the concept-relation tuple $\{q_u = \text{zookeeper}, q_r = \text{HasToDealWith}\}$, if an explicit concept node $u = \text{zookeeper}$ does not exist in the KB graph, it has to be matched to $u = \text{gamekeeper}$ using some heuristic in order to return a non-empty result. Assuming this matching is done correctly, a further round of matching has to be done in case the relation $r = \text{HasToDealWith}$ does not directly exist, having to match it with $r = \text{InteractWith}$. In the event that both q_u and q_r both don't exist, getting

a non-empty result is hence contingent on the matching heuristic getting these two matching steps right. One can see how designing such matching heuristics could be tricky, and even if reasonably well-functioning, they would always have some failure cases.

Specifically, COMET is a left-to-right, large, pretrained language model, specifically GPT-2 [170], which is further finetuned on a constructed training split where every edge of a static, traditional commonsense knowledge graph is converted to a training example. This allows the model to not only memorize the original knowledge base, but also generalize to newly phrased concepts and relations at test time. COMET is queried in mostly the same way as one would the original KB it was trained on, the only difference being that instead of a graph lookup, we feed in the query to the model in the form of a tag-structured prompt $x.q_u < SEP > o.q_r$

COMET-ATOMIC provides inferences on cause-effect interrelations between events pertaining to nine relation types such as oReact (effect on others due to the event), oWant (inferences about wants of the receiver of event), etc. The prefix ‘x’ indicates an effect or cause on the actor, and ‘o’ denotes the same on others. Given an utterance from the previous speaker, we draw up to 5 inferences pertaining to each of oEffect, oReact, and oWant relation types to construct plausible references for the target response. For example, for an utterance ‘I will make the arrangements. It will be great.’, one of the inferences corresponding to oEffect is ‘feel excited’, depicting a plausible state of the next dialog speaker. However, such outputs are typically phrases, and we discuss transformation to fluent sentences in Section 5.2.2. Similarly, we use inferences pertaining to ‘CausesDesire’ and ‘HasFirstSubevent’ relation types from COMET-ConceptNet.

Dialog Corpus Retrieval For a test dialog context under consideration, one is likely to find similar contexts occurring in some of the training dialogs, given a sufficiently well-sized corpus training split. Using retrieval, we can identify such contexts and use their responses as pseudo-references for the test-time response. Our approach is related to Galley et al. [58], who propose the Δ -BLEU measure which uses retrieval to produce pseudo-references. However, unlike our method, they require annotator qualityx scores to weigh them during evaluation. Moreover, though we utilize retrieval here for evaluation, methods of this kind have found success in many generation setups as well. [90, 106, 155].

Spearman Rank Correlation / Kendall Tau Rank Correlation

Setup	1 human written reference			4 human written references		
	Dataset	SINGLE [107]	PARAPHRASE -SINGLE	SCARCE -SINGLE(Ours)	MULTI [69]	PARAPHRASE -MULTI
BLEU4	0.09 / 0.07	0.13 / 0.09	0.30 / 0.21	0.28 / 0.20	0.27 / 0.19	0.36 / 0.25
BLEU3	0.06 / 0.04	0.11 / 0.07	0.29 / 0.20	0.24 / 0.17	0.24 / 0.17	0.35 / 0.24
BLEU2	0.04 / 0.03	0.08 / 0.06	0.28 / 0.19	0.20 / 0.14	0.21 / 0.14	0.33 / 0.23
BLEU1	0.02 / 0.02	0.06 / 0.04	0.25 / 0.17	0.19 / 0.13	0.18 / 0.12	0.29 / 0.21
ROUGE-L	0.07 / 0.05	0.09 / 0.06	0.26 / 0.18	0.20 / 0.14	0.20 / 0.14	0.32 / 0.22
METEOR	0.11 / 0.07	0.09 / 0.06	0.24 / 0.17	0.23 / 0.16	0.22 / 0.15	0.30 / 0.21
EmbeddingAvg	0.03 / 0.02	0.02 / 0.01	0.02 / 0.02	0.10 / 0.07	0.10 / 0.07	0.08 / 0.05
SkipThought	-0.00 / 0.00	-0.03 / -0.02	0.09 / 0.07	0.07 / 0.05	0.05 / 0.04	0.13 / 0.10
BERT-Prec	0.27 / 0.19	0.28 / 0.19	0.38 / 0.26	0.32 / 0.22	0.32 / 0.22	0.41 / 0.28
BERT-Rec	0.10 / 0.06	0.09 / 0.06	0.24 / 0.16	0.23 / 0.16	0.21 / 0.15	0.30 / 0.21
Max. value	0.27 / 0.19	0.28 / 0.19	0.38 / 0.26	0.32 / 0.22	0.32 / 0.22	0.41 / 0.28

Table 5.1: Utterance level Spearman Rank Correlation [219] and Kendall Tau Rank Correlations [88]. (1) SCARCE-SINGLE augments the original single human written response (SINGLE) in DailyDialog dataset [107] using the devised method. It leads to large improvements in correlations across most of the metrics, when compared to SINGLE. (2) SCARCE-MULTI augments the MULTI dataset, again leading to improvements in correlations to human ratings, especially for BLEU and BERT-Prec metrics.

5.2.2 Context Adaptation

We note that commonsense knowledge outputs are often incomplete sentences, and we use simple templates to convert them to fluent sentences e.g. ‘feels excited’ gets transformed to ‘i feel excited’. (Detailed templates in §5.8.1).

Further, we note that references from knowledge sources are sometimes not completely adequate for the target context. For example, ‘event’ in the retrieved reference shown in Figure 5.1 can be updated to ‘party’ to construct a more apt reference. To adapt the retrieved text to better fit the target context we use employ an unsupervised decoding procedure, based on the approach of Qin et al. [168], that uses gradient ascent to search for output text that maximizes (1) fluency with the left context (approximated by the likelihood of the output text under a pretrained GPT-2 model) and (2) similarity to the original text from the knowledge source (approximated by the likelihood of the original text under the output text’s token-level word distributions). The method utilizes a heuristic update procedure to iteratively refine a differentiable proxy for the output text (a sequence token-level word distributions), while keeping the model parameters fixed. More details can be found in Qin et al. [168] and in §5.8.2.

5.3 Experiments

We investigate the extent to which automated metrics on an evaluation dataset correlate with human ratings of system outputs. We use the human ratings collected by Gupta et al. [69], who collected utterance level human ratings using Amazon Mechanical Turk (AMT). They used a collection of 100 dialogue contexts that are randomly selected from the DailyDialog dataset. The generated responses from various methods are rated in terms of appropriateness (from 1-5) by 5 different AMT workers. They collected and considered outputs from following methods: CVAE [255], HRED [210], Seq2Seq [230], Dual-encoder [119], and Human-written responses. We report Spearman rank correlation and Kendall tau rank correlation of human ratings against ngram overlap metrics such as BLEU [151], METEOR [12], ROUGE-L [111], and embedding based metrics like cosine similarity of average word embedding (EmbAvgSim) [240] or Skip Thought Embedding [94], and BertScore [254] (BERT-Prec,BERT-Rec).

We compare the correlations across following setups: **SINGLE** [107]: Original DailyDialog dataset that had one reference per context; **SCARCE-SINGLE**: Proposed method along with SINGLE reference; **MULTI** [69]: Upto 5 human written references. **SCARCE-MULTI**: Reference responses from the devised method along with MULTI references. Additionally, we report the results when using PARAPHRASE instead of SCARCE: **PARAPHRASE-SINGLE** and **PARAPHRASE-MULTI**. Paraphrasing is a popular approach for automated data augmentation. Paraphrasing via backtranslation (BT) [208] is known to be an effective, domain-independent way to generate good quality paraphrases [239]. We use the BT model from [244] with its default hyperparameters to sample multiple paraphrases per human written reference.

Results: We observe that most of the metrics show large improvements in correlations to human ratings for appropriateness when used along with SINGLE or MULTI (Table 5.1). In fact, rank correlations across most of the metrics are better for SCARCE-SINGLE compared to MULTI, even though former uses only single human written reference while latter uses upto 5 human written references¹. Additionally, we observe that PARAPHRASE produces little or no improvements in correlations with human ratings (Table 5.1). We posit that for a given response, alternate responses constitute a strictly richer subspace than that of response paraphrases, that tend to be lexico-syntactically variant but semantically invariant.

¹Rank correlations for SINGLE and MULTI deviate from the values in Gupta et al. [69], who (in private communication with us), confirmed that the final dataset and code available on their repo does lead to the numbers we report.

Method	BLEU4	BERT-Prec
SCARCE-SINGLE	0.30	0.38
SCARCE-SINGLE variants:		
COMMONSENSE only	0.24	0.31
RETRIEVAL only	0.29	0.36
RETRIEVAL only (5% corpus)	0.17	0.28
w/o CONTEXT-ADAPT	0.26	0.37

Table 5.2: Analyzing impact of various components

Analyzing impact of various components: To understand the impact of various components, we report BLEU-4 and BERT-Prec scores with some variants of SCARCE-SINGLE (Table 5.2). We note that even considering only one knowledge source (COMMONSENSE-only, RETRIEVAL-only) leads to good Spearman rank correlations of automated metrics to human ratings. Thus, the additive effect (SCARCE-SINGLE) shows rather small incremental benefit. Moreover, RETRIEVAL by itself does better than COMMONSENSE, though at smaller corpus availability (e.g. 5%), COMMONSENSE performs better. Finally, not using context adaptation (w/o CONTEXT-ADAPT) leads to significant performance drop.

Quality of Auto-generated References: We check the quality of SCARCE references by recruiting human annotators, showing them the reference along with the dialog context, and requesting them to tag each reference as appropriate, neutral, or not-appropriate. We randomly select 110 responses each from SCARCE and MULTI. We observe that in aggregate, 15% of the references from SCARCE (fully automatically generated) were annotated as not appropriate, compared to 7% for MULTI (Details in §5.11.1).

Evaluation with a Second Dataset: The human ratings dataset from Gupta et al. [69] does not include outputs from some recent models such as Transfertransfo [243], which is a GPT-2 model fine-tuned on DailyDialog. We collect additional pairwise human preference ratings for 135 test instances with where one of the responses is from the Transfertransfo model, and the other one is from Seq2Seq or HRED (Seq2Seq and HRED had best overall human ratings). Each response was tagged by 3 annotators using Amazon Mechanical Turk, and we pick the majority rating. Kendall Tau rank correlation for Bleu4 for SINGLE, SCARCE-SINGLE and MULTI are 0.03, 0.31 and 0.17, while for BERT-Prec are 0.11, 0.11, and -0.03 respectively. Thus, SCARCE-SINGLE again performs similar or better than SINGLE and MULTI.

5.4 Discussion

Transferability to more languages: Transferability of our approach to more languages is one aspect that merits discussion. While commonsense resources aren’t readily available in all languages, a workaround can be to use off-the-shelf MT to translate before querying into English versions of the commonsense resources, and then translate back retrieved information. Furthermore, we note that while commonsense knowledge was useful, removing the COMMONSENSE method and relying on retrieval alone causes only relatively modest drop in performance (see Table 5.2). Thus, for languages lacking commonsense resources, one may still attain good gains in reference based evaluation by retrieving and adapting from dialog corpus alone.

Reference-less metrics: We note that while comparisons of using the devised approach against using reference-free metrics [120, 226] would be interesting, the focus of the current chapter is on improving reference-based evaluation via unsupervised reference augmentation. While reference-less metrics offer convenience to work with zero or a very small number of references, reference-based metrics can be advantageous on several fronts. Reference-based evaluation can be more interpretable under certain situations by identifying the reference that matches the most with a given system output. Reference-based evaluations allow for easy incorporation of additional references — in contrast, many learned model-based metrics will require retraining if additional annotations become available.

5.5 Related Work

Prior work explores many ways to improve over single-reference evaluation without collecting multiple ones. Fomicheva et al. [55] obviate need for multiple references in MT by generating many “alt-hypotheses” via test-time dropout from the same model. Sai et al. [198] and Gupta et al. [69] collect additional manually annotated responses for dialog contexts. Compare to them, our method of automatically collecting additional references automatically is more scalable.

Automatic data augmentation in NLP has largely been used for increasing training data [51, 52, 235]. In this chapter, we use retrieved dialog instances and commonsense knowledge base to augment reference set for a given dialog context. Δ -Bleu [58] and uBLEU [249] also use retrieval to produce pseudo-references for dialog response evaluation. Compared to Δ -Bleu and uBLEU, our work is different since we utilize commonsense knowledge base and perform contextual

adaptation. Prior work in dialog response generation has explored the use of commonsense knowledge base [125] as well as retrieval [126, 218] – in contrast, our focus is on augmenting reference set for improving evaluation.

Automatic model-based metrics like ADEM [120] and RUBER [226], that incorporate context while scoring for evaluation, at first glance seem to reduce the need for multiple references. However, these metrics have been found to suffer from several peculiar problems. For instance, ADEM can't discriminate between gold responses and certain classes of adversarial negatives e.g. reversed gold responses and repeating the context as the response [197]. Sato et al. [202] evaluate dialog systems through their ability at selecting valid responses from a semi-automatically curated candidate list. Mehri and Eskenazi [133] introduce the unsupervised, reference-free USR metric, that leverages a suite of RoBERTa [118] models, each fine-tuned to score one of five dialog aspects e.g. *Natural* and *Uses Knowledge*. Mehri and Eskenazi [132] further expand their USR metric to eighteen aspects from the initial five.

5.6 Conclusion

In this chapter, through our approach named SCARCE, we demonstrate how knowledge sources can be used in an automated and scalable manner to construct a diverse set of pseudo-references, starting with an initial set of human references and their dialog context. These pseudo-references can then be used along with the original human references to form a larger, augmented reference set.

The resulting reference set demonstrates higher correlation with human ratings of system outputs compared to the original reference set, across multiple models as well as several automatic evaluation metrics such as BLEU, METEOR and BERTScore.

Our approach was based on the intuition of the dialog context as an evolving microplan towards the complete dialog, with each of the human references as well as the model response representing one-step extensions. We noted that automated evaluation metrics, that only compare the realizations, cannot by themselves effectively use the dialog context. However, they can effectively use the context if an explicit, additional set of plausible one-step extensions or “rollouts” of the microplan, that serve as pseudo-references, are added to the reference set. Our approach also showed two knowledge sources that can effectively generate such extensions after a small amount of post-adaptation (see 5.2.2), namely the commonsense KB COMET and retrieved examples from the same corpus.

Our experiments confirm the efficacy of our method, not only outperforming the origi-

nal reference set but also its lexico-syntactically paraphrased expansion. This underscores the intuition that generating semantically diverse and novel extensions of the dialog context “microplan” in the manner we devise is critical to improving correlation with human judgements.

In future, we plan to incorporate other commonsense types into SCARCE, such as social [201] and moral [56]. We also hope to explore human-in-the-loop setups which build on SCARCE to collect even better references.

Broader Takeaways

The first broad takeaway from this chapter is that when performing reference-based, automatic evaluation of a NLG model, the exact quality of each individual reference need not be very high and match the level of a human authored reference. Consider the situation where the reference set keeps improving on atleast one and maintains a reasonable level of the other, amongst the two factors of:

- Mean quality
- Diversity sufficient to be representative of the distribution of possible responses

Under such circumstances, the reference-based evaluation can potentially continue to improve in terms of correlation with human evaluation, depending on the expansion strategy being used. Whether it actually improves depends on the specifics of the task under consideration, and the reference set expansion strategy employed. For instance, in this chapter, we saw earlier how SCARCE gave considerable improvements in correlation with human ratings, while PARAPHRASE , since the latter’s merely lexico-syntactic variation failed to provide any marginal benefit.

The second broad takeaway is that, for text-to-text NLG tasks such as dialog, summarization and prompt-based story generation, it is potentially useful to devise ways of involving the textual input in the reference-based automatic evaluation process, which typically only compares the reference to the response, i.e the NLG model’s output, and that too often at the surface level, given the way most popular metrics such as BLEU [151] and BERTScore [253] . This is especially true, as we in this chapter, for tasks such as dialog and prompt-based story generation, where the subspace of correct responses (outputs) given the context (input) is large. While purely reference-based evaluation which doesn’t take the input into account is forced to rely on gold standard information at the surface-realization level alone, evaluation strategies which use the input can effectively use information at the micro-planning level, by virtue of conditioning on the input. This can be done either by explicitly predicting out alternative gold

surface realizations continuing the microplan starting with the input (as we did in this chapter), or by implicitly conditioning on the input and representing it better in the evaluation process.

The third takeaway from our chapter is that it underscores the continued relevance of reference-based metrics, notwithstanding the emergence of referenceless, model-based metrics such as ADEM [120] and RUBER [226]. Though ostensibly referenceless and hence free from the variance caused by reference quality, the models underlying these metrics are often tied into specific notions of denoising, or trained by learning to discriminate between references and suites of outputs from models which are state-of-the-art only at the time of publication of the metric. As NLG models improve, the score returned by such metrics can become less discriminative over time, unable to distinguish between the multiple new candidate models under consideration which might all be considerably better than the now-erstwhile SOTA models used to learn the scoring function. Reference-based automatic evaluation is free of these concerns, and if its primary disadvantage of annotator cost is sufficiently alleviated through reference set expansion strategies like SCARCE, it can continue to remain a competitive alternative, or a useful companion, to evaluation using referenceless metrics.

5.7 Additional Results

5.7.1 Additional Correlation Results

Table 5.3 shows Spearman rank correlation scores with p-values.

5.7.2 Quality Assessment based on RUBER

As a second, automated way of ascertaining response quality, we use the unreferenced part of the RUBER metric [226], that uses a pretrained model to score quality of responses based on context alone. Here, we use the RUBER checkpoint² from [198], that first pretrains on a large Reddit dataset, followed by finetuning on DailyDialog. SINGLE and MULTI have a quality of ≈ 0.72 , while for RETRIEVAL the values is 0.63 . COMMONSENSE is found to have the most superior quality at 0.82, surpassing even MULTI.

²tinyurl.com/ynqd54tt

Spearman Rank Correlation (p-values)

Setup	1 human written reference			4 human written references			
	Dataset	SINGLE [107]	PARAPHRASE -SINGLE	SCARCE -SINGLE(Ours)	MULTI [69]	PARAPHRASE -MULTI	SCARCE -MULTI(Ours)
BLEU-4		0.093 (0.04)	0.135 (0.00)	0.302 (0.00)	0.281 (0.00)	0.269 (0.00)	0.357 (0.00)
BLEU-3		0.055 (0.22)	0.105 (0.02)	0.291 (0.00)	0.243 (0.00)	0.238 (0.00)	0.345 (0.00)
BLEU-2		0.040 (0.37)	0.082 (0.07)	0.275 (0.00)	0.203 (0.00)	0.206 (0.00)	0.327 (0.00)
BLEU-1		0.024 (0.59)	0.062 (0.17)	0.250 (0.00)	0.191 (0.00)	0.178 (0.00)	0.295 (0.00)
ROUGE-L		0.071 (0.11)	0.088 (0.05)	0.259 (0.00)	0.197 (0.00)	0.196 (0.00)	0.317 (0.00)
METEOR		0.106 (0.02)	0.094 (0.04)	0.243 (0.00)	0.227 (0.00)	0.217 (0.00)	0.299 (0.00)
EmbeddingAvg		0.030 (0.50)	0.015 (0.73)	0.025 (0.58)	0.099 (0.03)	0.096 (0.03)	0.079 (0.08)
SkipThought		-0.003 (0.95)	-0.033 (0.46)	0.087 (0.05)	0.065 (0.15)	0.053 (0.24)	0.129 (0.00)
BERT-Prec		0.270 (0.00)	0.279 (0.00)	0.378 (0.00)	0.319 (0.00)	0.322 (0.00)	0.407 (0.00)
BERT-Rec		0.096 (0.03)	0.094 (0.04)	0.240 (0.00)	0.232 (0.00)	0.212 (0.00)	0.304 (0.00)
Max. value		0.270	0.279	0.378	0.319	0.322	0.407

Table 5.3: Utterance level Spearman Rank Correlation [219] with p-values. (1) SCARCE-SINGLE augments the original single human written response (SINGLE) in DailyDialog dataset [107] using the devised method. It leads to large improvements in correlations across most of the metrics, when compared to SINGLE. (2) SCARCE-MULTI augments the MULTI dataset, again leading to improvements in correlations to human ratings, especially for BLEU and BERT-Prec metrics. Additionally, we note that almost all of the correlation values with SCARCE-MULTI are statistically significant with $p < 0.05$.

5.7.3 Diversity of References

We investigate the diversity of the references by computing self-BLEU scores [257] among references from PARAPHRASE vs SCARCE. For fair comparison, we randomly chose 4 references from corresponding method. We observe self-BLEU4 scores of 0.36 for PARAPHRASE compared to only 0.13³ for SCARCE.

5.8 Additional Details on Context Adaptation

5.8.1 Templates to convert Knowledge Base Outputs to Full Sentences

Table 5.4 lists the set of templates and rules used to transform semi-structured COMET outputs to surface natural language forms.

³Note that lower self-BLEU denotes more diverse

Condition	Action
Type is oEFFECT Example: oEFFECT (excited)	Prepend ‘I feel’ => ‘I feel excited.’
Type is oWANT Example: oWANT (to thank personx)	Prepend ‘I’ => ‘I want to thank personx.’
Type is oREACT Example: oREACT (have a party)	Prepend ‘I will’ => ‘I will have a party.’
Word PERSONX Example: i thank PERSONX.	Replace with ‘you’ => ‘I thank you.’

Table 5.4: Templates and rules to transform semi-structured COMET outputs to surface NL forms.

5.8.2 Unsupervised Decoding Procedure For Context Adaptation

We use the author’s own implementation⁴ of their DELOREAN decoding algorithm from [168]. We use default hyperparameters from their implementation, that use the non-finetuned *gpt2-medium* checkpoint as the LM atop which the unsupervised, gradient-based decoding procedure is run. Note that the model parameters are not updated in any way - the gradient computation and updates here are happening w.r.t the states, or more specifically, the state activation. More specifically, authors devise an iterative procedure wherein they alternatively perform forward and backward passes. In the forward pass, the current output Y is updated as per the likelihood of the underlying decoder. In the backward pass, the output is updated to be as similar as possible to the sentence Z from the knowledge source using back-propagation. However, since Y is discrete, we maintain a soft representation \tilde{Y} of the output Y wherein \tilde{Y}_i represents the logits at the i^{th} position as per the underlying decoder. Next, we shall describe the backward and forward passes of the iterative procedure:

- 1: In backward pass, we update logits based on the gradients of a content-matching loss function $\nabla_{\tilde{Y}} L(\tilde{Y}_{t-1}, Z)$ giving backward logits \tilde{y}_t^b
- 2: Next, we perform forward pass using the underlying decoder for steps 1 to N . During forward pass at step t , we compute the logits \tilde{y}_t^f based on left context i.e. X and $Y_{<t}$. Next we perform weighted averaging of the forward and backward logits at step t to arrive at the final logits to be used for the next time step in forward pass.

⁴tinyurl.com/2lqp9z6s

\tilde{Y}_i is initialized by performing a forward pass conditioned only on X as per greedy decoding. We alternatively perform backward and forward passes till convergence. Final response is obtained via the resulting logit outputs \tilde{Y} .

Specifically, we use their “counterfactual” setup, where an ending e_{old} is adapted from its old context c_{old} to an altered, new context c_{new} , generating a new, predicted ending \hat{e}_{new} . In our case, c_{new} is the dialog context for the turn under evaluation d_t^{past} . In the RETRIEVAL case, c_{old} is the context of the retrieved candidate turn $x_{t'}^{past}$. For the COMMONSENSE case, c_{old} is also our current context, i.e the same as c_{new} - we’re simply attuning the already drawn inference better to the current context.

5.9 Retrieval Similarity Function - Details

Consider a dialog d , broken up by turns as $\{C_1 \dots C_t, C_{t+1} = d_t^{resp}, C_{t+2} \dots C_T\}$, where $t + 1$ denotes the turn currently under evaluation. For the context-response C_t^1, \hat{r}_t pair to be evaluated, we retrieve pseudo-references based on a combination of a) Past $d_t^{past} = C_t^{t-L_b}$ b) Gold response d_t^{resp} c) Future $d_t^{future} = C_{t+2+L_f}^{t+2}$. L_b and L_f are past and future context windows. Our retrieval similarity function is a sum of the log scores between each corresponding element of the turn under evaluation with the candidate turn.

$$\begin{aligned} Sim(d_t, x_{t'}) &= \log S_{bm25}(d_t^{past}, x_{t'}^{past}) + \log S_{bm25}(d_t^{resp}, x_{t'}^{resp}) \\ &\quad + \log S_{bm25}(d_t^{future}, x_{t'}^{future}) \end{aligned}$$

We set $L_b = L_f = 2$ without specific tuning, as an intuitive tradeoff between enough specificity and enough possibility of relevant candidates.

BM25 [191] or “Best Match 25” is a tfidf like similarity. Its specific form is:

$$S_{BM25}(q, d) = \sum_{w_i \in q} \log \left(\frac{N}{df_i} \right) \frac{(k_1 + 1)tf_i}{k_1((1 - b) + b \frac{dl}{avdl}) + tf_i}$$

Here, tf_i and df_i are the term frequency in the current document and the document frequency (in the corpus). N is corpus size, while dl and $avdl$ are current and average document lengths. b controls extent of document length normalization, while k_1 controls effect of term frequency. With $b = 0$ and $k_1 \rightarrow \infty$, this reduces to simple tfidf. Here, we use default gensim values, $b = 0.7$, $k_1 = 0.5$

5.10 Qualitative Examples

In Tables 5.5, we list some examples, each illustrating a turn of a test dialog with its immediate past, future, the four additional human references from [69] (shown under MULTI 2,3 and MULTI 4,5), followed by automated response sets from different sub-components of SCARCE.

5.10.1 Before/After CONTEXTADAPT

In Example 4-4 of Table 5.5, we can observe how “*Yes , I’m young , and unmarried . It’s no problem for me to travel frequently .*” gets context-adapted (shown as +CA, short for CONTEXTADAPT) to “*Yes , I’m able to understand English . It ‘ s not that I don’t understand English .*” that indeed does match the preceding dialog better. Similarly, in Example 50-2 of Table 5.5, we can see how “*Well, that might be acceptable if you handle insurance fees*” is modulated stronger to the context that asks about duration, getting adapted to “*Well, that’s a lot of time to wait for the draft to be drawn.*”. Note that we omit this row for the examples where it simply leaves the input unchanged, or produces outputs that are noticeably unfaithful or ill-formed.

5.10.2 Complementarity of Components

Sometimes, a component may suffer from example specific issues e.g. In Example 35-2, the COMMONSENSE approaches misinterpret what is a driving ticket in the context of the dialog as an event ticket, drawing inferences accordingly. However, even in such cases, the other component salvages the situation and ensures overall response set remains healthy - e.g., here, RETRIEVAL produces pertinent responses like *Could you tell me how you dealt with it this time?*, *No I haven’t. What about you ?* etc. In Example 10-3, we see the opposite situation, where the responses from RETRIEVAL are somewhat less relevant, but COMMONSENSE produces very pertinent responses such as “*i decline the date*” and “*I go on another date*”

5.11 Human Evaluation Details

5.11.1 Quality of References

The quality of references were judged by two graduate students from a university where the medium of instruction is English. The annotators were requested to ignore minor grammar issues, and focus more on the content of the response.

No	Type	Text
0-5	CONTEXT	A: I also have blow if you prefer to do a few lines.
	FUTURE	A: come on man ! i even got dope and acid ! try some !.
	SINGLE	B: No, I am ok, really.
	MULTI 2,3	B: what is your price ? ; B: no thanks, I'm out of cash
	MULTI 4,5	B: yeah , what have you got ?; B: maybe tomorrow , i 'm running late .
	RETRIEVAL	B: Well , I guess Zina figured you couldn't blow it if you were in the hotel room .
	RETRIEVAL w CA	B: I have a few lines of blow , but it is not necessary in the way of the rest of the script .
	COMMONSENSE	B: i want to go to bathroom; B: i want to play game; B: i want to have rest; B: i want to sleep
1-1	CONTEXT	B: The taxi drivers are on strike again A: What for?
	FUTURE	A: it is really a hot potato .
	SINGLE	B: They want the government to reduce the price of gasoline .
	MULTI 2,3	B: they want more pay for their time ; B: they are not happy with their permit related expenses .
	MULTI 4,5	B: they want healthcare and benefits ; B: i don't know . i just know there are no cabs in the city
	RETRIEVAL	B: Well . Don't worry about it , I will be happy to share the taxi with you .
	RETRIEVAL w CA	B: I think the price is a little high , can't you reduce it?
		B: Well , they are not going to be able to get their fares back .
		B: I think the reason is a little more complicated than that .
4-4	CONTEXT	A: Do you think you can make yourself easily understood in English ?
	SINGLE	B: yes , in most circumstances .
	MULTI 2,3	B: my english is fine . my keyboard sometimes just acts up . ; B: yes, and i'm constantly trying to improve my communication.
	MULTI 4,5	B: i do n't know . i may be to nervous and then i 'll mess up . ; B: yes , i believe so . i 've been speaking english since birth .
	RETRIEVAL	B: Yes , i'm young , and unmarried . It's no problem for me to travel frequently .
	RETRIEVAL w CA	B: Yes , i'm able to understand English . It 's not that I don't understand English .
10-3	CONTEXT	B: Hold on , please . Let me check it for you . Yes , you're right . You will keep it for 3 days .
	FUTURE	A: Well , now I want to change the date from 24th to 28th .
	SINGLE	B: ok , that shall be arranged .
	MULTI 2,3	B: we can easily do that for you . ; B: i 'm sorry but someone's reserved the room for 28th . i can offer you a different room ?
	MULTI 4,5	B: i can extend your stay time but it may cost more since it is going into the holiday . ;
	RETRIEVAL	B: i can give you a different double room for the 28th at a discounted price .
	RETRIEVAL w CA	B: All right . May I have your name ? ; B: Apparently there is only \$ 57 left in your account . ;
		B: Here you are . What is the rate ? ; B: i prefer not to move first .
	COMMONSENSE	B: i decline the date ; B: i go on another date; B: i go on a date;
		B: i get nervous; B: i get rejected; B: i get dumped; B: i feel sad
23-10	CONTEXT	B: I mainly eat baked chicken , because there's not a lot of fat . A: that does sound pretty good .
	SINGLE	B: i know it does , and that 's because it is .
	MULTI 2,3	B: yes , it is very good and pretty healthy ; B: it is good , you should come over and i will make you dinner .
	MULTI 4,5	B: thanks , it 's been a struggle , but i am starting to see results . ;
	RETRIEVAL	B: i appreciate your kind words , i am getting used to eating more healthy .
	RETRIEVAL w CA	B: You should try it . You won't be disappointed ; B: Do you have any plans for tomorrow ? ; B: That's it ? ;
		B: Just about . ; B: I like fish . Please bring me an order of that .
	COMMONSENSE	B: i want to cook dinner; B: i want to go to movie; B: i want to write letter ;
		B: i will open your mouth; B: i will think about what you want; B: i will drink some water;
35-2	CONTEXT	B: I've never gotten one . Have you ? A: I got a ticket yesterday .
	FUTURE	A: i parked in the red zone .
	SINGLE	B: really ? what did you get one for ?
	MULTI 2,3	B: what was it for ? ; B: how much is it ?
	MULTI 4,5	B: do you resent you got a ticket ? ; B: are you going to appeal the ticket ?
	RETRIEVAL	B: I've gotten a few ; B: No , I haven't . What about you ? ; B: Could you tell me how you dealt with it this time ? ;
	RETRIEVAL w CA	B: I will explain it in detail in the next class , could you preview it ?
		B: i give ticket to you; B: i give ticket to the cashier; B: i give ticket to the clerk;
	COMMONSENSE	B: i feel happy . ; B: i feel excited . ; B: i feel annoyed . ; B: i feel disappointed . ;
		B: i see what else they can do . ; B: i see what else they can do to get the ticket; i go to the event

Table 5.5: Example context-response pairs from the test split of DailyDialog, showing the automated responses returned by different sub-components of SCARCE. CONTEXTADAPT is shortened to CA for brevity.

5.12 Computing Details

The GPUs used for COMMONSENSE and CONTEXTADAPT experiments were a Geforce Rtx 2080 and TitanX Pascal respectively.

Chapter 6

**VisCTG: Improving Plausibility of
Microplanning for Concept-To-Text
Generation Through
Retrieve-Caption-Generate
(AAAI 2022)**

[Method][Knowledge]

We're already able to see isolated cases where Cyc is learning things on its own. Some of the things it learns reflect the incompleteness of its knowledge and are just funny. For example, Cyc at one point concluded that everyone born before 1900 was famous, because all the people that it knew about and who lived in earlier times were famous people.

Doug Lenat, speaking to The Austin Chronicle in 1999, about his commonsense KB/reasoner Cyc

In this chapter, we devise and investigate enhancements to SOTA pretrained generator models to improve their microplanning ability, and consequently, their output quality. The specific approach we take to study microplanning here is in the context of accomplishing concept-to-text generation tasks (see §1.1.2) such as the *generative commonsense reasoning* task a.k.a *Commongen* [110], where the communicative goal is to generate a sentence describing a plausible situation involving a given set of input concepts. As an example, consider the input *{horse, carriage, draw}*. Two potential adequate outputs for this would be *The carriage is drawn by the horse.* and *The Sun God's carriage has seven horses drawing it.* (See Table 6.4 for more actual examples from our corpus). Specifically, we improve their abilities in terms of the lexicalization and referring expression generation subtasks, with a particular focus on pairwise lexical relationships, especially those pertaining to commonsense plausibility. We are well aware of the other outstanding challenges within these subtasks such as intra-sentence content ordering and local discourse coherence, as well as the outstanding microplanning subtask of sentence aggregation; but decide to first focus on the aforementioned subtasks and challenges given that concept-to-text generation tasks provide an ideal testbed to isolate out and study these.

We identify several critical issues in baseline model outputs for this task, like poor commonsense plausibility, inadequate pairwise lexical relationships, incomplete or missing arguments and referring expressions, and dullness/lack of specificity.

We build from a fundamental concern as a starting point to posit our ameliorating approach. Is language as a modality itself sufficient to learn the type of commonsensical pairwise lexical relationships necessary to microplan more adequate, situation-describing sentences given the input concept sets? If not, why so? And more importantly, if not, how can one address this? Would incorporating information from another modality help?

The Zipfian nature of language by itself leads to a large number of concepts with relatively fewer occurrences. For pairwise co-occurrences , this problem compounds even further. However, a second, more important concern is the well noted phenomenon of reporting bias [67] — wherein unusual, exceptional and “newsworthy” events, concepts and relationships e.g. bananas being red, are mentioned more in text corpora compared to their usual, mundane and obvious counterparts e.g bananas being yellow. Note that *reporting bias* here does not refer to any kind of *inductive bias* w.r.t. models, but the bias that exists in the medium of text itself in terms of the distribution of real world events about which text is created i.e. they are “reported” in text.

A third concern is the effect of the Gricean Maxim of Quantity, whereby speakers say only as much as is necessary, omitting information which the listener is assumed to know from commonsense. As a result, typical sentences in corpora often omit information about sufficiently obvious relationships between concepts e.g., plates being atop a table, or boats typically being afloat on an underlying water body.

We posit that these issues could indeed have a significant effect on the NLG model’s learning for CommonGen, and incorporating information from another modality such as vision could significantly help dampen this effect.

We investigate the use of multimodal information contained in images as an effective method for enhancing the commonsense of large, pretrained models for text generation. We perform experiments using BART and T5 as base NLG models. Note, however, that our method is agnostic to the nature of pretrained base architectures used, and does not exploit any particulars of transformers, masked pretraining or any other specifics of the BART and T5 architectures. We acknowledge the presence of a vast diversity of base architectures for generation in the research prior to these both, and use these two simply as a first step in demonstrating our approach’s efficacy. We call our approach *VisCTG: Visually Grounded Concept-to-Text Generation*. VisCTG involves captioning images representing appropriate everyday scenarios, and using these captions to enrich and steer the generation process.

Specifically, the intervention we devise to the E2EN2PP is the addition of an *Input Expansion Layer* between the *Input* and the *Embedding Layer*. Before passing the input string to the

Embedding Layer, the *Input Expansion Layer* symbolically augments it with the captions of retrieved relevant images described above. Figure 6.1 illustrates our intervention.

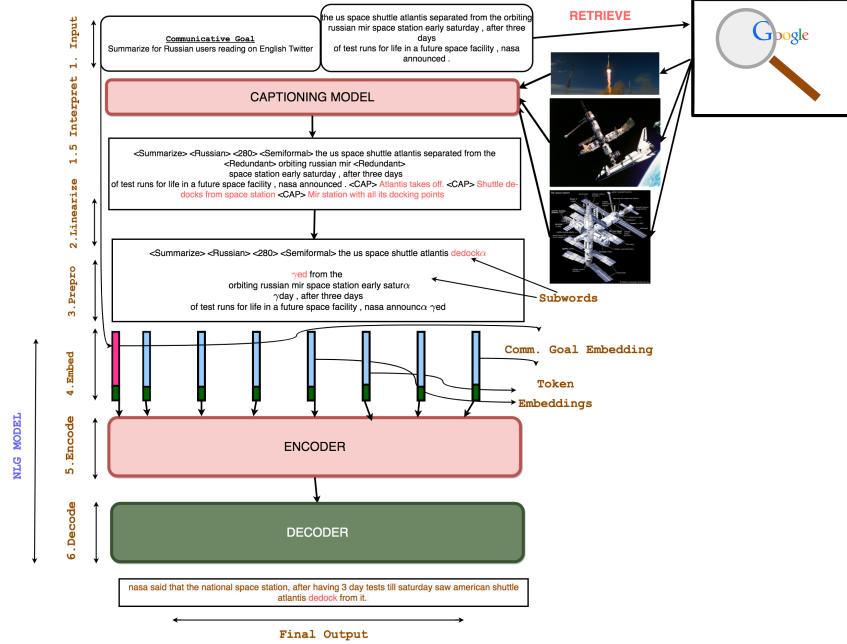


Figure 6.1: An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter 6. Here, the task is to summarize the given input news article to within 280 characters. The text marked out in carrot-red in the *Final Output*, i.e *dedocked* is clearly picked up by the model from the caption-expanded portion of the input (also marked in carrot-red)

Comprehensive evaluation and analysis demonstrate that VisCTG noticeably improves model performance while successfully addressing aforementioned issues noticed in the baseline generations.

6.1 Introduction

Large pretrained neural models have seen increasing popularity for NLP tasks and applications. This includes SOTA text generation models such as BART [104] and T5 [172]. Larger corpora and better pretraining losses are major reasons driving these gains. However, despite increasing attention on the commonsense of models through works like COMET [18], studies have shown that even large pretrained models still struggle with commonsense tasks that humans can reason through very easily [225]. We believe that there is commonsense information in

<p><i>{stand, hold, umbrella, street}</i></p> 	<p><i>{food, eat, hand, bird}</i></p> 
baseline: A holds an umbrella while standing on the street capt: a woman walking down a street holding an umbrella VisCTG: A woman stands on a street holding an umbrella.	baseline: hand of a bird eating food capt: a person holding a small bird in their hand VisCTG: A bird eats food from a hand.
<p><i>{cat, bed, pet, lay}</i></p> 	<p><i>{fence, jump, horse, rider}</i></p> 
baseline: A cat is laying on a bed and petting it. capt: a cat laying on a bed with a stuffed animal VisCTG: A cat laying on a bed being petted.	baseline: A rider jumps over a fence. capt: a horse is jumping over a wooden fence VisCTG: A rider jumps a fence on a horse.

Table 6.1: Examples of retrieved images, associated captions, baseline and VisCTG (our visually grounded model’s) generations for select concept sets. Note that the images and captions are used as an intermediary to guide the final generation and thus the final generation need not be faithful to them. E.g. there is nobody petting the cat in the image or caption, but since the VisCTG output is conditioned on both the concept set and the caption, it includes *being petted*.

other modalities like vision, beyond what is reported [67] in text, which can possibly augment commonsense and enhance decision-making processes of text generation models.

In this chapter, we show this is true by improving the performance of Transformer-based text generation models on concept-to-text generation using visual grounding, which we call *VisCTG: Visually Grounded Concept-to-Text Generation*. Concept-to-text generation is a high-level formulation of several constrained text generation and data-to-text natural language generation (NLG) tasks. These are challenging tasks that have seen increasing interest, and involve generating natural language outputs given certain pre-conditions, e.g. specific words in the outputs, or from a collection of structured or semi-structured inputs. They typically involve converting a set of inputs into natural language text. These inputs can normally be thought of as *concepts*, or high-level words or structures, that play an important role in the generated text.

CommonGen [110] involves generating sentences that effectively describe everyday scenarios from concepts sets, which are words that must appear in the output. CommonGen is

challenging as effective relational reasoning ability using commonsense knowledge is required. Models must also possess the compositional generalization capabilities to coalesce together different concepts. CommonGen is an effective benchmark for constrained text generation and commonsense as its task formulation and evaluation methodology are rather broadly applicable.

We experiment on CommonGen using BART and T5. An initial analysis (§6.3.1) of baseline generations shows several issues related to commonsense, specificity, and fluency. We hypothesize that these can be addressed through image captions (§6.3.2). Images representing everyday scenarios are commonplace, and typically logical and grounded in commonsense. Captioning models can also normally produce decent captions for everyday images, which can be used to guide and enhance the generation process. See Table 6.1 for examples.

Expounding on this, we use a pretrained image captioning model on MSCOCO captions [114] to caption the top retrieved images for each concept set (§6.4.1,6.4.2). We use these captions as additional information to augment inputs to our generation models (§6.4.3). Extensive evaluation (§6.6) demonstrates that VisCTG improves model performance and commonsense while addressing the baseline inadequacies.

6.2 Dataset, Models, and Metrics

6.2.1 CommonGen Dataset

The original CommonGen dataset is made up of 35,141 concept sets (consisting of 3 to 5 keywords each) and 79,051 sentences, split into train, dev, and test splits. Since the original test set is hidden, we partition the original dev set into new dev and test splits for the majority of our experiments. We do, however, ask the CommonGen authors to evaluate our best VisCTG models on the original test set (more in §6.6). The training set remains the same. We refer to the original dev and test sets as dev_O and test_O , and these new splits as train_{CG} , dev_{CG} , and test_{CG} . Table 6.2 contains information about these splits. Their relative sizes and distribution of concept set sizes within each are kept similar to the originals.

6.2.2 Models: T5 and BART

We use pretrained text generation models T5 and BART, both the base and large versions. Both are seq2seq Transformer models. T5 has strong multitask pretraining. BART is pretrained as

Dataset Stats		Train _{CG}	Dev _O	Test _O	Dev _{CG}	Test _{CG}
# sets	concept	32,651	993	1,497	240	360
size = 3		25,020	493	-	120	-
size = 4		4,240	250	747	60	180
size = 5		3,391	250	750	60	180

Table 6.2: Statistics of CommonGen dataset splits.

Model\Metrics	BLEU-4	CIDEr	SPICE
Reported BART-large	27.50	14.12	30.00
Reported T5-base	18.00	9.73	23.40
Reported T5-Large	30.60	15.84	31.80
Our BART-base	28.30	15.07	30.35
Our BART-large	30.20	15.72	31.20
Our T5-base	31.00	16.37	32.05
Our T5-large	33.60	17.02	33.45

Table 6.3: Comparing dev_O performance of our re-implemented models to those in Lin et al. [110]. Bold represents where we reach/exceed reported numbers. Results averaged over two seeds for our models. Lin et al. [110] did not report BART-base. See §6.2.3 for metric explanations for comparison of all metrics.

a denoising autoencoder to reproduce original from noised text. We use their HuggingFace implementations.

We train two seeded versions of each model on train_{CG} and evaluate their performance on dev_O. These serve as the baselines for our experiments. Using the numbers in Lin et al. [110] as comparison, we validate our implementations. We use the hyperparameters from Lin et al. [110], beam search for decoding, and select the final epoch as the one reaching maximum ROUGE-2 [112] on the dev split. From Table 6.3, we observe that our re-implementations reach or exceed reported results in Lin et al. [110] on most metrics.

6.2.3 Evaluation Metrics

We use several evaluation metrics, including those in Lin et al. [110] such as BLEU [151], CIDEr [228], SPICE [4], and coverage (cov). These (other than cov) assess similarity between human references and generations. In particular, CIDEr captures a combination of sentence similarity, grammaticality, saliency, importance, and accuracy. SPICE maps texts to semantic scene graphs and calculates an F-score over these graphs’ tuples. Lin et al. [110] note that SPICE correlates highest with human judgment for CommonGen. Cov measures the average percentage of input

Concept Set	Baseline Generation	Human Reference
{horse, carriage, draw}	horse drawn in a carriage	The carriage is drawn by the horse.
{dog, house, eat}	A dog eats hay in a house	The dog eats food inside the house.
{cow, horse, lasso}	A cow is lassoing a horse.	A group of men riding horses lassoing a cow.

Table 6.4: Example generations from our baseline models versus human references.

concepts covered by the output text in any form.

We also use BERTScore [253] and Perplexity (PPL). BERTScore measures BERT [40] embeddings similarity between individual tokens, serving as a more semantic rather than surface-level similarity measure. We multiply by 100 when reporting BERTScore. PPL serves as a measure of fluency, with lower values representing higher fluency. We use GPT-2 [170] for PPL. For all metrics other than PPL, higher means better performance.

6.3 Initial Analysis and Motivation

6.3.1 Baseline Model Generations

We conduct an initial analysis of the baseline model outputs, and observe that several lack fluency and commonsense plausibility. Some are more like phrases than complete, coherent sentences, e.g. “*body of water on a raft*”. Others miss important words, e.g. “*A listening music and dancing in a dark room*” misses a noun before *listening*. A large portion of generations are generic and bland, e.g. “*Someone sits and listens to someone talk*”. This may be an instance of the *dull response problem* faced by generation models [42, 229], where they prefer safe and frequent responses independent of input information.

Many generations are also implausible. For example, “*body of water on a raft*” is implausible as the phrases “*body of water*” and “*a raft*” are coalesced together incorrectly i.e., the roles of *raft* and *body of water* are incorrectly swapped. A similar issue occurs with the {*horse, carriage, draw*} example in Table 6.4. At times the models also cannot understand what certain nouns can do i.e., their affordances e.g. “*A dog checking his phone on a pier*”. Several other examples of this can be found in Table 6.4.

6.3.2 Images and Captions

Images that represent everyday scenarios are quite prevalent for almost any reasonable concept set. Further, the images are typically grounded in commonsense. For example, searching {*cow*,

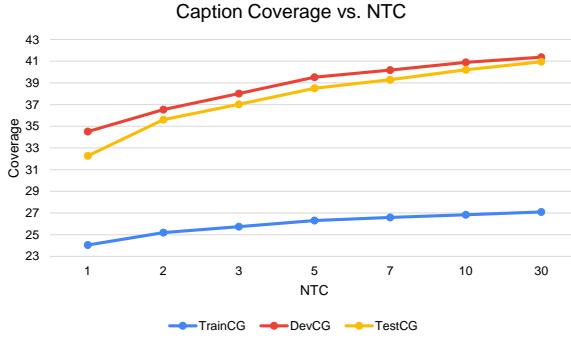


Figure 6.2: Graph displaying the average coverage (out of 100) by the top NTC captions in aggregate per concept set.

horse, lasso} will result in many images of cowboys riding horses and lassoing cows, rather than the illogical situation of “*A cow is lassoing a horse.*” described by the baseline generation in Table 6.4. Many everyday images are relatively similar to those in image captioning datasets such as MSCOCO, so pretrained captioning models should work quite effectively. We thus hypothesize that using images and their captions to visually ground concept-to-text generation can potentially deal with issues mentioned in §6.3.1. Retrieved images with corresponding captions generated by a pretrained image captioning model (see §6.4.2) and final baseline and VisCTG generations for select concept sets are in Table 6.1.

Textual corpora also suffer from *reporting bias* [67], where everyday, commonsense albeit “uninteresting” actions (walking), objects (bench) and facts (bananas are yellow) are underrepresented compared to real-world frequency, while “newsworthy” actions (murdering), objects (spaceships) and facts (blue GMO bananas) are exaggerated. This seeps even into large pretrained text models [214]. Using visual data and models dampens this bias, likely improving the commonsense of generations.

6.4 Methodology

6.4.1 Image Retrieval

We first obtain images for each concept set in our three splits. Image captioning datasets such as MSCOCO and Flickr are typically too small and focused to be effective for our purposes since we must cover numerous different concept sets. Further, a search engine is more generalizable.

We decide to use Google Images. On a sample of concept sets, the retrieved images using other search engines were inappropriate; they did not incorporate most input keywords nor

Augmented Input → Final Generation

wave fall board surfer <s> a surfer riding a wave on a surfboard → **A surfer is falling off his board into the waves.**

dance stage front crowd <s> a crowd of people watching a man on a stage <s> a man is holding a microphone in front of a crowd → **A man dances in front of a crowd on stage.**

stand hold umbrella street <s> a woman walking down a street holding an umbrella <s> a woman walking down a street holding an umbrella <s> a girl holding a pink umbrella in a city <s> a man holding an umbrella in a city <s> a group of people standing under a umbrella → **A group of people standing on a street holding umbrellas.**

Table 6.5: Examples of augmented inputs and final generations for varying values of NTC.

handle homonyms well. For example, “*sports+fan+watch*” yields images of fans watching a sports game on Google images, but images of hand watches on Bing and DuckDuckGo.

We queried input concept sets by concatenating keywords with plus signs (+), and used *simple-image-scrapers*¹ to obtain URLs of the top 30 results. The image was scraped only if the URL ended in *.png*, *.jpeg*, *.jpg*, or *.gif*. The received content was verified to be valid images using *pillow*², otherwise skipped. Retrieved images were typically of high quality and corresponded well to the concepts. See Table 6.1 for examples.

6.4.2 Image Captioning

After retrieving images, we use a PyTorch-based implementation³ of the FC image captioning model [122, 189], which generates a caption via an LSTM initialized with a pseudo token obtained by feeding the image into a deep CNN followed by a linear projection. We use a pre-trained FC model trained on the MSCOCO dataset with pretrained Resnet-101 image features. As most of our retrieved images represent everyday scenarios and are relatively similar to those in MSCOCO, the pretrained model performs quite well. See example captions in Table 6.1.

6.4.3 Caption Selection and Input Augmentation

After we have captions $S_c = \{c_1, c_2, \dots, c_n\}$ for each concept set in all three splits, we reorder them by descending coverage to the concept set to obtain $S_{c'} = \{c'_1, c'_2, \dots, c'_n\}$. If two captions are tied for coverage, we keep them in their original search result order. This allows us to select

¹<https://pypi.org/project/simple-image-download/>

²<https://pypi.org/project/Pillow/>

³<https://github.com/ruotianluo/self-critical.pytorch>

the captions that have highest coverage and are most relevant.

Since most retrieved images and corresponding captions cover only a fraction of the entire concept set, and the quality of each varies, we hypothesize that using multiple captions for generation may lead to more robust and higher-quality outputs with more coverage. The models may learn to assimilate together information from caption(s) while generating final texts. Hence, we try experiments using different numbers of top captions within $S_{c'}$, a parameter we call NTC (Number of Top Captions). We try $NTC = 1, 2, 3, 5, 7, 10$, and do not go above $NTC = 10$ as Figure 6.2 shows that coverage gains from $10 \rightarrow 30$ are minor. Figure 6.2 also illustrates that captions have relatively low individual coverage, especially compared with outputs from models trained on CommonGen, which is why we do not use them as a baseline.

The captions are concatenated together and onto the concept set using $< s >$ separator tokens. These serve as augmented inputs to BART and T5. They learn to convert these augmented inputs to human references during training, and are fed the augmented inputs (corresponding to the value of NTC) during validation and testing. Some examples of augmented inputs and generations can be found in Table 6.5.

6.5 Experiments

6.5.1 Model Training and Selection

For training VisCTG models, we mainly follow baseline hyperparameters, barring learning rate (LR) that is tuned per NTC value, and the maximum encoder length which is chosen depending on the tokenizer and value of NTC to ensure the entire input sequence can fit onto the encoder. We train two seeds per model.

For each model, we choose the epoch corresponding to highest ROUGE-2 on dev_{CG} , and use beam search for decoding. NTC itself is a hyperparameter, so while we train separate versions of each model corresponding to different NTC values, the final chosen models correspond to the NTC values that performed best on dev_{CG} when averaged over both seeds. We then use the final chosen models to generate on both $test_{CG}$ and $test_O$, and report the results in §6.6.

6.5.2 Human Evaluation

We conduct two human evaluations: one using Amazon Mechanical Turk (AMT), and one using an expert linguist. For the AMT study, we ask annotators to evaluate 86 $test_{CG}$ examples per

Metrics	BART-base ($NTC = 5$)			BART-large ($NTC = 2$)		
	Baseline	VisCTG	p-value	Baseline	VisCTG	p-value
ROUGE-1	43.96±0.03	45.44±0.08	1.58E-05	45.67±0.25	46.91±0.31	1.58E-05
ROUGE-2	17.31±0.02	19.15±0.21	1.58E-05	18.77±0.04	20.36±0.05	1.58E-05
ROUGE-L	36.65±0.00	38.43±0.07	1.58E-05	37.83±0.29	39.23±0.01	1.58E-05
BLEU-1	73.20±0.28	75.65±0.78	6.94E-05	74.45±0.21	78.80±0.28	6.94E-05
BLEU-2	54.50±0.14	59.05±0.07	6.94E-05	56.25±0.78	61.60±0.85	6.94E-05
BLEU-3	40.40±0.14	44.90±0.42	6.94E-05	42.15±0.49	47.00±0.71	6.94E-05
BLEU-4	30.10±0.14	34.10±0.57	3.82E-03	32.10±0.42	36.25±0.78	2.08E-04
METEOR	30.35±0.35	31.95±0.07	6.94E-05	31.70±0.14	34.00±0.14	6.94E-05
CIDEr	15.56±0.10	16.84±0.05	6.94E-05	16.42±0.09	18.35±0.13	6.94E-05
SPICE	30.05±0.07	31.80±0.28	6.94E-05	31.85±0.21	34.60±0.28	6.94E-05
BERTScore	59.19±0.32	61.44±0.02	1.58E-05	59.95±0.29	62.85±0.30	1.58E-05
Coverage	90.43±0.17	90.66±1.39	0.33*	94.49±0.53	96.49±0.24	1.58E-05
PPL	80.39±3.65	72.45±0.79	1.58E-05	80.37±4.51	68.46±5.90	1.58E-05

Table 6.6: Automatic eval results for BART on $test_{CG}$ over two seeds. Bold corresponds to best performance on that metric. We include stat sig p-values (from Pitman’s permutation test [163]) for VisCTG compared to the baseline. Insignificant ones ($\alpha = 0.1$) marked with *.

model. Our evaluation is based on pairwise comparison of VisCTG and baseline model outputs. We ask human annotators to choose which amongst the two outputs (presented in a random order per example) has better *Overall Quality*. There are 3 choices - O1: VisCTG is better, O2: baseline is better, O3: both are indistinguishable. To aggregate multiple annotations per example, we find the fraction of responses towards each outcome value as the per-example distribution. We then find the sample mean of this outcome distribution over all examples. For sample mean and significance testing, we are interested in the values for O1 vs. O2.

For the expert linguist study, our expert is a native English speaker with a graduate degree in linguistics from a North American university. The expert is asked to annotate three aspects for 50 BART-large⁴ $test_{CG}$ examples - *Overall Quality (Overall)*, *Commonsense Plausibility (Commonsense)*, and *Fluency (Fluency)*. For all aspects, we have a pairwise-comparison evaluation setup similar to that for AMT.

6.6 Results and Analysis

Automatic evaluation results on $test_{CG}$ are in Tables 6.6 and 6.7, and results on $test_O$ in Table 6.8.⁵ Graphs displaying BLEU-4, CIDEr, and SPICE (the metrics on the CommonGen leader-

⁴Since this is the best performing VisCTG model - see §6.6.

⁵Evaluated by the CommonGen authors on their hidden test set.

Metrics	T5-base (NTC = 2)			T5-large (NTC = 1)		
	Baseline	VisCTG	p-values	Baseline	VisCTG	p-values
ROUGE-1	44.63±0.13	46.26±0.07	1.58E-05	46.32±0.26	46.93±0.22	7.26E-04
ROUGE-2	18.40±0.14	19.78±0.30	1.58E-05	19.59±0.12	20.01±0.23	0.02
ROUGE-L	37.60±0.16	38.91±0.27	1.58E-05	39.20±0.21	39.52±0.43	0.06
BLEU-1	73.60±0.85	76.80±0.28	6.94E-05	77.55±0.35	78.65±0.21	4.65E-03
BLEU-2	57.00±0.71	60.30±0.28	6.94E-05	60.80±0.28	61.55±0.35	0.07
BLEU-3	42.75±0.49	46.25±0.64	6.94E-05	46.50±0.00	47.10±0.57	0.11*
BLEU-4	32.70±0.42	36.10±0.85	6.94E-05	36.20±0.14	36.40±0.28	0.21*
METEOR	31.05±0.49	32.70±0.00	6.94E-05	33.20±0.00	33.65±0.49	0.49*
CIDEr	16.26±0.25	17.65±0.02	6.94E-05	17.79±0.01	17.94±0.25	0.23*
SPICE	31.95±0.07	33.40±0.28	6.94E-05	33.90±0.42	34.55±0.21	0.03
BERTScore	61.40±0.34	62.42±0.17	1.58E-05	62.67±0.09	62.72±0.03	0.34*
Coverage	90.96±1.77	94.48±1.39	1.58E-05	94.40±0.02	95.95±0.45	1.58E-05
PPL	83.04±1.62	77.50±3.86	3.16E-05	81.78±4.63	73.41±4.32	1.58E-05

Table 6.7: Automatic eval results for T5 on test_{CG} over two seeds. Bold corresponds to best performance on that metric. We include stat sig p-values (from Pitman’s permutation test [163]) for VisCTG compared to the baseline. Insignificant ones ($\alpha = 0.1$) marked with *.

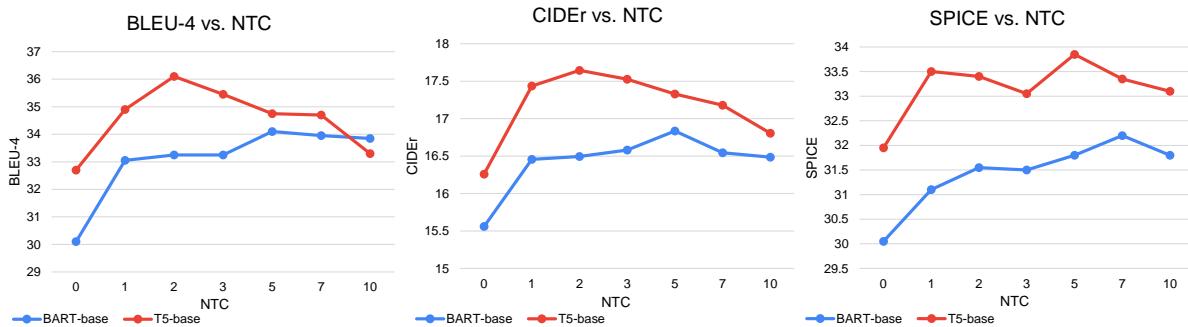


Figure 6.3: BLEU-4, CIDEr, and SPICE on test_{CG} over different values of NTC for BART-base and T5-base.

board⁶) on test_{CG} over different NTC values are in Figure 6.3. Human evaluation results on test_{CG} are in Tables 6.9 and 6.10. Optimal NTC values for BART-base, BART-large, T5-base, and T5-large are 5, 2, 2, and 1, respectively. These are the VisCTG results reported in the aforementioned tables. Table 6.11 contains qualitative examples.

6.6.1 Analysis of Automatic Evaluation Results

We see from Tables 6.6 and 6.7 that VisCTG outperforms the baselines on all metrics across the models on test_{CG}. Performance gains are strong and statistically significant for BART-base,

⁶<https://inklab.usc.edu/CommonGen/leaderboard.html>

Models\Metrics	ROUGE-2/L		BLEU-3/4		METEOR	CIDEr	SPICE	Coverage
T5-base (reported baseline)	14.63	34.56	28.76	18.54	23.94	9.40	19.87	76.67
T5-large (reported baseline)	21.74	42.75	43.01	31.96	31.12	15.13	28.86	95.29
BART-large (reported baseline)	22.02	41.78	39.52	29.01	31.83	13.98	28.00	97.35
EKI-BART [47]	-	-	-	35.945	-	16.999	29.583	-
KG-BART [117]	-	-	-	33.867	-	16.927	29.634	-
RE-T5 [233]	-	-	-	40.863	-	17.663	31.079	-
T5-base VisCTG	22.83	44.98	45.749	34.722	31.809	16.173	28.808	92.92
T5-large VisCTG	23.83	45.76	47.376	36.409	33.012	16.815	29.629	95.54
BART-base VisCTG	21.73	43.43	43.235	32.291	30.86	15.187	27.403	88.98
BART-large VisCTG	23.68	45.07	48.031	36.939	33.215	17.199	29.973	94.86

Table 6.8: Automatic eval results of VisCTG models on test_O, evaluated by CommonGen authors. We compare to reported baseline numbers in Lin et al. [110] (they did not evaluate BART-base), and models on their leaderboard with publications at time of writing that outperform baselines. Their leaderboard reports BLEU-4, CIDEr, and SPICE. Bold corresponds to best performance (for those three) per model type+size.

Model	O1	O2	O3	IAA
BART-base	0.45	0.33	0.22	0.72
BART-large	0.62	0.18	0.20	0.55
T5-base	0.46	0.33	0.21	0.72
T5-large	0.46	0.34	0.20	0.74

Table 6.9: Avg. AMT eval results on test_{CG} for *overall quality*. O1: VisCTG wins, O2: baseline wins, O3: both indistinguishable. Bold corresponds to higher fractional outcome between O1 and O2. All results are statistically significant based on paired two-tailed t-tests and $\alpha = 0.1$. The inter-annotator agreement (IAA) is the average direct fractional agreement (where both annotators choose O1 or O2) over all examples. See §6.5.2 for further details.

Model	Aspect	O1	O2	O3
		Overall	0.44	0.24
BART-large	Commonsense	0.32	0	0.68
	Fluency	0.56	0.12	0.32

Table 6.10: Avg. expert linguist eval results on test_{CG} for BART-large. O1: VisCTG wins, O2: baseline wins, O3: both indistinguishable. Bold corresponds to higher fractional outcome between O1 and O2 per aspect. See §6.5.2 for further details.

BART-large, and T5-base. VisCTG appears relatively less effective for T5-large which is the strongest baseline, and hence improving its performance may be more difficult.

From Table 6.8, we see that VisCTG models substantially outperform corresponding baselines reported in Lin et al. [110] on test_O. T5-base VisCTG outperforms the reported T5-base and large baselines across metrics, and BART-base VisCTG performs similarly to the reported

Method	Text
Concept set	{sit, chair, toy, hand} (example 1)
Captions	a little girl sitting on a chair with a teddy bear <s> a small child sitting on a chair with a teddy bear <s> a young boy sitting on a chair with a skateboard <s> a man sitting on a chair with a remote
BART-base-BL	hands sitting on a chair
BART-base-	A boy sitting on a chair with a toy in his hand.
VisCTG	
Human reference	A baby sits on a chair with a toy in one of its hands.
Concept set	{food, eat, hand, bird} (example 2)
Captions	a bird is perched on a branch with a hand <s> a person holding a small bird in their hand
BART-large-BL	hand of a bird eating food
BART-large-	A bird eats food from a hand.
VisCTG	
Human reference	A small bird eats food from someone's hand.
Concept set	{bench, bus, wait, sit} (example 3)
Captions	a man sitting on a bench with a book <s> a person sitting on a bench with a laptop
T5-base-BL	A bus sits on a bench.
T5-base-VisCTG	A man sits on a bench waiting for a bus.
Human reference	The man sat on the bench waiting for the bus.
Concept set	{jacket, wear, snow, walk} (example 4)
Captions	a young boy in a red jacket is standing in the snow <s> a man in a red jacket is standing in the snow
BART-large-BL	walking in the snow wearing a furry jacket
BART-large-	A man is walking in the snow wearing a jacket.
VisCTG	
Human reference	Jamie took a walk out into the snow with only a T shirt on and instantly went back inside to wear his jacket.
Concept set	{hold, hand, stand, front} (example 5)
Captions	a man holding a pair of scissors in front of a wall
T5-large-BL	Someone stands in front of someone holding a hand.
T5-large-VisCTG	A man stands in front of a man holding a hand.
Human reference	A man stands and holds his hands out in front of him.
Concept set	{bag, put, apple, tree, pick} (example 6)
Captions	a person holding a apple in a tree <s> a bunch of apples are growing on a tree <s> a close up of a green apple with a tree <s> a bunch of apples are growing on a tree
BART-base-BL	A man is putting apples in a bag and picking them up from the tree.
BART-base-	A man puts a bag of apples on a tree.
VisCTG	
Human reference	I picked an apple from the tree and put it in my bag.

Table 6.11: Qualitative examples for test_{CG}. BL stands for baseline. Concept set refers to the input keywords and Captions refers to the captions (separated by <s>) used by the VisCTG model for that particular example to produce its final generation.

BART-large baseline. BART-large VisCTG outperforms the reported baseline, EKI-BART [47], and KG-BART [117]. These are SOTA published CommonGen BART models that use external knowledge from corpora and KGs. We show that visual grounding is more effective, and BART-large VisCTG would place very high on the leaderboard.⁶ T5-large VisCTG outperforms the reported baseline, but lags behind the SOTA published RE-T5 [233].

Figure 6.3 shows that as NTC increases, BLEU-4, CIDEr, and SPICE increase to a peak, and taper off after. This is expected as we saw in Figure 6.2 that the rate of increase of coverage declines with larger NTC. The latter images and captions are of diminishing quality, and hence using too many negatively affects model performance.

6.6.2 Analysis of Human Evaluation Results

Table 6.9 shows that VisCTG outperforms the baseline on all four models based on human annotators (with high IAA). Annotators, on average, prefer VisCTG outputs over baseline outputs on overall quality, especially for BART-large. Table 6.10 illustrates that VisCTG outperforms the baseline model for BART-large based on an expert linguist’s perspective. VisCTG outputs are highly preferred, on average, over the baseline on all three aspects of overall quality, commonsense, and fluency. This aligns with our automatic results in §??, where VisCTG outperforms the baselines across all models.

6.6.3 Qualitative Analysis

Table 6.11 shows several baseline outputs that contain issues from §6.3.1, e.g., incomplete and/or illogical sentences. Human references are all fluent and logical. VisCTG can usually generate much higher-quality text than the baselines.

The baseline outputs for ex. 1-2 are phrases lacking arguments, and all illogical for ex. 1-3. Using captions, VisCTG successfully adjusts semantic roles of entities, replaces incorrect subjects, fixes dependency structure, and grounds generations in commonsense. For ex. 1, captions are of the form “{X} sitting on a chair with {Y}”, where {X} is a subject and {Y} an object. VisCTG output has similar structure, being fluent and logical with higher coverage. The baseline output also has an incorrect subject of “hands”. Our VisCTG output contains an additional entity (not present in the input set) of “boy” as subject, likely since it is a subject in the captions. This highlights the usefulness of visual grounding, as the image space can provide additional commonsense information not present in the text (e.g. toys are associated with children/boys). For ex. 2, the baseline output treats “hand of a bird” as a single entity, the subject. Captions

separate “*bird*” and “*hand*” into two, likely guiding the VisCTG output to do so. For ex. 3, the baseline misplaces “*bus*” as subject. Captions are of form “{X} *sitting on a bench* {Y}”, where {X} is a logical subject and {Y} is an expression. The VisCTG output has this structure, with correct subject and commonsense, and higher coverage. Overall, we see that visual grounding guides the model to learn which nouns/subjects can perform which actions (e.g. “*hands*” cannot sit on a chair but a “*boy*” can), which is a major baseline deficiency discussed in §6.3.1.

For ex. 4, the baseline output lacks a subject that the captions contain, likely guiding the VisCTG output to contain one: “*a man*”. For ex. 5, the baseline output is generic due to uses of “*someone*”. VisCTG’s output is more specific and refers to “*man*”, likely because the caption (though not very fitting) includes a “*man*” subject. Even for captions that fit the concepts less, structure and fluency can still be exploited.

Overall, we see that the baselines simply try to coalesce together the input concepts into a form of English syntax, often failing to do so effectively. VisCTG models can produce more grammatical, fluent, and logical text by exploiting the syntactic and dependency structures of the captions. Further, the visual grounding improves the commonsense of the generations. The images inherently capture commonsense by representing everyday scenarios, and this commonsense info is rarely explicitly included in text. Hence, large text-based models such as our baselines tend to not know this info, whereas VisCTG models learn it through the grounding.

VisCTG is, however, still a far way off from perfect. For ex. 6, its output is less logical and lower coverage than the baseline’s. The captions are all simplistic and low coverage; the first is illogical, and some others are of the form “*a bunch of apples* {...} *on a tree*”, likely negatively impacting the generation. Ex. 4’s human reference is creative, which is an area where VisCTG still lacks in comparison. For ex. 5, while VisCTG edits “*someone*” to “*man*”, it is unable to merge the two instances of “*man*” or adjust the sentence to be more coherent. These weaknesses are likely because captions tend to be simplistic (due to the captioning model’s training data), limiting VisCTG’s ability to make heavier edits. VisCTG, unsurprisingly, appears to depend quite heavily on the captions, and hence the quality of the images and captioning model.

6.7 Related Work

Constrained Text Generation: There have been several works on constrained text generation. Miao et al. [136] use Metropolis-Hastings sampling to determine Levenshtein edits per generation step. Feng et al. [50] devise Semantic Text Exchange to adjust topic-level text semantics.

Data-to-text NLG: E2E-NLG [44] and WebNLG [64] are two popular NLG benchmarks with structured inputs - meaning representation (MR) and triple sequences, respectively. Montella et al. [138] use Wiki sentences with parsed OpenIE triples as weak supervision for WebNLG.

Commonsense Injection and Incorporation: One large commonsense knowledge graph (KG) is COMET, trained on KG edges to learn connections between words and phrases. EKI-BART [47] and KG-BART [117] use external knowledge (from corpora and KGs) to improve BART’s performance on CommonGen. Distinctly, VisCTG uses visual grounding and shows higher performance (see §6.6). Visual Commonsense Reasoning (VCR) [250] involves answering commonsense-related multiple-choice questions about images. Our work uniquely focuses on injecting commonsense into seq2seq Transformer models like BART and T5 for text generation.

Multimodal Machine Learning and NLP: There has been more work on multimodality, in areas like representation and video captioning, but little for constrained and data-to-text NLG [10, 63]. There is work on pretrained multimodal models like ViLBERT [121], which are mainly encoders that jointly represent images and text rather than seq2seq models, and would be ill-suited for generation. Further, unlike these models which are pretrained, VisCTG exploits per-example visual information to fix specific issues for each concept set.

6.8 Conclusion and Future Work

In conclusion, we motivated and explored the use of visual grounding for improving the microplanning abilities of Transformer models for concept-to-text generation tasks. We christen our method VisCTG: Visually Grounded Concept-to-Text Generation. Extensive experiments on BART and T5 showed its efficacy on the CommonGen task. Comprehensive evaluation and analysis showed that VisCTG boosts model performance and commonsense while addressing baseline deficiencies.

Our empirical findings support our hypothesis that relying on language alone is insufficient to learn a good NLG model for CommonGen and could cause the issues we observed in our baseline outputs e.g those in Table 6.4 . They also confirm our intuition that incorporating information from the visual modality can in part ameliorate this insufficiency. Furthermore, they support the case for intervening in the E2ENLP and introducing an *Input Expansion Layer* between the *Input Layer* and *Embedding Layer* to symbolically augment the input with captions of retrieved images, as described in Figure 6.1.

Potential future work includes improving image search and captioning, e.g. better selection of images during retrieval or using a stronger captioning model. Video captioning and image generation rather than retrieval can also be explored. Further, VisCTG can be investigated for other data-to-text NLG tasks, e.g. WebNLG.

In this chapter, we successfully devised enhancements to SOTA pretrained generator models to improve their microplanning ability, and consequently, their output quality, while accomplishing concept-to-text generation tasks (see §1.1.2) such as the *generative commonsense reasoning* task a.k.a *Commongen* [110].

Our work supported the case for two ways of intervening in the end to end NLG pipeline to improve microplanning abilities of such NLG models - namely incorporating i) an *input expansion* stage and ii) Having a feedback loop from the NLG model’s *Output Layer* to its *Input Layer*.

6.8.1 Broader Takeaways

The broader takeaway from our findings is that in any task where the communicative goal (or the “input text” part of the communicative goal) leaves gaps w.r.t. the relations amongst different parts of the input to be filled in a “plausible”, commonsensical kind of way, reporting bias is naturally bound to be a problem for any model trained on typical natural language corpora. Examples of such input information include sets of concepts (as in our case), recipes or Wikipedia infoboxes. In such situations, using another modality (which could be something like images/audio, or even another language) which has *lesser* or *different* reporting bias, to “expand” the underspecified input information is a potential architectural enhancement to explore in order to improve the microplanning i.e the plausibility and internal structure of sentences. Implementing this kind of input “expansion” requires a mechanism to ground the input into the other modality, and then reground it back. In the case of concept-to-text generation tasks, the simple nature of the input information and the availability of well optimized search engines greatly simplifies and streamlines the grounding process, which is unlikely to be as straightforward in the general case. Furthermore, even the regrounding process is simplified due to the presence of well-developed captioning models.

As an example, consider the task of summarizing social media posts from forums/subreddits related to a religion with its primary mode of religious discourse and scripture being a non-English one e.g., Islam or Judaism, who have the bulk of their scriptures, commentaries and other resources in Arabic and Hebrew respectively, which we shall refer to henceforth as the

scriptural language. In this case, one potential architectural enhancement based on the same principle as ours, would be as follows:

1. Translate the input social media post x to the scriptural language using an off-the-shelf English→scriptural language translation model.
2. Take the translated input $T_{scriptural}(x)$ and use it to retrieve similar sentences from any large corpora of religious texts, commentary, discourse in the scriptural language.
3. Translate back each of the retrieved sentences $s_{candidate}^i \in Retrieve(T_{scriptural}(x))$ to English using an off-the-shelf scriptural language→English translation model.
4. Just as we did with the captions of retrieved images, augment the input x with the set of top K most relevant from amongst the back-translated candidates $T_{English}(s_{candidate}^i)$.
5. Train the models with the now-augmented inputs.

Here, the scriptural language plays the same role as images/visual modality in this chapter. Since the scriptural language is more likely to have a wider coverage and range of religious terminology, arguments and text, it would naturally suffer from lesser reporting bias. The English→scriptural language and scriptural language→English models serve as the grounding and regrounding mechanisms respectively.

The second takeaway from our findings is that they underscore the added potential utility of joint spaces which embed together different modalities, such as the very recent CLIP representation [171] from OpenAI. Instead of using retrieval from a search engine as in our case, one can directly compute cross-modal similarities in this space, e.g., between a given input text and an image.

6.8.2 Deeper Theoretical Questions

Here we highlight some deeper theoretical questions this chapter and the intuition/rationale behind it, and the fact that the resultant architectural improvements work, raise. We do not have the moment have an answer to these questions, nonetheless we thought it was worthwhile to document them.

1. How does one quantify reporting bias of a modality/language? Given two modalities, is there a theoretically sound way of quantifying which one has more & which one has lesser reporting bias?
2. How does one quantify if the reporting biases of two modalities exactly overlap or if they are disjoint? When one combines information from two modalities, what is the reporting

bias of the augmented information space?

In the case of text and vision, beyond intuition, one can also present the argument that their reporting biases are bound to be different since they represent two different ways of sampling the world - the motivation for taking an image of an object or a set of objects is different from that of writing about it in text. Furthermore, in vision, one always ends up unintentionally capturing additional objects in images without explicitly intending to do so - this is because one cannot solely have images of an object in focus without it interacting with other objects based on its affordances, as well as having a background, both of which then appear in the image.

6.9 Appendices

Model\Metrics	ROUGE-2/L		BLEU-3/4		METEOR	CIDEr	SPICE	BERTScore	Cov
Reported BART-large	22.13	43.02	37.00	27.50	31.00	14.12	30.00	-	97.56
Reported T5-base	15.33	36.20	28.10	18.00	24.60	9.73	23.40	-	83.77
Reported T5-Large	21.98	44.41	40.80	30.60	31.00	15.84	31.80	-	97.04
Our BART-base	15.91	36.15	38.30	28.30	30.20	15.07	30.35	58.26	93.44
Our BART-large	17.27	37.32	39.95	30.20	31.15	15.72	31.20	58.58	95.03
Our T5-base	17.27	37.69	41.15	31.00	31.10	16.37	32.05	60.32	94.44
Our T5-large	17.90	38.31	43.80	33.60	32.70	17.02	33.45	61.39	96.26

Table 6.12: Performance of our re-implemented CommonGen models on dev_O compared to the original numbers reported in Lin et al. [110]. Note that for our models, results are averaged over two seeds, and that the original authors did not experiment with BART-base or report BERTScore. Bold indicates where we match or exceed the corresponding reported baseline metric.

6.10 Full Re-implementation versus Reported Model Numbers

See Table 6.12 for a full comparison (across all metrics) of our re-implemented CommonGen models compared to the original reported baseline numbers in Lin et al. [110].

Instructions: Commonsense Situation Description Evaluation Study (Click to expand)

Read the instructions given below slowly and carefully:

Below you are given three questions. In each question, we show you a **conceptset (C)** and **two sentence descriptions (S_A,S_B)** of situations using this conceptset. The **Conceptset C** is a group of keywords of everyday, basic objects, concepts or things (table, toothbrush, dog, fence etc) In Sentence S_A/S_B, our **models A and B** each try to **construct and describe a real-world situation** using these **concepts and things**. Models A and B **may not always succeed** in this attempt. You may see all the below cases happening

- 1) Sometimes, the situation they construct maybe very sensible and real-world. Sometimes, it may sound nonsense or imaginary.
- 2) In some other cases, the situation maybe sensible but the model may not have written that fluent or nice-sounding a description.
- 3) Sometimes, the situation description will have covered all the keywords/concepts well. But some other times, it may forget to include some concepts properly, even though the description is good otherwise.

We want you to read what has been written in S_A/S_B and tell us which one is better overall i.e, which of model A and B has done better for this example. Though it is upto your judgement how to decide this exactly, you may consider these 3 aspects 1) How Commonsense/Real-World the situation is 2) Fluency of the situation Description 3) Coverage of the ConceptSet.

Q.1 Answer the questions asked about the conceptset C1 and situation description pair S1_A, S1_B given below. We ask you to pick which of Model A vs Model B does better.

(a)

ConceptSet C1

{ shirt || wear || microphone || singe }

Situation Description SA_1

A man wearing a white shirt sings at a microphone.

Situation Description SB_1

A man wearing a white shirt sings into a microphone.

Q 1.1 Which of SA_1 and SB_1 is the better and more appropriate situation description overall, after you've considered various factors like fluency, commonsense plausibility and coverage as per your judgement?

A

B

Hard to prefer one over the other

(b)

Figure 6.4: Snapshots of human evaluation: a) instructions seen by annotator and b) an example with questions.

6.11 Pretrained FC Image Captioning Model Details

The image encoder is a pretrained Resnet-101 [188], where the global avg. pooling of the final convolutional layer output, a vector of dim. 2048, is taken per image. Spatial features are extracted from the output of a Faster R-CNN [5, 188] with ResNet-101 [73], trained by object and attribute annotations from Visual Genome [100]. For captioning, the dimensions of LSTM hidden state, image feature embedding, and word embedding are all set to 512. Please see Luo et al. [122], particularly Sections 3.3 and 5.1, and Rennie et al. [189], particularly Sections 2 and 5, for more.

Method	Text
Concept set	{sunglass, wear, lady, sit}
Captions	a woman sitting on a bench with a cell phone <s> a woman sitting on a bench with a book
T5-base-BL	A lady sits in a sunglass.
T5-base-VisCTG	A lady wearing sunglasses sits on a bench.
Human reference	The lady wants to wear sunglasses, sit, relax, and enjoy her afternoon.
Concept set	{music, dance, room, listen}
Captions	a person is standing in a room with a bed <s> a woman is holding a laptop in a room
BART-large-BL	A listening music and dancing in a dark room
BART-large-VisCTG	A group of people are dancing and listening to music in a room.
Human reference	A boy danced around the room while listening to music.
Concept set	{pool, water, slide, slide}
Captions	a boat is parked in a water with a boat
T5-large-BL	A girl slides into a pool and slides into the water.
T5-large-VisCTG	A group of people slide down a slide into a pool of water.
Human reference	A boy slides down a bouncy slide into a pool of water.
Concept set	{rock, water, stand, body}
Captions	a bird sitting on a rock in a body of water
T5-large-BL	a body of water standing on rocks
T5-large-VisCTG	A man standing on a rock near a body of water.
Human reference	A bird standing on a large rock in a body of water.
Concept set	{card, deck, shuffle, hand}
Captions	a person holding a cell phone in their hand <s> a person holding a pair of scissors in their hand
BART-large-BL	a hand shakes a deck of cards
BART-large-VisCTG	A man shuffles a deck of cards with his hand.
Human reference	A man shuffles a deck of cards in his hands.
Concept set	{chase, ball, owner, dog, throw}
Captions	a dog is standing in the grass with a frisbee <s> a dog is playing with a frisbee in the grass
T5-base-BL	owner throws a ball to his dog during a chase.
T5-base-VisCTG	A dog is throwing a ball at its owner.
Human reference	The owner threw the ball for the dog to chase after.
Concept set	{body, water, bench, sit}
Captions	a bench sitting on a beach next to a body of water <s> a man is sitting on a bench with a cell phone <s> a bench sitting on a of a beach <s> a bench sitting in the middle of a lake <s> woman sitting on a bench with a bird in the background
BART-base-BL	A woman sitting on a bench with water in her body.
BART-base-VisCTG	A man sits on a bench near a body of water.
Human reference	The woman sat on the bench as she stared at the body of water.
Concept set	{bench, sit, talk, phone}
Captions	a man sitting on a bench with a cell phone <s> a woman sitting on a bench with a cell phone <s> a man sitting on a bench with a cell phone <s> a person sitting on a bench with a skateboard <s> a man sitting on a bench with a laptop
BART-base-BL	A man sitting on a bench talking to his phone.
BART-base-VisCTG	A man sitting on a bench talking on his cell phone.
Human reference	The woman sits on the bench to talk on her daughter on the phone.

Table 6.13: Further qualitative examples for test_{CG}. *BL* stands for baseline. *Concept set* refers to the input keywords and *Captions* refers to the captions (separated by <s>) used by the VisCTG model for that particular example to produce its final generation.

6.12 BART and T5 Model Training and Generation Details

T5-large has 770M params, T5-base 220M params, BART-large 406M params, and BART-base 139M params. Two seeded versions of each baseline and VisCTG model are trained. For decoding, we use beam search with a beam size of 5, decoder early stopping, a decoder length penalty of 0.6, a decoder maximum length of 32, and a decoder minimum length of 1 for all models. We

use a maximum encoder length of 32 for the baselines and for the VisCTG models: up to 160 for BART and 256 for T5. A batch size of 64 for T5-base and BART-base, 32 for BART-large, and 8 for T5-large is used for training. We 500 warmup steps for BART-large, and 400 for T5-base, T5-large, and BART-base. All models are trained up to a reasonable number of epochs (e.g. 10 or 20) and early stopping using our best judgment is conducted, e.g. if metrics continuously drop for several epochs. Learning rates for VisCTG models were determined by trying several values (e.g. from 1e-6 to 1e-4), and finding ones which result in decent convergence behavior, e.g. dev metrics increase steadily and reach a maximum after a reasonable number of epochs. For the final models (e.g. best NTC values for VisCTG), learning rates are (each set consists of {BART-base,BART-large,T5-base,T5-large}): baselines = {3e-05,3e-05,5e-05,2e-05}, VisCTG = {1e-05,5e-06,2e-05,2e-05}.

Google Colab instances were used for training, which used either a single V100 or P100 GPU. Most of the training experiments were performed using a single V100. BART-base models trained in approx. 1 hour, T5-base models in approx. 1.5 hours, BART-large models in approx. 2 hours, and T5-large models in approx. 6 hours.

6.13 Human Evaluation Details

The Amazon Mechanical Turk (AMT) human evaluation was performed through paid annotators on AMT. Annotators were from Anglophone countries with > 97% approval rate. Each example was evaluated by up to three annotators. Each AMT task page or HIT contained 2 actual examples and a “quality-check” example in random order. Specific instructions and a question snippet can be seen in Figure 6.4.

On every annotation page, we include one randomly chosen “quality-check” example from a list of such hand-crafted examples, in addition to two actual examples with VisCTG and baseline outputs. The hand-crafted examples are constructed to have an obviously good and an obviously bad output pair, and are sourced from Lin et al. [110]. If an annotator answers the quality-check question wrong (e.g. they choose the obviously bad output), their two remaining actual example annotations are excluded while compiling results.

The time given for each AMT task instance or HIT was 8 minutes. Sufficient time to read the instructions, as calibrated by authors, was also considered in the maximum time limit for each HIT/task. Annotators were paid 98 cents per HIT. The rate of payment (\$7.35/hour) exceeds the minimum wage rate for the USA (\$7.2/hour) and hence constitutes fair pay. We neither solicit, record, request, or predict any personal information pertaining to the AMT crowdworkers.

The expert linguist evaluation included a human subject institutional board protocol and a rate of payment of \$15/hour, also exceeding the minimum wage rate for the USA.

6.14 Further Qualitative Examples

See Table [6.13](#) for further qualitative examples.

March 25,2022

Chapter 7

Better Microplans For Concept-to-Text Tasks By Self Introspecting Input Expansion (Under Preparation) [Method]

I think it's very important to have a feedback loop, where you're constantly thinking about what you've done and how you could be doing it better

Elon Musk, in an interview to *Mashable*

7.1 Introduction

In this chapter, we devise and investigate enhancements to SOTA pretrained generator models to improve their microplanning ability, and consequently, their output quality, while accomplishing concept-to-text generation tasks (see §1.1.2) such as the *generative commonsense reasoning* task a.k.a *Commongen*. Specifically, we improve their abilities in terms of the lexicalization and referring expression generation subtasks, with a particular focus on pairwise lexical relationships, especially those pertaining to commonsense plausibility.

As we saw in the earlier Chapter 6, through our initial qualitative as well as qualitative analysis (See 6.3.1), model outputs even from otherwise state of the art, large pretrained generators such as BART [104] and T5 [173] suffer from problems such as poor commonsense plausibility, inadequate pairwise lexical relationships, incomplete or missing arguments and referring expressions, and dullness/lack of specificity.

In that chapter, we had devised a model-agnostic strategy of augmenting the input concept set by routing through the visual modality using a combination of retrieval and captioning. In this chapter, we explore an entirely alternative family of potential model-agnostic augmenting strategies to alleviate the aforementioned shortcomings. Our approach is based on the observation that large, pretrained generator models, by virtue of having been trained to predict masked out words given their contexts on large corpora such as BookCorpus, also tend to acquire a measure of factual and commonsense knowledge. This has been shown by a wide body of prior research [160, 213] that evaluates their ability to predict out masked object or subject spans from lexicalization of triples e.g., *[MASK] was the Chancellor of Prussia*, the triples being drawn from knowledge bases like Freebase. [16].

We aim to devise an overarching model architecture based on breaking the aggregate process of generation into two passes through the base model (e.g., BART), each of which can distinctly leverage one of the two abilities of these NLG models, in order, i.e., i) as a concept augmentation/expansion mechanism ii) as a text sequence to text sequence transducer i.e., in other words, a generator. Such a formulation makes increased sense in particular for concept-to-text generation tasks, especially CommonGen, since the communicative goal is to construct a sentence describing a sufficiently complete, commonsense plausible situation involving all the given input concepts e.g., *The Sun God’s carriage has seven horses drawing it.* given *{horse, carriage, draw}*. Since the number of input concepts is typically between 3 to 5, the input is often underspecified, and a large number of situations (both plausible and implausible) can be potentially constructed by bringing in associated additional concepts. Thus, a generation model has the two-fold role of

1. Constructing an appropriate set of concepts sufficient to build one or more plausible situations involving them
2. Composing the concepts together into a fluent, plausible and coherent sentence

Consider, for example, the input *{jacket, wear, snow, walk}*. With just the baseline model, we get the output “*walking in the furry snow wearing a jacket.*”. We can see this sentence as somewhat plausible, though lacking fluency by having a missing subject and a misplaced asso-

ciation of *furry* with *snow*. Running keyword extraction on this output sentence would give us the concept *furry* in addition to those already in the input concept set. Now, after augmenting the input concept set with *furry*, we again run the baseline model with this augmented concept set as input, this time producing the output sentence “*A man is wearing a furry jacket as he walks in the snow*”. This output sentence is fluent as well as plausible.

7.2 Proposed Approach

7.2.1 Inference

At inference time, in the first pass, we only use the model as if one were querying a knowledge base – given the input concept set, we let the NLG model first generate its output as usual. Then, we use surface-level concept extraction heuristics, specifically, the KeyBERT keyword extraction algorithm ¹ to extract out keywords from this output. By only extracting keywords from this initial model output, we suppress its abilities as a generator for the time being, and only use it to extract associated concepts. Treating the extracted keywords as additional concepts, we augment them to the original input concept set, hence effectively expanding it. With this augmented input concept set, we now get the second stage NLG output, which we treat as our final result.

7.2.2 Finetuning

Using the above approach during finetuning (i.e., the task-specific training of the pretrained generator model) , however, could lead to two potential pitfalls, affecting the convergence as well as time of the training process, as we explain next.

1. Since the model is still adapting itself to the input format and domain of the task inputs, as well as learning the conditional dependence of reference text on input text, its own outputs (i.e., those inferred by decoding from the model, rather than the references) are likely to be ill-formed during the initial few training steps. Extracting keywords from these ill-formed and high-variance outputs and feeding them back into the model through input augmentation is likely to affect the stability and convergence of initial finetuning.
2. Training splits are typically much larger (in our case ≈ 20 times) than test splits in terms of number of examples. Furthermore, we iterate over each example many times due to

¹<https://github.com/MaartenGr/KeyBERT>

the finetuning process spanning over multiple epochs. As a result, the cumulative additional time entailed by having to do model inference/decoding for each training step is considerable.

Hence, at finetuning time, to simplify the training process and circumvent these pitfalls , we simply extract keywords directly from reference(s) corresponding to each training example.

In summary, our approach in this chapter devises a beneficial way of intervening in the typical E2EN2PP to improve microplanning abilities of the NLG model. Figure 7.1 illustrates the interventions devised. Though we restrict our experiments to BART and T5 as the underlying models, our input augmentation approach can potentially work as an enhancement in tandem with any sufficiently large and significantly pretrained model architecture with known KB-like abilities as discussed in §7.1.

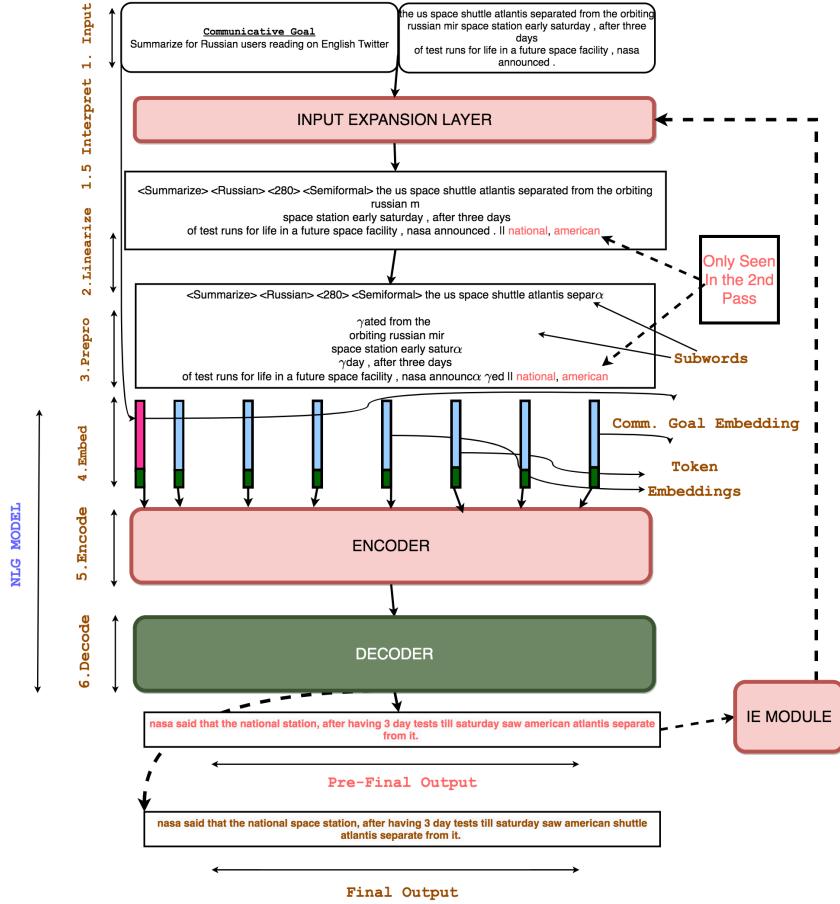


Figure 7.1: An illustration of how the E2EN2PP fleshed out in Figure 1.2 would work in action for the actual generation task and input example, after incorporating the Intervention in Chapter ???. Here, the task is to summarize the given input news article to within 280 characters. The dotted arrows represent the second pass of inference, after expanding the input with information extracted by the IE module from the Stage 1/ “Pre-Final” output (marked in red)

7.3 Intended Conclusion

Thus, by a two stage decoupling of the pretrained models abilities as a associative knowledge store and a transduction model, we posit that we shall be able to significantly alleviate the problematic aspects of the one-stage outputs from the same models, enhancing its microplanning abilities.

We expect to see these relative improvements reflected both in automatic metric based and human assessments of our devised model outputs as compared to their corresponding baseline model’s outputs.

March 25,2022

Part III

Macroplanning

Chapter 8

Viable Content Selection and Refex Generation Through Pragmatic Backoff For Chess Commentary Generation (ACL 2018)

[Knowledge][New Task]

When one shows someone the king in chess and says: “This is the king”, this does not tell him the use of this piece – unless he already knows the rules of the game up to this last point: the shape of the king. You could imagine his having learnt the rules of the game without ever having been shown an actual piece. The shape of the chessman corresponds here to the sound or shape of a word.

Ludwig Wittgenstein, *Philosophical Investigations*

In this chapter, we formulate a novel NLG task that in terms of the challenges it poses with respect to the classical NLG pipeline, sits at the boundary of macroplanning and microplanning

— it both covers the macroplanning sub-challenge of content selection (what to say?) and the microplanning sub-challenge of referring expression generation (how to refer to the various entities, events, relationships and arguments about which you have chosen to say?).

Specifically, we present and examine the problem of generating natural language descriptions of chess game moves. Our communicative goal is to generate a short, interesting commentary after a given single move in an ongoing chess game, given the pre-move and post-move board states as input information. For this task, we introduce a new large-scale chess commentary dataset and devise methods to generate commentary for individual moves in a chess game. The introduced dataset consists of more than 298K chess move-commentary pairs across 11K chess games.

Consider the various repercussions that a single move in a chess game engenders. Even a single move can change many inter-piece relationships and piece states in the game, including those between pieces that did not themselves change position during the move. (e.g., a black rook can threaten the white knight once a black knight blocking the horizontal path between them moved). The NLG model faces three key challenges:

1. What type of comment to make? One can describe the move and the game itself (*Move Description*), describe the quality of the move (*Move Quality*). We assume the desired comment type to be additionally given as part of the input, and augment our dataset input to address the same.
2. What to comment on out of the many updated states and relationships so that its interesting from the game’s perspective? This is related to the *content selection* subtask that in turn falls in the *macroplanning* stage.
3. How to address and refer to the interesting pieces and their relationships in an interesting way from the game’s perspective? This is related to the *referring expression generation* subtask from microplanning.

In order to address challenges 2 and 3, the NLG model encoder has to learn to encode the 2 8×8 board states in a game-pragmatically sensitive way. Furthermore, it has to accomplish this *tabula rasa*, without any prior knowledge of the game’s rules.

Acquiring a game-sensitive, pragmatic understanding of the input state is essential to solve both the macroplanning and microplanning challenges involved. Performing inadequately on either of these challenges leads to the trap of generating *common or dull language* [42, 229], that would not satisfy the communicative goal. We find that acquiring such an understanding *tabula rasa* is too challenging for a typical attentional LSTM-based encoder-decoder model in-

stantiating E2EN2PP, leading to the *common response problem* as anticipated, and consequently, inferior performance to even template based baselines on both automatic and human metrics.

Based on our observations, we devise an alternative model that includes an additional *Pragmatic Interpretation Layer* to discretely featurize the board states using a game library, essentially backing off to pragmatic game knowledge to viably declutter the input states, thereby simplifying the understanding and overcoming the microplanning and macroplanning issues observed. Consequently, the model is now able to outperform all baselines, including the template-based one, on both automatic and human metrics.

The devised intervention in the E2EN2PP that needs to be done can be seen in Figure 8.1

Through a human study on predictions for a subset of the data that deals with direct move descriptions, we observe that outputs from our models are rated similar to ground truth commentary texts in terms of correctness and fluency.

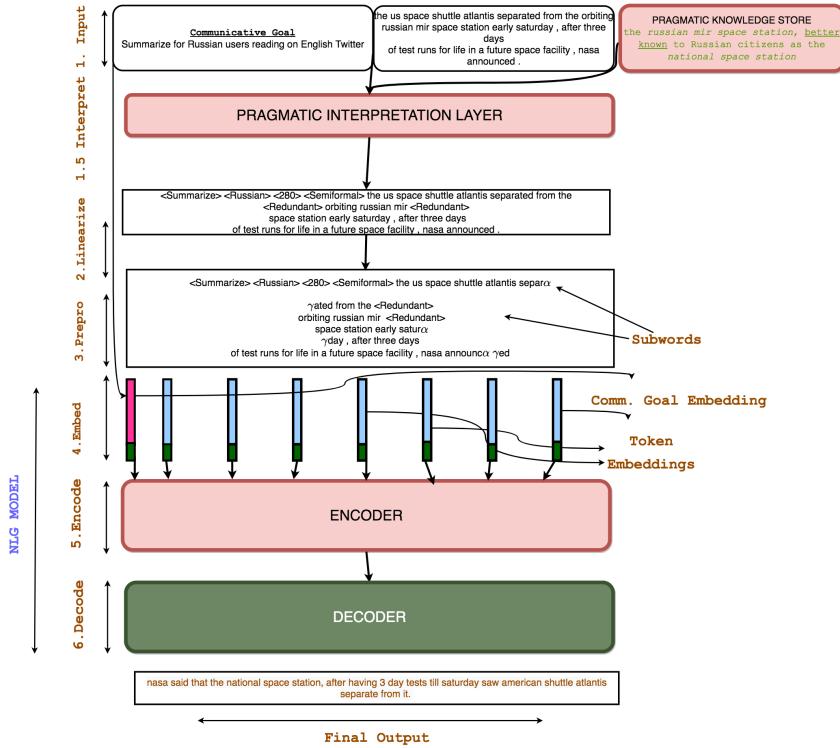


Figure 8.1: An illustration of how *End-to-End Neural NLG Pseudo-Pipeline* would work in action for an actual generation task and input example, after incorporating the Intervention in Chapter 8. Here, the task is to summarize the given input news article to within 280 characters. Note that this is a *Pseudo-Pipeline*, since the layers do not correspond to sub-tasks of NLG; moreover, they cannot be learnt or updated independently. The specific intervention shown here is the introduction of a *Pragmatic Interpretation Layer* that takes in the raw board states and featurizes them into a collection of discrete game-pertinent features.

8.1 Introduction

A variety of work in NLP has sought to produce fluent natural language descriptions conditioned on a contextual grounding. For example, several lines of work explore methods for describing images of scenes and videos [86], while others have conditioned on structured sources like Wikipedia infoboxes [103]. In most cases, progress has been driven by the availability of large training corpora that pair natural language with examples from the grounding [113]. One line of work has investigated methods for producing and interpreting language in the context of a game, a space that has rich pragmatic structure, but where training data has been hard to come by. In this chapter, we introduce a new large-scale resource for learning to correlate natural language with individual moves in the game of chess. We collect a dataset of more than

298K chess move/commentary pairs across $\approx 11K$ chess games from online chess forums. To the best of our knowledge, this is the first such dataset of this scale for a game commentary generation task. We provide an analysis of the dataset and highlight the large variety in commentary texts by categorizing them into six different aspects of the game that they respectively discuss.

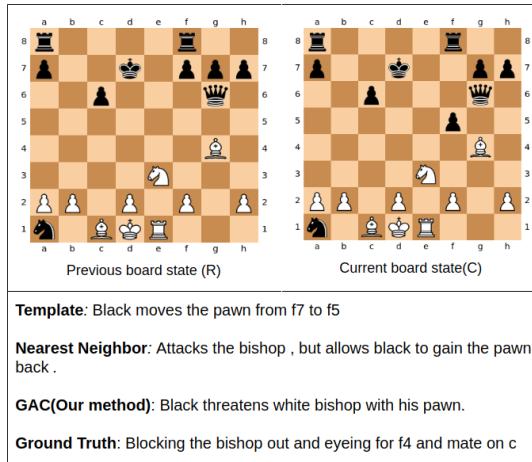


Figure 8.2: Move commentary generated from our method (Game-aware neural commentary generation (GAC)) and some baseline methods for a sample move.

Automated game commentary generation can be a useful learning aid. Novices and experts alike can learn more about the game by hearing explanations of the motivations behind moves, or their quality. In fact, on sites for game aficionados, these commentaries are standard features, speaking to their interestingness and utility as complements to concrete descriptions of the game boards themselves.

Game commentary generation poses a number of interesting challenges for existing approaches to language generation. First, modeling human commentary is challenging because human commentators rely both on their prior knowledge of game rules as well as their knowledge of effective strategy when interpreting and referring to the game state. Secondly, there are multiple aspects of the game state that can be talked about for a given move — the commentator’s choice depends on the pragmatic context of the game. For example, for the move shown in Figure 8.2, one can comment simply that the pawn was moved, or one may comment on how the check was blocked by that move. Both descriptions are true, but the latter is most salient given the player’s goal. However, sometimes, none of the aspects may stand out as being most salient, and the most salient aspect may even change from commentator to commentator. Moreover, a human commentator may introduce variations in the way he or she chooses to talk

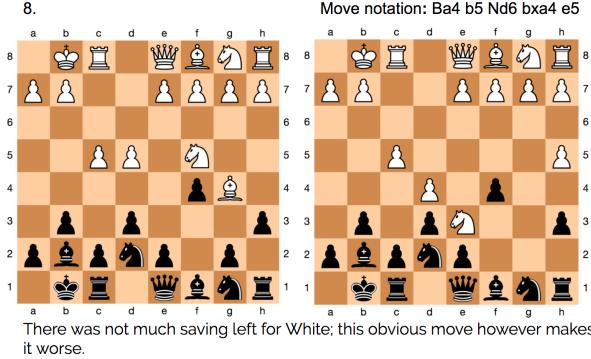


Figure 8.3: A multi-move, single commentary example from our data. Here, the sequence of moves $Ba4 \rightarrow b5 \rightarrow Nd6 \rightarrow bxa4 \rightarrow e5$ is commented upon.

about these aspects, in order to reduce monotony in the commentary. This makes the dataset a useful testbed for the content selection and referring expression sub-skills of NLG.

There has been some, albeit very limited, prior work which has explored game commentary generation. [109, 195] have explored chess commentary generation, but for lack of large-scale training data their methods have been mainly based on rules defined manually. [85] have explored commentary generation for the game of Shogi, proposing a two-step process where salient terms are generated from the game state and then composed in a language model. In contrast, given the larger amount of training data available to us, our devised model uses a trainable neural architecture to predict commentaries given the game state. Our model conditions on semantic and pragmatic information about the current state and explicitly learns to compose, conjoin, and select these features in a recurrent decoder module. We perform an experimental evaluation comparing against baselines and variants of our model that ablate various aspects of our devised architecture. Outputs on the ‘Move Description’ subset of data from our final model were judged by humans to be as good as human written ground truth commentaries on measures of fluency and correctness.

8.2 Chess Commentary Dataset

In this section we introduce our new large-scale *Chess Commentary* dataset, share some statistics about the data, and discuss the variety in type of commentaries. The data is collected from the online chess discussion forum [gameknot . com](http://gameknot.com), that features multiple games self-annotated with move-by-move commentary.

The dataset consists of 298K aligned game move/commentary pairs. Some commentaries

Statistic	Value
Total Games	11,578
Total Moves	298,008
Average no. of recorded steps in a game	25.73
Frequent Word Types ¹	39,424
Rare Word Types	167,321
Word Tokens	6,125,921
Unigram Entropy	6.88
Average Comment Length (in #words)	20.55
Long Comments (#words > 5)	230745 (77%)

Table 8.1: Dataset and Vocabulary Statistics

Category	Example	% in data	Val acc.
Direct Move Description	An attack on the queen	31.4%	71%
Move Quality	A rook blunder.	8.0%	90%
Comparative	At this stage I figured I better move my knight.	3.7%	77.7%
Planning / Rationale	Trying to force a way to eliminate d5 and prevent Bb5.	31.2%	65%
Contextual Game Info	Somehow, the game I should have lost turned around in my favor .	12.6%	87%
General Comment	Protect Calvin , Hobbs	29.9%	78%

Table 8.2: Commentary texts have a large variety making the problem of content selection an important challenge in our dataset. We classify the commentaries into 6 different categories using a classifier trained on some hand-labelled data, a fraction of which is kept for validation. % data refers to the percentage of commentary sentences in the tagged data belonging to the respective category.

are written for a sequence of few moves (Figure 8.3) while others correspond to a single move. For the purpose of initial analysis and modeling, we limit ourselves to only those data points where commentary text corresponds to a single move. Additionally, we split the multi-sentence commentary texts to create multiple data points with the same chess board and move inputs.

What are commentaries about? We observe that there is a large variety in the commentary texts. To analyze this variety, we consider labelling the commentary texts in the data with a predefined set of categories. The choice of these categories is made based on a manual inspection of a sub-sample of data. We consider the following set of commentary categories (Also shown in Table 8.2):

- **Direct move description (MoveDesc²):** Explicitly or implicitly describe the current

²MoveDesc & ‘Move Description’ used interchangeably

move.

- **Quality of move (Quality³):** Describe the quality of the current move.
- **Comparative:** Compare multiple possible moves.
- **Move Rationale or Planning (Planning):** Describe the rationale for the current move, in terms of the future gameplay, advantage over other potential moves etc.
- **Contextual game information:** Describe not the current move alone, but the overall game state — such as possibility of win/loss, overall aggression/defence, etc.
- **General information:** General idioms & advice about chess, information about players/tournament, emotional remarks, retorts, etc.

The examples in Table 8.2 illustrate these classes. Note that the commentary texts are not necessarily limited to one tag, though that is true for most of the data. A total of 1K comments are annotated by two annotators. A SVM classifier [153] is trained for each comment class, considering the annotation as ground truth and using word unigrams as features. This classifier is then used to predict tags for the train, validation and test sets. For “Comparative” category, we found that a classifier with manually defined rules such as presence of word “better” performs better than the classifier, perhaps due to the paucity of data, and thus we use this instead . As can be observed in Table 8.2, the classifiers used are able to generalize well on the held out dataset.

8.3 Game Aware Neural Commentary Generations (GAC)

Our dataset D consists of data points of the form $(S_i, M_i, G_i), i \in \{1, 2, \dots, |D|\}$, where S_i is the commentary text for move M_i and G_i is the corresponding chess game. S_i is a sequence of m tokens $S_{i1}, S_{i2}, \dots, S_{im}$. We want to model $P(S_i|M_i, G_i)$. For simplicity, we use only current board (C_i) and previous board (R_i) information from the game. $P(S_i|M_i, G_i) = P(S_i|M_i, C_i, R_i)$.

We model this using an end-to-end trainable neural model, that models conjunctions of features using feature encoders. Our model employs a selection mechanism to select the salient features for a given chess move. Finally a LSTM recurrent neural network [76] is used to generate the commentary text based on selected features from encoder.

³Quality and ‘Move Quality’ used interchangeably

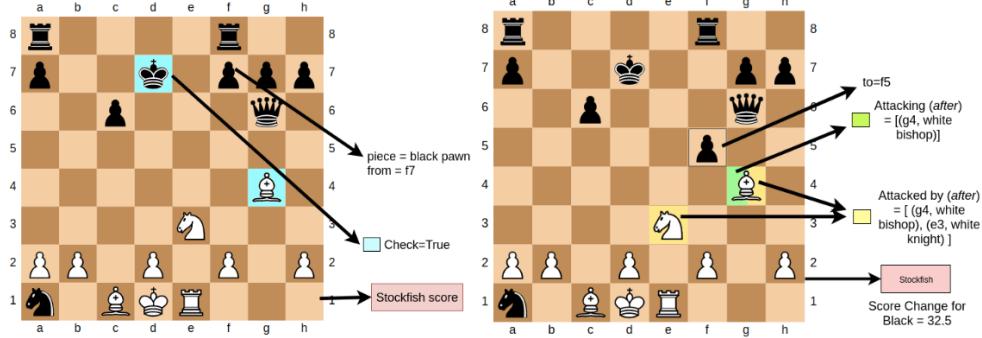


Figure 8.4: The figure shows some features extracted using the chess board states before (*left*) and after (*right*) a chess move. Our method uses various semantic and pragmatic features of the move, including the location and type of piece being moved, which opposing team pieces attack the piece being moved before as well as after the move, the change in score by *Stockfish* UCI engine, etc.

8.3.1 Incorporating Domain Knowledge

Past work shows that acquiring domain knowledge is critical for NLG systems [124, 186], particularly data-to-text NLG systems where the data is . Commentary texts cover a range of perspectives, including criticism or goodness of current move, possible alternate moves, quality of alternate moves, etc. To be able to make such comments, the model must learn about the quality of moves, as well as the set of valid moves for a given chess board state. We consider the following features to provide our model with necessary information to generate commentary texts (Figure 8.4):

Move features $f_{move}(M_i, C_i, R_i)$ encode the current move information such as which piece moved, the position of the moved piece before and after the move was made, the type and position of the captured piece (if any), whether the current move is castling or not, and whether there was a check or not.

Threat features $f_{threat}(M_i, C_i, R_i)$ encode information about pieces of opposite player attacking the moved piece before and after the move, and the pieces of opposite player being attacked by the piece being moved. To extract this information, we use the *python-chess*⁴ library

Score features $f_{score}(M_i, C_i, R_i)$ capture the quality of move and general progress of the game. This is done using the game evaluation score before and after the move, and average rank of pawns of both the players. We use *Stockfish* evaluation engine to obtain the game

⁴<https://pypi.org/project/python-chess/>

evaluation scores.⁵

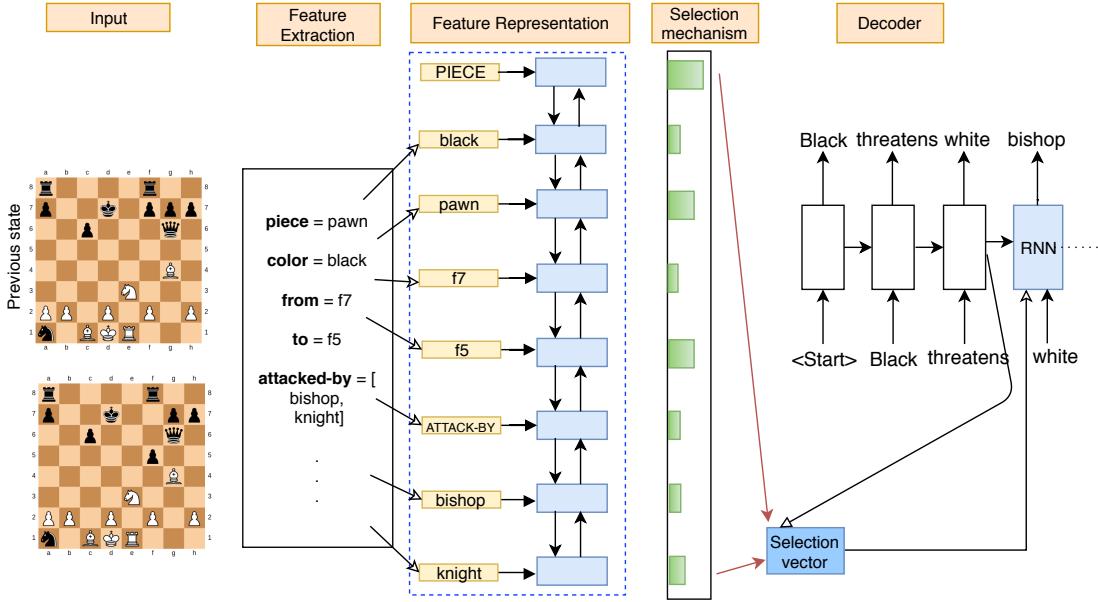


Figure 8.5: The figure shows a model overview. We first extract various semantic and pragmatic features from the previous and current chess board states. We represent features through embedding in a shared space. We observe that feeding in feature conjunctions helps a lot. We consider a selection mechanism for the model to choose salient attributes from the input at every decoder step.

8.3.2 Feature Representation

In our simplest conditioned language generation model **GAC-sparse**, we represent the above described features using sparse representations through binary-valued features. $g_{sparse}(M_i, C_i, R_i) = \text{SparseRep}(f_{move}, f_{threat}, f_{score})$

For our full **GAC** model we consider representing features through embeddings. This has the advantage of allowing for a shared embedding space, which is pertinent for our problem since attribute values can be shared, e.g., the same piece type can occur as the moved piece as well as the captured piece. For categorical features, such as those indicating which piece was moved, we directly look up the embedding using corresponding token. For real valued features such as game scores, we first bin them and then use corresponding number for embedding lookup. Let E represent the embedding matrix. Then $E[f_{move}^j]$ represents embeddings of j^{th} move feature, or in general $E[f_{move}]$ represents the concatenated embeddings of all move features. Similarly, $E(f_{move}, f_{threat}, f_{score})$ represents concatenated embeddings of all the features.

⁵<https://stockfishchess.org/about/>

8.3.3 Feature Conjunctions

We conjecture that explicitly modeling feature conjunctions might improve the performance. So we need an encoder that can handle input sets of features of variable length (features such as pieces attacking the moved piece can be of variable length). One way to handle this is by picking up a canonical ordering of the features and consider a bidirectional LSTM encoder over the feature embeddings. As shown in Figure 8.5, this generates conjunctions of features.

$$g^{enc} = \text{BiLSTM}^*(\{E(f_{move}, f_{threat}, f_{score})\})$$

Here $E()$ represents the embedding matrix as described earlier and BiLSTM^* represents a sequential application of the BiLSTM function. Thus, if there are a total of m feature keys and embedding dimension is d , $E(f_{move}, f_{threat}, f_{score})$ is a matrix of $m * d$. If hidden size of BiLSTM is of size x , then g^{enc} is of dimensionality $m * x$. We observe that different orderings gave similar performance. We also experimented with running k encoders, each on different ordering of features, and then letting the decoder access to each of the k encodings. This did not yield any significant gain in performance.

The GAC model, unlike GAC-sparse, has some advantages as it uses a shared, continuous space to embed attribute values of different features, and can perform arbitrary feature conjunctions before passing a representation to the decoder, thereby sharing the burden of learning the necessary feature conjunctions. Our experiments confirm this intuition — GAC produces commentaries with higher BLEU as well as more diversity compared to GAC-sparse.

8.3.4 Decoder

We use a LSTM decoder to generate the sentence given the chess move and the features g . At every output step t , the LSTM decoder predicts a distribution over vocabulary words taking into account the current hidden state h_t , the input token i_t , and additional selection vector c_t . For GAC-sparse, the selection vector is simply an affine transformation of the features g . For GAC model selection vector is derived via a selection mechanism.

$$\begin{aligned} o_t, h_t^{dec} &= \text{LSTM}(h_{t-1}^{dec}, [\text{concat}(E_{dec}(i_t), c_t)]) \\ p_t &= \text{softmax}(W_o[\text{concat}(o_t, c_t)] + b_s) \end{aligned}$$

where p_t represents the probability distribution over the vocabulary, $E_{dec}()$ represents the decoder word embedding matrix and elements of W_o matrix are trainable parameters.

Selection/Attention Mechanism: As there are different salient attributes across the dif-

ferent chess moves, we also equip the GAC model with a mechanism to select and identify these attributes. We first transform h_t^{dec} by multiplying it with a trainable matrix W_c , and then take dot product of the result with each g_i .

$$\begin{aligned} a_t^{(i)} &= \text{dot}(W_c * h_t^{dec}, g_i^{enc}) \\ \alpha_t &= \text{softmax}(a_t) \\ c_t &= \sum_{i=1}^{|g|} \alpha_t^{(i)} g_i^{enc} \end{aligned}$$

We use cross-entropy loss over the decoding outputs to train the model.

8.4 Experiments

We split each of the data subsets in a 70:10:20 ratio into train, validation and test. All our models are implemented in Pytorch version 0.3.1 [152]. We use the ADAM optimizer [93] with its default parameters and a mini-batch size of 32. Validation set perplexity is used for early-stopping. At test-time, we use greedy search to generate the model output. We observed that beam decoding does not lead to any significant improvement in terms of validation BLEU score.

We observe the BLEU [151] and BLEU-2 [228] scores to measure the performance of the models. Additionally, we consider a measure to quantify the diversity in the generated outputs. Finally, we also conduct a human evaluation study. In the remainder of this section, we discuss baselines along with various experiments and results.

8.4.1 Baselines

In this subsection we discuss the various baseline methods.

Manually-defined template (TEMP) We devise manually defined templates [181] for ‘Move Description’ and ‘Move Quality’ categories. Note that template-based outputs tend to be repetitive as they lack diversity – drawing from a small, fixed vocabulary and using a largely static sentence structure. We define templates for a fixed set of cases which cover our data. (For exact template specifications, refer to Appendix B)

Nearest Neighbor (NN): We observe that the same move on similar board states often leads to similar commentary texts. To construct a simple baseline, we find the most similar move

N_{MCR} from among training data points for a given previous (R) and current (C) board states and move M . The commentary text corresponding to N_{MCR} is selected as the output. Thus, we need to consider a scoring function to find the closest matching data point in training set. We use the *Move*, *Threat* and *Score* features to compute similarity to do so. By using a sparse representation, we consider total of 148 *Move* features, 18 *Threat* features, and 19 *Score* features. We use sklearn’s [154] NearestNeighbor module to find the closest matching game move.

Raw Board Information Only (RAW): The RAW baseline ablates to assess the importance of our pragmatic feature functions. This architecture is similar to GAC, except that instead of our custom features $A(f(R_i, C_i))$, the encoder encodes raw board information of current and previous board states.

$$A_{RAW}(R_i, C_i) = [Lin(R_i), Lin(C_i)]$$

$Lin()$ for a board denotes it’s representation in a row-linear fashion. Each element of $Lin()$ is a piece name (e.g., *pawn*) denoting the piece at that square with special symbols for empty squares.

8.4.2 Comment Category Models

As shown earlier, we categorize comments into six different categories. Among these, in this chapter we consider only the first three as the amount of variance in the last three categories indicates that it would be extremely difficult for a model to learn to reproduce them accurately. The number of data points, as tagged by the trained classifiers, in the subsets ‘Move Description’, ‘Move Quality’ and ‘Comparative’ are 28,228, 793 and 5397 respectively. We consider separate commentary generation models for each of the three categories. Each model is tuned separately on the corresponding validation sets. Table 8.3 shows the BLEU and BLEU-2 scores for the devised model under different subsets of features. Overall BLEU scores are low, likely due to the inherent variance in NLG tasks such as dialog response generation and data-to-text description (of which our task is an example) generation tasks, where even adequate outputs sometimes do not match references due to many possible outputs being adequate for the same input [146]. A precursory examination of the outputs for data points selected randomly from the test set indicated that they were reasonable. Figure 8.6 illustrates commentaries generated by our models through an example (a larger list of qualitative examples can be found in Appendix C).

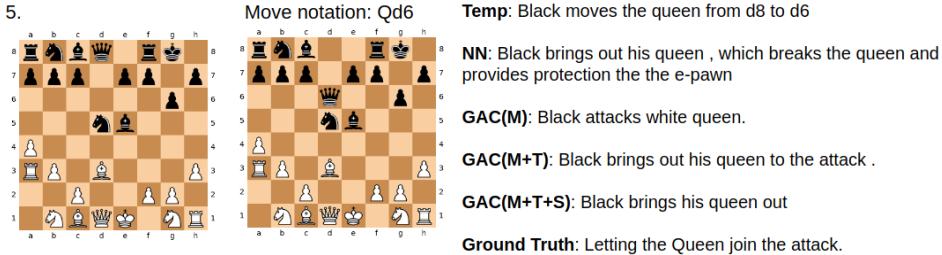


Figure 8.6: Outputs from various models on a test example from the **MoveDesc** subset.

Which features are useful? In general, adding *Threat* features improves the performance, though the same is not always true for *Score* features. One reason for the importance of *Threat* features might be that the change in value of a *Threat* feature often includes comment-worthy events such as the king coming under check, or going out of check. Learning to detect the activation and de-activation of a situation where one piece threatens another is challenging in the absence of *Threat* features, since the piece being threatened and the piece which is threatening may even both not be identical to the piece which was immediately moved. Making such inferences would require the model to not just memorize all the piece positions but also, after each move, to implicitly compute whether a threat is activated by iterating over all opposing piece pairs.

The changes in *Score* features may often correspond to subtle changes later in the game tree, and are perhaps unlikely to immediately trigger the use of specific phrases in resultant commentary. Furthermore, for lot of cases where they do trigger use of specific phrases, this might be due to activation or de-activation of threats, or some specific specialities of the move (e.g., Castling) which is already explicitly captured by the *Move* and *Threat* features. This might explain the observation that addition of *Score* features does not always improve performance.

Qual has higher BLEU scores than the other datasets due to smaller vocabulary and lesser variation in commentary. As can be observed in Table 8.4, *Threat* features are useful for both ‘Move Quality’ and ‘Move Description’ subsets of data. Adding *Score* features helps for ‘Move Quality’ subset. This intuitively makes sense since *Score* features directly encode proxies for move quality as per a chess evaluation engine.

8.4.3 A Single Model For All Categories

In this experiment, we merge the training and validation data of the first three categories and tune a single model for this merged data. We then compare its performance on all test sentences in our data. **COMB** denotes using the best GAC model for a test example based on its original

Dataset	Features	BLEU	BLEU-2	Diversity
MoveDesc	TEMP	0.72	20.77	4.43
	NN (M+T+S)	1.28	21.07	7.85
	RAW	1.13	13.74	2.37
	GAC-sparse	1.76	21.49	4.29
	GAC (M+T)	1.85	23.35	4.72
Quality	TEMP	16.17	47.29	1.16
	NN (M+T)	5.98	42.97	4.52
	RAW	16.92	47.72	1.07
	GAC-sparse	14.98	51.46	2.63
	GAC(M+T+S)	16.94	47.65	1.01
Comparative	NN (M)	1.28	24.49	6.97
	RAW	2.80	23.26	3.03
	GAC-sparse	3.58	25.28	2.18
	GAC(M+T)	3.51	29.48	3.64

Table 8.3: Performance of baselines and our model with different subsets of features as per various quantitative measures.

(**S** = Score, **M**= Move, **T** = Threat features;) On all data subsets, our model outperforms the **TEMP** and **NN** baselines. Among devised models, GAC performs better than GAC-sparse & RAW in general. For NN, GAC-sparse and GAC methods, we experiment with multiple feature combinations and report only the best as per BLEU scores.

class (e.g., *Desc*) and computing the BLEU of the sentences so generated with the ground truth. **GAC-all** represents the GAC model learnt on the merged training data.

As can be seen from Table 8.5, this does not lead to any performance improvements. We investigate this issue further by analyzing whether the board states are predictive of the type of category or not. To achieve this, we construct a multi-class classifier using all the *Move*, *Threat* and *Score* features to predict the three categories under consideration. However, we observe accuracy of around 33.4%, which is very close to the performance of a random prediction model. This partially explains why a single model did not fare better even though it had the opportunity to learn from a larger dataset.

Category-aware model (CAT) We observed above that with the considered features, it is not possible to predict the type of comment to be made, and the GAC-all model results are better than COMB results. Hence, we extend the GAC-all model to explicitly provide with the information about the comment category. We achieve this by adding a one-hot representation of the category of the comment to the input of the RNN decoder at every time step. As can be seen in the Table 8.5, CAT(M) performs better than GAC-all(M) in terms of BLEU-4, while performing slightly worse on BLEU-2. This demonstrates that explicitly providing information

Dataset	Features	BLEU	BLEU-2	Diversity
MoveDesc	GAC (M)	1.41	19.06	4.32
	GAC (M+T)	1.85	23.35	4.72
	GAC (M+T+S)	1.64	22.82	4.29
Quality	GAC (M)	13.05	48.37	1.61
	GAC (M+T)	14.22	49.57	1.54
	GAC(M+T+S)	14.44	51.79	1.48
Comparative	GAC(M)	3.10	19.84	2.88
	GAC(M+T)	3.51	29.48	3.64
	GAC(M+T+S)	1.15	25.44	3.14

Table 8.4: Performance of the GAC model with different feature sets. (**S** = Score, **M**= Move, **T** = Threat features;) Different subset of features work best for different subsets. For instance, *Score* features seem to help only in the Quality category. Note that the results for Quality are from 5-fold cross-validation, since the number of datapoints in the category is much lesser than the other two.

Dataset	Features	BLEU	BLEU-2	Diversity
All	COMB (M)	2.07	20.13	4.50
	COMB (M+T)	2.43	25.37	4.88
	COMB (M+T+S)	1.83	28.86	4.33
All	GAC-all(M)	1.69	20.66	4.67
	GAC-all(M+T)	1.94	24.11	5.16
	GAC-all (M+T+S)	2.02	24.70	4.97
All	CAT (M)	1.90	19.96	3.82

Table 8.5: The **COMB** approaches show the combined performance of separately trained models on the respective test subsets.

about the comment category can help the model.

8.4.4 Diversity In Generated Commentaries

Humans use some variety in the choice of words and sentence structure. As such, outputs from rule based templates, which demonstrate low variety, may seem repetitive and boring. To capture this quantitatively, and to demonstrate the variety in texts from our method, we calculate the entropy [211] of the distribution of unigrams, bigrams and trigrams of words in the predicted outputs, and report the geometric mean of these values. Using only a small set of words in similar counts will lead to lower entropy and is undesirable. As can be observed from Table 8.3, template baseline performs worse on the said measure compared to our methods for the 'MoveDesc' subset of the data.

Question	GT	GAC (M)	GAC (MT)	GAC (MTS)	GAC -sparse	TEMP	NN
Is commentary correct for the given move? (%Yes)	70.4	42.3	64.8	67.6	56.3	91.5	52.1
Can the move be inferred from the commentary? (%Yes)	45.1	25.3	42.3	36.7	40.8	92.9	42.3
Fluency (scale of (least)1 - 5(most))	4.03	4.15	4.44	4.54	4.15	4.69	3.72
Mean (Std. dev.)	(1.31)	(1.20)	(1.02)	(0.89)	(1.26)	(0.64)	(1.36)

Table 8.6: Human study results on *MoveDesc* data category. Outputs from GAC are in general better than ground truth, NN and GAC-sparse. TEMP outperforms other methods, though as shown earlier, outputs from TEMP lack diversity.

8.4.5 Human Evaluation Study

As discussed in the qualitative examples above, we often found the outputs to be good - though BLEU scores are low. BLEU is known to correlate poorly [146, 182, 242] with human relevance scores for NLG tasks. Hence, we conduct a human evaluation study for the best 2 neural (GAC,GAC-sparse) and best 2 non-neural methods (TEMP,NN).

Setup: Specifically, annotators are shown a chess move through previous board and resulting board snapshots, along with information on which piece moved (a snapshot of a HIT⁶ is provided in the Appendix D). With this context, they were shown text commentary based on this move and were asked to judge the commentary via three questions, shortened versions of which can be seen in the first column of Table 8.6.

We randomly select 100 data points from the test split of ‘Move Description’ category and collect the predictions from each of the methods under consideration. We hired two Anglo-phone (Lifetime HIT acceptance % > 80) annotators for every human-evaluated test example. We additionally assess chess proficiency of the annotators using questions from the chess-QA dataset by [32]. Within each HIT, we ask two randomly selected questions from the chess-QA dataset. Finally we consider only those HITs wherein the annotator was able to answer the proficiency questions correctly.

Results: We conducted a human evaluation study for the *MoveDesc* subset of the data. As can be observed from Table 8.6, outputs from our method attain slightly more favorable scores compared to the ground truth commentaries. This shows that the predicted outputs from our model are not worse than ground truth on the said measures. This is in spite of the fact that

⁶Human Intelligence Task

the BLEU-4 score for the predicted outputs is only ~ 2 w.r.t. the ground truth outputs. One reason for slightly lower performance of the ground truth outputs on the said measures is that some of the human written commentaries are either very ungrammatical or too concise. A more surprising observation is that around 30% of human written ground truth outputs were also marked as not valid for given board move. On inspection, it seems that commentary often contains extraneous game information beyond that of move alone, which indicates that an ideal comparison should be over commentary for an entire game, although this is beyond the scope of the current work.

The inter-annotator agreement for our experiments (Cohen’s κ [34]) is 0.45 for Q1 and 0.32 for Q2. We notice some variation in κ coefficients across different systems. While TEMP and GAC responses had a 0.5-0.7 coefficient range, the responses for CLM had a much lower coefficient. In our setup, each HIT consists of 7 comments, one from each system. For Q3 (fluency), which is on an ordinal scale, we measure rank-order consistency between the responses of the two annotators of a HIT. Mean Kendall τ [88] across all HITs was found to be 0.39.

To measure statistical significance of results, we perform bootstrap tests on 1000 subsets of size 50 with a significance threshold of $p = 0.05$ for each pair of systems. For Q1, we observe that GAC(M), GAC(M+T) and GAC(M+T+S) methods are significantly better than baselines NN and GAC-sparse. We find that neither of GAC(M+T) and GT significantly outperform each other on Q1 as well as Q2. But we do find that GAC(M+T) does better than GAC(M) on both Q1 and Q2. For fluency scores, we find that GAC(M+T) is more fluent than GT, NN, GAC-sparse, GAC(M). Neither of GAC(M) and GAC(M+T+S) is significantly more fluent than the other.

8.5 Related Work

Data-to-text NLG research has a long and rich history, with systems ranging from completely rule-based [33] to learning-based ones [29, 185, 187], which have had both practical successes [187] and failures [185]. Recently, there have been numerous works that propose text generation given input information such as structured records, biographies [103], recipes [91, 247], etc. A key difference between generation given a game state compared to these inputs is that the game state is an evolving description at a point in a process, as opposed to recipes (that are independent of each other), records (which are static) and biographies (which are one per person, and again independent). Moreover, our devised method effectively uses various types of semantic and pragmatic information about the game state.

In this chapter, we have introduced a new large-scale data for game commentary genera-

tion. The commentaries cover a variety of aspects like move description, quality of move, and alternative moves. This leads to a content selection challenge, similar to that noted in [242]. Unlike [242], our focus is on generating commentary for individual moves in a game, as opposed to game summaries from aggregate statistics as in their task.

One of the first NLG datasets was the SUMTIME-METEO [187] corpus with ≈ 500 record-text pairs for technical weather forecast generation. Liang et al [108] worked on common weather forecast generation using the WEATHERGOV dataset, that has $\approx 10K$ record-text pairs. A criticism of WEATHERGOV is that weather records themselves may have used templates and rules with optional human post-editing. There have been prior works on generating commentary for ROBOCUP matches [29, 134]. The ROBOCUP dataset, however, is collected from 4 games and contains about 1K events in total. Our dataset is two orders of magnitude larger than the ROBOCUP dataset, and we hope that it provides a promising setting for future NLG research.

8.6 Conclusions

In this chapter, we curate a dataset for the task of chess commentary generation and devise methods to perform generation on this dataset. We analyze our task and initial baseline model’s performance in terms of the challenges it poses along CNLP. We find that our initial baseline model, that is a typical attentional LSTM-based encoder-decoder framework instantiating E2EN2PP, is found to lapse into merely generating input-dependent common responses, underperforming template-based baselines. The model responses rarely choose the pertinent and interesting pieces as well as inter-piece relationships to talk about given the current move. They even fail to simply describe the piece that moved and its initial and final locations, as evinced by their underperforming even the template-based baseline which follows that simple strategy. This indicates their deficient performance on the content selection subskill. Furthermore, even when the responses choose the adequate piece (s) movements, and inter-piece relationships to describe, they seldom employs rich expressions such as *joining the attack*, *develops his position*, *putting in check* etc., indicating the model’s deficiency at referring expression generation

We posited that the aforementioned deficiencies were due to the model’s inability to understand the game states in the larger pragmatic context of the game, and devise a method that directly provides knowledge about game pragmatics to its decoder via backing off to a game-library based discrete featurization introduced as an *Intervention* in the form of a *Pragmatic Interpretation Layer* overcomes these challenges, resulting in a viable commentary generator

outperforming all the baselines, including the template-based one. The devised Intervention to E2EN2PP, that causes a departure from the end-to-end nature, is illustrated in Figure 8.1.

Our devised method effectively utilizes information related to the rules and pragmatics of the game. A human evaluation study judges outputs from the devised methods to be as good as human written commentary texts for the ‘Move Description’ subset of the data.

Subsequent work [215] has proposed reinforcement learning based game-playing agents that learn to play board games from scratch, learning end-to-end from both recorded games and self-play. An interesting point to explore is whether such pragmatically trained game state representations can be leveraged for the task of game commentary generation.

8.6.1 Broader Takeaways

Whenever there is a sharp disparity between the granularity at which the input (or the input portion of the communicative goal) states information and the granularity at which the output is expected to operate, one expects a gap between the understanding module and the generation module’s representations, and the generation module has the added burden of learning to map the more disparate than usual understanding representation to its own representation.

If filling this gap requires extensive acquisition of commonsense or other forms of knowledge (such as knowledge which can only be acquired through gameplay as in our case), it is possible that the above mappings may be deficient. In such a case, to perform macroplanning tasks such as content selection viably (which may otherwise be obstructed), it becomes necessary to devise some way of converting the input to the right granularity by incorporating this knowledge either completely or through some heuristics, thus bridging the granularity gap.

Consider the example of generating game summary commentary for sports such as soccer and American football, given only the various summary scores and highlights tables generated in aggregate at the end of the game. These tables may contain a wide, heterogenous range of information, with only limited lexical interpretation offered by the row and column names, which themselves are domain-specific terms such as *Home/Away*, *Possession*, including even acronyms such as *GA* (Goals Attempted) etc. Different parts of this information may become pertinent for different games. For example, *Away* games are generally harder for visiting teams, and even a closely fought draw or loss may entail not entirely negative commentary towards the visiting team e.g, *In a hard fought game, ... , In a close encounter in harsh conditions,* A NLG model which attempts to learn the game summary commentary task may have a hard time learning to select, tabula rasa, the right content by decluttering and combining the appropriate

row and column tuples. However, if one were to instead introduce a Interpretation layer, like in this chapter, which first does either or both of the below pre-processing steps would greatly ease the model’s ability to learn to content select and refer to game information appropriately.

1. Create a large number of game-pertinent, discrete 0-1 or multi-valued features, which it extracts from the game tables. e.g., a 0-1 feature can be Away games lost with a margin of only 2 goals
2. Perform preliminary lexicalization which uses simple rules to convert the table cells into simple lexical statements e.g., *Team X had a formation of 4 defenders, 4 midfielders and 2 strikers*. These statements can be included as additional parts of the input.

Appendix A: Additional Data Examples

Appendix B: Additional details for methods

Templates

- *Move Description*: For the Move Description category, we consider following templates:
 1. **Capture** moves : [PLAYERMOVED] captures the [CAPTUREDPIECE] at [FINALsquare] using the [PIECEMOVED] at [INITIALsquare].
 2. **Non-Capture** moves: [PLAYERMOVED] moves the [PIECEMOVED] from [INITIALsquare] to [FINALsquare].
 3. **Castling** moves: [PLAYERMOVED] does a castling.

For moves which lead to a CHECK in the resultant board state, an additional *putting the king in check* is added to the template. [PLAYERMOVED] (Black/White), [INITIALsquare], [FINALsquare], [PIECEMOVED] are filled in based on the move description on the input side.

- *Move Quality*: Based on the move score (as calculated by the chess engine *Stockfish*) $> \theta$ or $< \theta$, one of the following two is generated:

1. A good move.
2. A bad move. The threshold θ is found by tuning it on the validation set to maximize BLEU. We start from $\theta = 0$.

Text	Categories
Unpins and defends the knight , but it does n't matter , as the time is ripe .	Desc
He gets fed up and exchanges Queen for Rook .	Desc
Rxc3 , I just retake with my queen , whilst if he attempts defense with the bishop , then after 17.Bd2 , Ne4 , 18.Rxc3 , Nxg3 , 19.Rxc6 , Nxh1 , I 've won a rook outright .	Desc,Rationale
Preparing to castle , and threatening now white 's e pawn for real.	Desc
Simply getting my rook off that dangerous diagonal , and protecting the b pawn .	Desc
I throw in a check	Desc
Threatening mate with Qxh2	Desc,Quality
A punch drunk move !	Quality
This is not the best move.	Quality
The most logical move.	Quality
This move is dubious.	Quality
The check gains time to support the advance of the a-paw maybe Ke1 was better	Desc,Quality
I did n't want to retreat the N and I rejected 11 .	Rationale
I wish to both defend the pawn , and threaten indirectly the black queen , gaining a tempo	Rationale
it would suite me better if my opponent made a queenside castling , since then my advanced pawn on the d-file would assist in a future attack on the king 's position .	Comparative
but better would be nd2 to get the knight in the game , the queen rook , too .	Comparitive
i think it would have been better to play nxе5 and maintain a material advantage .	Comparitive
although not as effective as the bishop move , even 10.0-0-0 is better than the text , though 10 ... bg4 would have been very nasty .	Comparitive
fianchettoing , so that when black does complete his development , his b will be on a better diagnol .	Comparitive
He doesn't notice that his Knight is hanging ...	GameInfo
Now of course my forces are anchored around the pawns on e3 and h5 , and the black rook loses his hope of penetrating the white position on the e-file	GameInfo
Well, now the game will get interesting soon	GeneralInfo
He tries his trick , which of course is noticed	GeneralInfo
This is often what I will do , when I 'm playing white.	GeneralInfo

Table 8.7: Some commentary texts from each of the six categories. The **Categories** column lists those into which the example falls. As pointed out earlier, the category labels are not exclusive i.e., a text can belong to multiple categories, though texts with more than one category are few in our dataset. ('Desc' is short for 'Move Description')

Appendix C: Qualitative examples

Some qualitative examples.

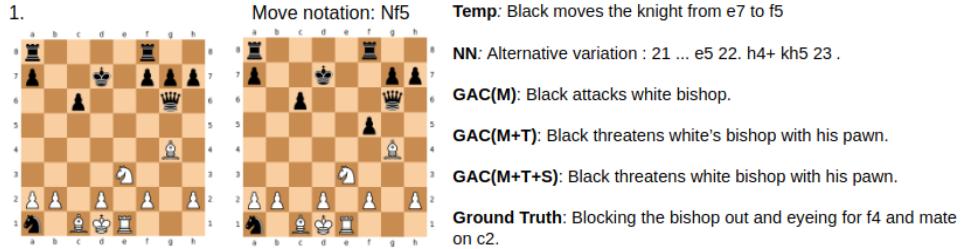


Figure 8.7: Example output 1: Move description subset of data.

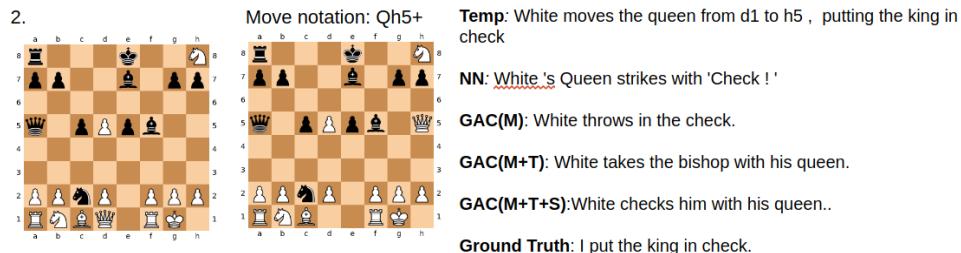


Figure 8.8: Example output 2: Move description subset of data.

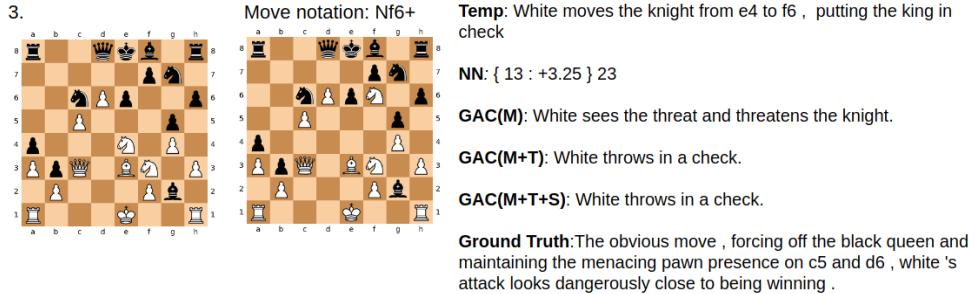


Figure 8.9: Example output 3: Move description subset of data.

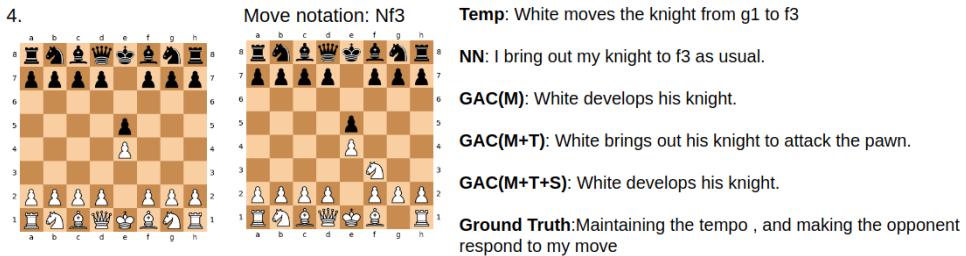


Figure 8.10: Example output 4: Move description subset of data.

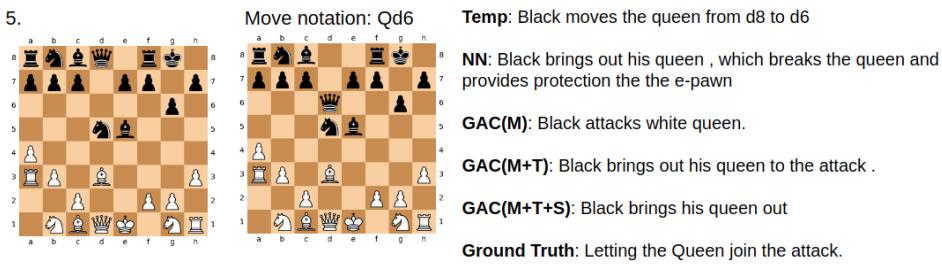


Figure 8.11: Example output 5: Move description subset of data.

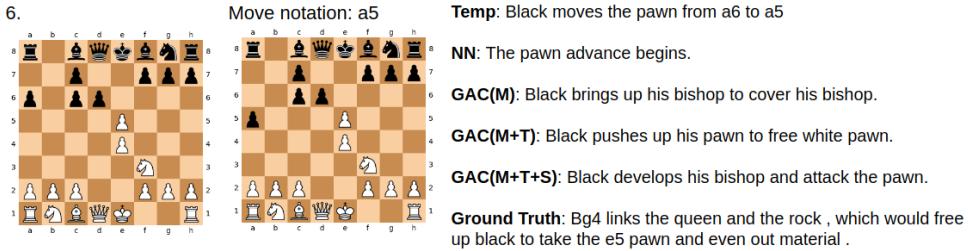


Figure 8.12: Example output 6: Move description subset of data.

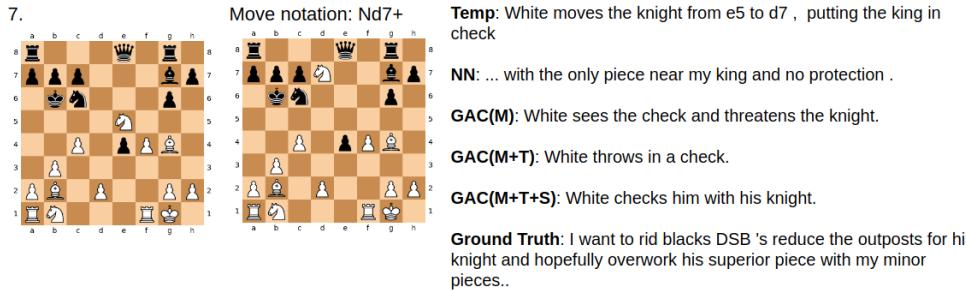


Figure 8.13: Example output 7: Move description subset of data.

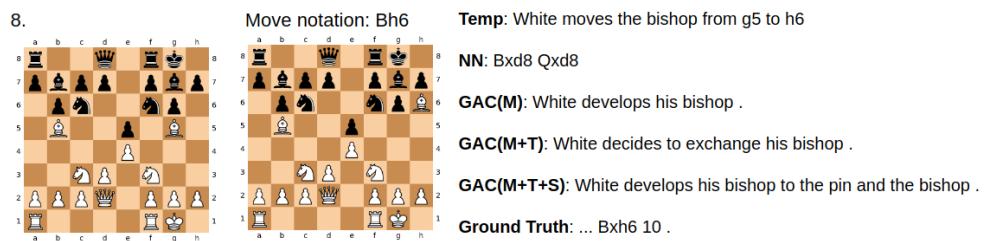


Figure 8.14: Example output 8: Move description subset of data.



Figure 8.15: Example output 1: Move quality subset of data.

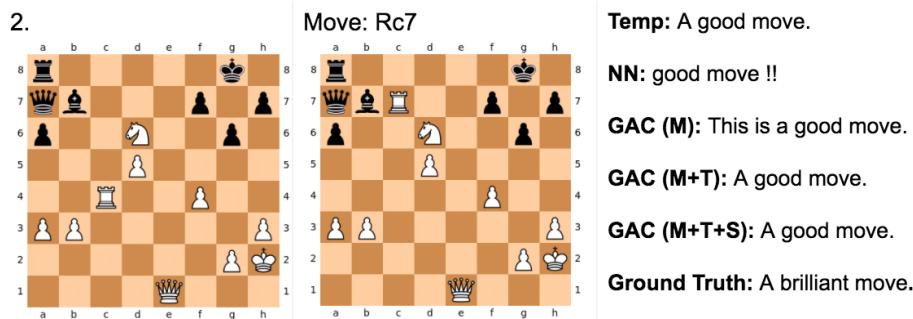


Figure 8.16: Example output 2: Move quality subset of data.

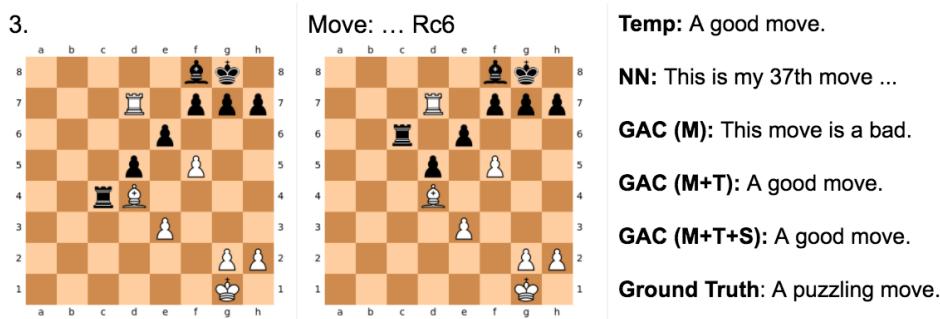


Figure 8.17: Example output 3: Move quality subset of data.

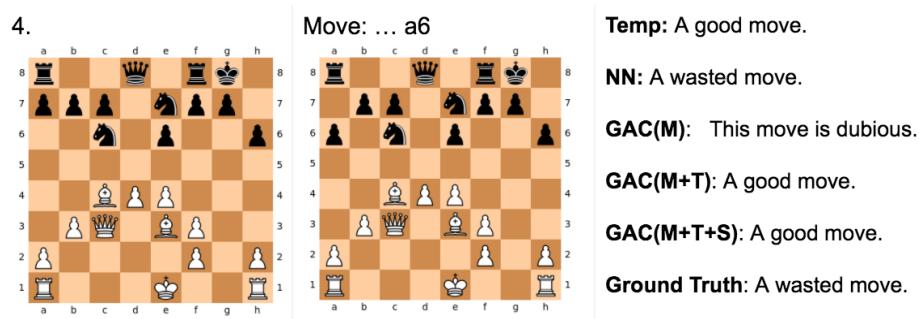


Figure 8.18: Example output 4: Move quality subset of data.

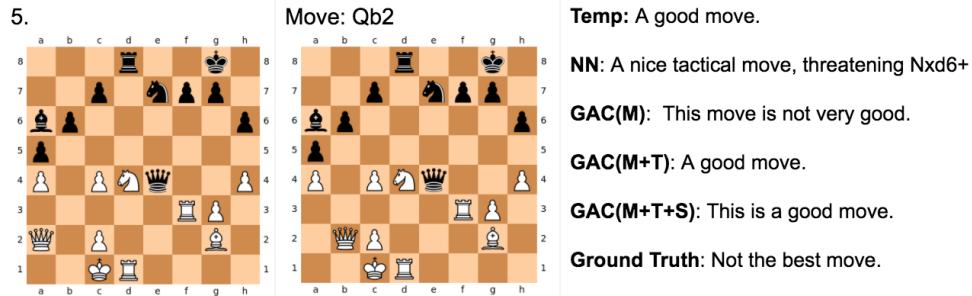


Figure 8.19: Example output 5: Move quality subset of data.

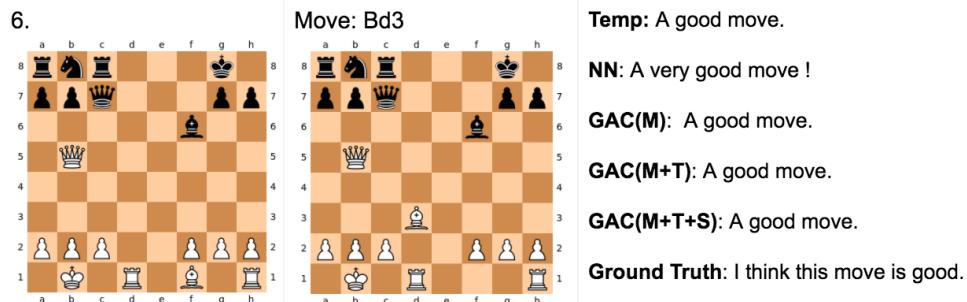


Figure 8.20: Example output 6: Move quality subset of data.

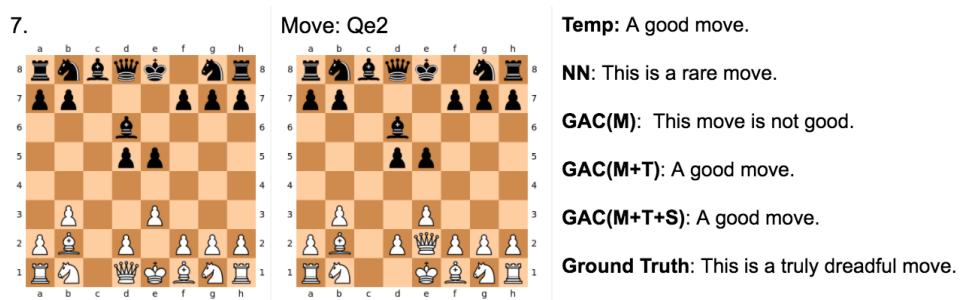


Figure 8.21: Example output 7: Move quality subset of data.



Figure 8.22: Example output 1: Comparative subset of data.

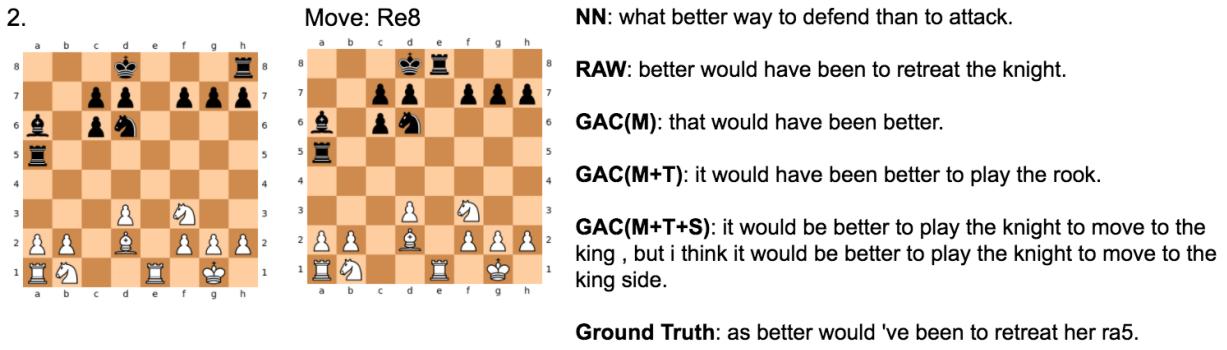


Figure 8.23: Example output 2: Comparative subset of data.

Appendix D: Additional information on AMT experiment

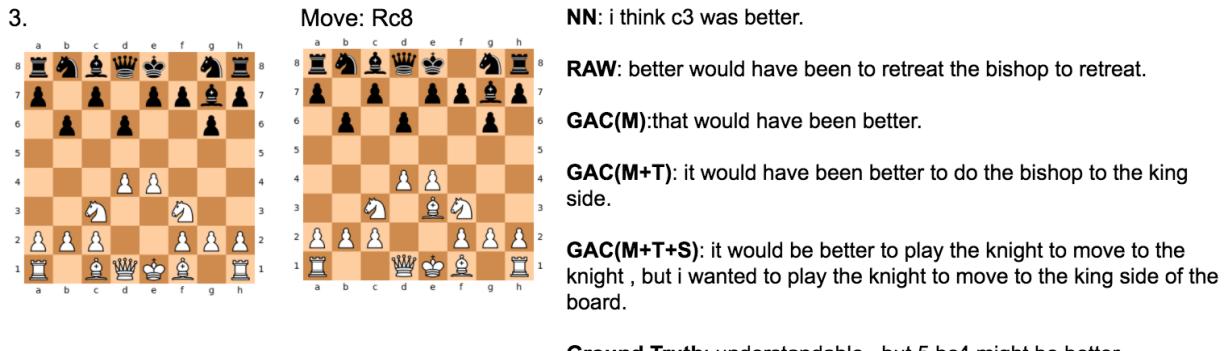


Figure 8.24: Example output 3: Comparative subset of data.

Instructions (Click to collapse)

This task requires basic knowledge of the game of chess. Please participate only if you have decent knowledge about chess.

Our first two questions check some basic knowledge of chess game.

Thereafter, for the remaining questions, you will be shown a chess move through the previous board and the resulting board, along with information on which piece moved. With this context, you will be shown a text commentary on the game. You have judge the commentary on :

- 1) Correctness: Is the text commentary a valid commentary for the chess board
- 2) Completeness: Does the commentary correctly describe the chess move which occurred. In other words, given the commentary and the *previous* board, would you be able to figure out the move which was taken?
- 3) English language Fluency: Is the commentary in fluent English?

Proficiency question 1.

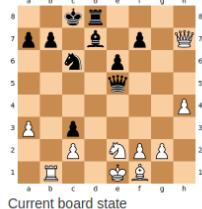


Current board state

Q. Is the game over with a checkmate?

Yes No

Proficiency question 2.



Current board state

Q. Does black knight attacks d4?

Yes No

Figure 8.25: AMT (Amazon Mechanical Turk) sample HIT (Human Intelligence Task): Part 1 of 2 : Two chess proficiency questions are asked at beginning of a HIT

Proficiency question 2.



Q. Does black knight attacks d4?

Yes No

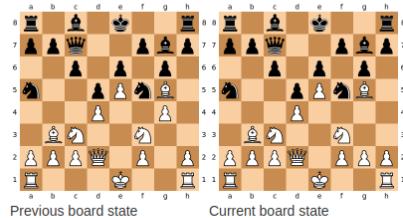
Current board state

You have judge the commentary on :

- 1) Correctness: Is the text commentary a valid commentary for the chess board
- 2) Completeness: Does the commentary correctly describe the chess move which occurred. In other words, given the commentary and the previous board, would you be able to figure out the move which was taken?
- 3) English language Fluency: Is the commentary in fluent English?

1. Commentary text: Back to standing in front of the king !

Which piece was moved: white pawn g2



1.1 Is commentary correct for the shown chess move?

Yes No

1.2 Can you infer what the move is from commentary given only the previous board state?

Yes No

1.3 On a scale of 1-5, with 5 being most fluent, rate the English fluency of the commentary text

1 2 3 4 5

2. Commentary text: \${c2}

Which piece was moved: \${m2}

Figure 8.26: AMT (Amazon Mechanical Turk) sample HIT (Human Intelligence Task): Part 2 of 2: 7 sets of questions are asked to judge quality of generated text. Each of the seven texts is output from a different method.

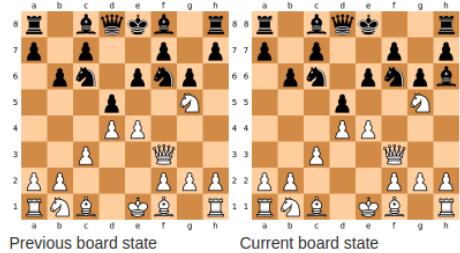


Figure 8.27: Commentary text: *I develop my bishop to the queen .*

An example instance where output commentary from our method was marked as not valid for the given chess move

Checking chess proficiency of annotators

Our proficiency test questions are chosen from a subset of questions by [32]. Each question consists of a chess board and a question about the board configuration or game situation. The paper formulates a range of question types such as enumerating pieces of a type, enumerating pieces of a player, whether one piece threatens another, and whether the configuration corresponds to a checkmate or stalemate. For simplicity we stick to only those question types that have binary answer response.

We classify the question types into **Easy** and **Hard** question types. Each annotator is presented with one **Easy** and one **Hard** question at the start of a HIT.

March 25,2022

Chapter 9

Macro-Level Controllable Generation based on Elements of Narrativity: The Narrative Reordering Problem *(AAAI 2022)* [New Task][Evaluation]

It is thus the narrative, and that alone, that informs us here both of the events that it recounts and of the activity that supposedly gave birth to it. In other words, our knowledge of the two (the events and the action of writing) must be indirect, unavoidably mediated by the narrative discourse, in as much as the events are the very subject of that discourse and the activity of writing leaves in it traces, signs or indices that we can pick up and interpret

Gerard Genette, *Narrative Discourse*, 1980

Narratology and its aspects as an axis of variation are understudied in NLP, as highlighted

recently in [162] . For NLG, narrative aspects such as e.g., narrative order, focus, person of the narrator, omniscience of the narrator etc, also variously called *narratological variables*, or *elements of narrativity*, can provide a rich source of control goals around which control tasks can be framed to better understand the macroplanning skill of NLG architectures.

Many implicit inferences can be drawn at each point by a reader while they are reading a text, depending on how it is structured, that can critically impact the text’s evolving interpretation and meaning in the reader’s mind. One such macro-structural aspect present in text with an underlying chronology of events (i.e., a “story”) as its background, is the order of their presentation. For narratives or stories, this is known as the *narrative order*. Reordering a narrative can impact the temporal, causal, event-based, and other inferences readers draw from it, that in turn can have strong effects both on its interpretation and interestingness.

In this chapter, we define and investigate the task of *Narrative Reordering* (NAREOR) where the communicative goal involves rewriting a given story in a different narrative order while preserving its plot. A NLG system that can rewrite the story in a different narrative order such that it retains its coherence will also ensure adequate interpretation and understanding by the reader. We present a dataset, NAREORC, with human rewritings of stories within ROCStories in non-linear orders, and conduct a detailed analysis of it. Further, we devise novel task-specific training methods with suitable evaluation metrics. We perform experiments on NAREORC using state-of-the-art large, pretrained NLG models such as BART and T5 and conduct extensive automatic and human evaluation of their outputs. We demonstrate that although our models can perform decently, NAREOR is a challenging task with potential for further exploration, with a significant gap still left to bridge between model and human performance. We also investigate two applications of NAREOR: generation of more interesting variations of stories and serving as adversarial sets for temporal/event-related tasks, besides discussing other prospective ones, such as for pedagogical setups related to language skills like essay writing and applications to medicine involving clinical narratives.

9.1 Introduction

From the onset of language, storytelling has been crucial to the transmission of knowledge [176]. It has been well-established that readers remember only an abstract representation of stories [203]. Before the printing press, classes engaged with oral teaching of scriptures, such as rabbis, underwent extensive training to reproduce them with no distortion [17]. Formally analyzing story structure commenced with the ancients, through works like Aristotle’s *Poetics*

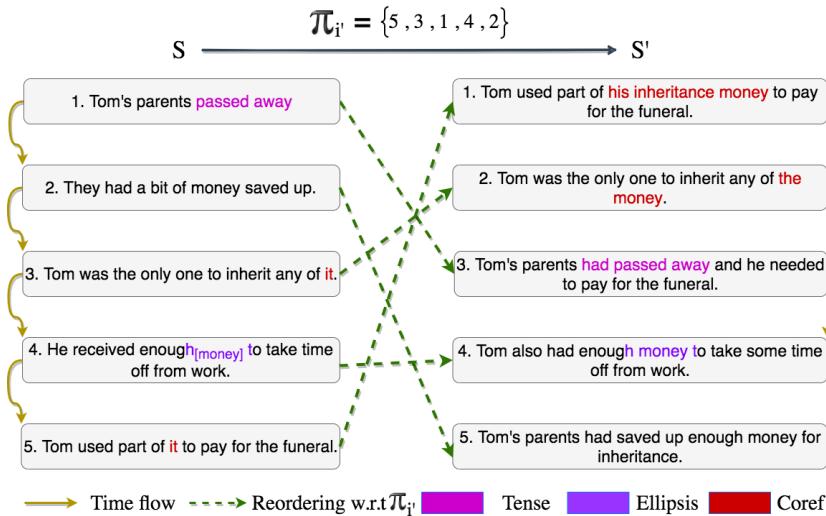


Figure 9.1: Example of our task and dataset, with original input story S on the left, target narrative order $\pi_{i'}$ on the top, and human rewritten story S' on the right.

[71]. These studies led to the concept of a *narrative*, distinct from story events.

For a story, there are two *orders*: the chronological order of events as they happened and their order as presented in text. These have been analyzed under different names [166]. We refer to them as *story order* and *narrative order*, or *story* and *narrative*, respectively. Genette [66] enlists typical orders observed in writing. A *linear* order narrates events in same sequence as story order. The *in medias res* order starts with events in the middle, goes back to the start, then proceeds to the end. Changing from near-*linear* to more “interesting” orders is prevalent in cinema, e.g., *The Imitation Game* starts with Turing’s post-WWII 1951 interrogation. *Memento* and *Naked Lunch* are known for their esoteric narrative orders - loosely described as retrograde (reverse of linear) and syllepsis (lacking chronological logic), respectively.

Morgan [140] explains how narratives surpass “mere chronicle”. Narrative orders of presenting materials in scientific *explanations* directly affects how researchers *interpret* and *understand* them since the order implies not only temporal but other inferences about causality, processes of change, etc. Narrative order can thus influence *model explainability*, especially for explanation generation [175], a recent area-of-interest [238].

In this work, we do not delve into the complex and somewhat subjective question of *which* narrative order is most suitable or “interesting”. We focus on *how* a given story in *linear* narrative order, i.e., story order, can be rendered in a specified, *non-linear*, *target* narrative order that differs from the story order while being still interpretable and preserving plot. We call this *Narrative Reordering*, or NAREOR. To the best of our knowledge, we are the first to define and investigate this task in a modern, machine learning-based setup.

There exists prior work which has addressed problems of a similar nature. Montfort [139] tries generating fiction narratives from basic existent-event info with a special focus on narrative order, using a rule and planning based approach. Unlike our work, their rule-based system does not involve learning. Moreover, being generation in a given narrative order from unstructured story elements rather than reordering and rewriting an existing story, their setting does not require solving challenges such as disentangling events from stories which are inherent in NAREOR.

The reordering portion of NAREOR can be formalized as follows: Formally, NAREOR involves converting a story S with sentences s_1, s_2, \dots, s_n to a reordered, rewritten story S' with sentences s'_1, s'_2, \dots, s'_n according to a given target narrative order $\pi_{i'}$. $\pi_{i'}$ is a permutation $\{\pi_{i'} | \pi_{i'} : i' \rightarrow f(i'); 1 \leq i' \leq n; f(i') = i\}$ mapping from target sentence¹ indices i' to original sentence indices i , where f is a one-to-one and onto function from $\{1, 2 \dots n\}$ to itself. In practice, we write $\pi_{i'}$ as the sequence $\{i = f(i')\}_{i'=1}^{i'=n}$ (f and i' become implied).

NAREOR's challenges are evident from the example in Figure 9.1. Simply reordering sentences is far from sufficient, as rewritten text must be adjusted to handle coreference, tense, and other discourse dependencies. Reichenbach times [178] refer to the speech time, event time and reference time of an event as it appears in a story. These refer to the story times when the event is spoken about (ST), when the event actually happened (ET), and the reference point (RT) with respect to which it is spoken about. As an example, consider the sentence "By this time three years ago, it was already two years past graduation". For the "graduation" event, the event time is five years ago, the reference time is three years ago, and the speech time is now. Narrative order affects tense since it can change 2 of these 3 Reichenbach times [178] – the speech and reference times. NAREOR involves pinpointed and critical edits; a single missed or incorrect edit can result in an entirely different or invalid plot. Since $\pi_{i'}$ can be seen as a *control*, NAREOR is a controllable generation task (see §9.9 for discussion).

NAREOR is also a novel form of story-level paraphrasing and can be used to generate more interesting variations of stories (§9.5.1). Outputs can also serve as challenge sets for temporal or event-based tasks such as sentence ordering to assess the temporal reasoning capabilities of models (§9.6). NAREOR can also be potentially useful for pedagogical setups related to language skills such as essay writing, and applications to medicine involving clinical narratives (§9.6).

To complement NAREOR, we present a dataset, NAREORC, with human rewritings of stories from ROCStories [141] in non-linear orders. We conduct a thorough analysis, examining

¹For simplicity, we assume *narrative* to break up into sentence units. Our task is still very challenging as shown through this chapter.

various ways humans modify the text when reordering (§9.2). We find that, amongst others, NAREOR requires models to have the ability to perform four challenging types of rewrites, which we also refer to as change types.

1. **Ellipsis:** Ellipsis refers to the common phenomenon of excluding words or phrases to prevent staid repetition at a target position, when they have already occurred once at a reference position in the proximal context and their re-occurrence at the target position can be inferred obviously by a reader. E.g., *John had an icecream sundae. Mary did too.* In the mentioned example, “did too” replaces or elides “had an icecream sundae”, which is the inferred meaning. Since NAREOR requires rewriting the story in a new, target narrative order, the reference position may possibly no longer remain within context proximal to the target position. As a result, the erstwhile omitted phrase would have to be explicitly written out while rewriting for the target narrative order.
2. **Tense:** As we already touched upon in our discussion on Reichenbach times previously in this chapter, NAREOR can often require performing significant tense changes.
3. **Timexes:** Narratives which are linear i.e, in the story order, sometimes require use of time expressions to explicitly indicate the quantum or ranges of duration between consecutive events. For example, like in “*John had a heart attack. In an hour, he was dead*”. However, they seldom require indefinite time expressions such as “*After a while*,” and “*Before that*,” etc, since the order of occurrence itself largely clarifies the order in which events happen relative to each other. However, while rewriting for a target narrative order different from story order, this property no longer holds, and the narrative has to introduce and rely on such indefinite time expressions to ensure readers continue to interpret the order of occurrence of events correctly.
4. **Coreference:** Nominal and pronominal referring expressions are often used to refer to the later mentions of an entity in a coreference mention chain, after they have been already named explicitly earlier in the discourse. However, the relative order of occurrence of these mentions within the discourse may change in the specified target narrative order, with the latter mentions potentially even preceding the named mentions. When this happens, it becomes necessary to rewrite these pronominal and nominal mentions to more explicit, named mentions to prevent ambiguity and ensure a correct and plot-preserving interpretation on the part of the reader.

Examples of rewrites pertaining to each of these types can be found in Table 9.1.

We perform experiments with BART, T5, and GPT-2 on NAREORC using novel, task-motivated

training methods we devise (§9.3). We evaluate our models with both an automatic and human evaluation along with qualitative analysis (§9.5). We demonstrate that our devised training methods are effective but have room for further improvement. We illustrate that NAREOR is indeed a challenging task with potential for further exploration.

9.2 Dataset: NAREORC

9.2.1 Dataset Construction

Source Corpus: ROCStories has $\approx 98.5K$ five-sentence English stories (see Table 9.1 for examples). For the dev and test splits, each example contains a four-sentence story prefix with a one-sentence coherent and incoherent ending. We treat the coherent endings as the fifth sentences for NAREORC’s dev and test stories.

Assigning Target Narrative Orders: The target narrative order $\pi_{i'}$ is not part of the ROCStories input. We devise a randomized procedure to assign a reasonable $\pi_{i'}$ for each example. We sample 3 permutations from the set of non-identity $n!-1$ permutations.² We find Kendall τ correlations [88] between identity permutation I_n , $\{1,2,3,4,5\}$, and each of the three permutations, retaining the lowest as $\pi_{i'}$. We prefer this to sampling at random because we want our examples to be sufficiently non-trivial w.r.t. the task.

Supervised & Unsupervised Splits: We set aside 600, 200, 200 stories from train, dev, and test splits of ROCStories. These act as NAREORC’s trainSup, devSup, and testSup splits, for which we collect human references. Remaining stories in each ROCStories split are retained as trainUnsup, devUnsup, and testUnsup of size 95161, 1671, 1671.

Human Annotation: For trainSup and devSup, we annotate one reference per example. For testSup, we collect two each to help reference-based metrics. We conduct our study on AMT. To understand task difficulty, we ask a “Hardness” question with options *VeryEasy*, *Easy*, *Moderate*, *Hard*, *VeryHard*. On average, annotators found $\approx 70\%$ of rewritings to be *Moderate* or *Hard*, demonstrating that NAREOR is quite difficult even for humans. More details in §9.10.

²In our case, $n = 5$ as we experiment with ROCStories.

9.2.2 Dataset Analysis

Overall Statistics

We find human-rewritten stories S' are $\approx 1.2x$ as long as input stories S on average in words and characters. We expect some increase given the narrative reordered story favors resolution of sentence-order dependent elements like ellipses (s_4 and s'_4 in Figure 9.1) and pronouns (s_3 and s'_2 in Figure 9.1) to explicit forms. It also requires insertion of time expressions (e.g., *Before that*, 3rd row, Table 9.1) to clarify the now disrupted flow.

Unique n-gram ratio $UR_n(S)$ is the fraction of unique n-grams of length n in S . We observe all three mean URs ($n = 1, 2, 3$) to decrease from input to reference story. UR_1 : $0.692 \rightarrow 0.669$, UR_2 : $0.940 \rightarrow 0.931$, UR_3 : $0.989 \rightarrow 0.984$. Increased n-gram repetition could have reasons similar to length increase, causing cross-sentence repetition. Figure 9.1 demonstrates this: S only has one instance of *money*. Conversion of *inherit any of it* (s_3) \rightarrow *inherit any of the money* (s'_2) and *enough to take time* (s_4) \rightarrow *enough money to take some time* (s'_4), among other changes, results in four in S' .

How Verb Forms Change

We note changes in occurrence distribution across verb-related *pos* tags from S to S' using NLTK's *pos* tagger. Gerund fraction (*pos=VBG*) (e.g., *I like playing*) increases $7.7\% \rightarrow 9.5\%$. Past participle fraction (*pos=VBN*) (e.g., *He had broken it*) \approx doubles, $6.5\% \rightarrow 12.4\%$. Past tense fraction (*pos=VBD*) (e.g., *He broke it*) decreases $60.9\% \rightarrow 54.6\%$. Other verb-related *pos* fractions remain fairly constant. Increase in past participle can be explained by frequent conversion to past perfect tense during reordering (e.g., *parents passed away* \rightarrow *parents had passed away* in Figure 9.1).

How Narrative Reordering Alters Sentences

We look at corresponding sentence pairs $\{s_i, s'_{i'}\}$ in each story, specifically 4 linguistic change types – ellipsis, tense, time expressions (timexes), coreference. We tried detecting these using off-the-shelf tools, and did not find any for ellipsis. Timex detectors like SUTime [27] only mark strict timexes (e.g., *last Sunday*) but not others (e.g., *before midsems*). We hence hand-annotate these four for each $\{s_i, s'_{i'}\}$ per testSup example. These are further described in Table 9.1. We find over half (51.5%) the examples show ≥ 3 of 4 change types at once, and 89.5% show ≥ 2 . This shows that NAREOR requires performing different changes in tandem.

Change Type	Story Examples with Changes Highlighted
Ellipsis (Sent: 5.7%) (Stor: 27.5%)	S: 1. All of the Ross family has red hair, except Henry. 2. Henry has blonde hair that is very curly. 3. Henry's father often teases Henry's mother about the mailman. 4. The mailman has blonde, curly hair, but he is very ugly. 5. <i>His dad's teasing makes Henry feel bad.</i> ; $\pi_{i'}: \{1, 5, 4, 2, 3\}$ S': 1. All of the Ross family has red hair, except Henry. 2. <i>His dad's teasing about the mailman makes Henry feel very bad.</i> 3. This is because the mailman has blonde, curly hair, but he is very ugly. 4. Henry also has blonde hair that is very curly. 5. Henry's father often teases Henry's mother about the mailman.
Tense (Sent: 19.1%) (Stor: 64.0%)	S: 1. Sam bought a new SUV. 2. It was all wheel drive. 3. He figured he would take it off road. 4. <i>He hit a few hard bumps and broke his suspension.</i> 5. Sheepishly, he brought it to the dealership for repair. ; $\pi_{i'}: \{2, 3, 5, 1, 4\}$ S': 1. Sam's SUV was an all wheel drive. 2. He thought he could take it for a spin off road. 3. Embarrassed by the outcome of his drive, Sam took the car to the dealership for repair. 4. He had just bought the SUV. 5. <i>The car had hit a few hard bumps and the suspension broke when Sam took it off road.</i>
Timexes (Sent: 34.0%) (Stor: 85.5%)	S: 1. There was once a kitten that did not have a home. 2. <i>The poor kitten walked around cold and hungry.</i> 3. One day, a nice lady let the kitten into her home. 4. The woman gave the kitten food and a bed. 5. The kitten was happy to be adopted. ; $\pi_{i'}: \{4, 2, 5, 1, 3\}$ S': 1. A woman gave a home to a cat. 2. Before that it was cold and hungry. 3. It made the cat happy to have a home. 4. The little cat originally was homeless. 5. But in the end, it met the nice woman and she let it in.
Coreference (Sent: 20.7%) (Stor: 71.5%)	S: 1. Jimmy wandered around the city looking for a place for a soda. 2. Before he knew it, he was in an unfamiliar area. 3. He was scared of strangers and didn't want to ask anyone. 4. Soon a policeman came by and asked if he was lost. 5. <i>He told him that he was lost.</i> ; $\pi_{i'}: \{5, 4, 2, 1, 3\}$ S': 1. Jimmy told a police officer that he was lost. 2. He was lucky the police showed up in the first place. 3. He had no idea where he was. 4. He had wandered off when trying to find somewhere to buy a soda. 5. It was pretty terrifying being all alone in a mysterious area with strangers.

Table 9.1: Sentence pairs in testSup stories are annotated for 4 linguistic change types common in NAREORC. Sent denotes % of sentence pairs showing that change type. Stor denotes story pairs (S, S') where \geq one sentence pair shows that change type.

9.3 Methodology

9.3.1 Training Methods

We introduce two task-specific training methods.

NAR-denoise (NAR-d)

This is partially inspired by how humans rewrite; a common approach is to first reorder sentences naively (simply swap positions), then make other changes. NAR-d attempts to mimic this, learning to convert from naive orderings to high-quality text. It involves two stages of model training.

1. **Denoise-1S:** Stage 1 is unsupervised training through story-level denoising. We use train-

Unsup without human-written reorderings, and simulate them using the original human-written ROCStories (the outputs during training). Deletion and swapping of tokens are used to create inputs from these stories that simulate naive reorderings. This noising aims to emulate the reverse of the content editing that occurs during NAREOR. Specifically, we randomly delete 12.5% of tokens and swap another 12.5%. We found human-rewritten stories were, on average, in combination of token length (longer) and swappings, $\approx 25\%$ different from the originals. We split this between deletion and swapping to approximate naively-reordered stories. Story sentences S are first reordered as per $\pi_{i'}$ to produce S'_{naive} , then each is edited to fit the new narrative. We swap tokens as humans often swap words like coreferent mentions based on how the narrative order changes. Hence, this stage learns to denoise text by converting noised versions to human-written text.

2. Denoise-2S: The second stage is supervised training atop the model above. The inputs are the 600 original stories in trainSup, with sentences naively reordered as per target narrative order $\pi_{i'}$ to S'_{naive} , and the outputs are the human rewritings of these. The model learns to further translate from naively-reordered text to fluent human-written text.

NAR-reorder (NAR-r)

Unlike NAR-d, NAR-r models themselves handle reordering given the target order rather than naive reordering beforehand.

- **Input Encoding Scheme:** We describe how the task input $\{S, \pi_{i'}\}$ is encoded as a token sequence for both Stage-1 and 2 training. To enable the model to distinguish different sentences, we prefix each $s \in S$ with a tag from $\langle a \rangle$ to $\langle e \rangle$. We specify $\pi_{i'}$ as a sequence of these, separated from S by $\langle sep \rangle$. NAREOR involves rearranging mention types among coreference chains (see §9.2.2), so we use NeuralCoref [80] to detect these chains. For each, we assign a unique uppercase tag ($\langle X \rangle$) to replace its mentions. At the end of the input, we list each tag and the head mention of its coreference chain in order. We then append $\langle st \rangle$ to mark the end of the input. An illustration of the scheme follows: $\langle a \rangle$ Since I had front seat tickets, I was able to directly see $\langle X1 \rangle$. $\langle b \rangle$ $\langle X1 \rangle$ tried to reach out with $\langle X1 \rangle$ $\langle X2 \rangle$. $\langle c \rangle$ I grabbed $\langle X2 \rangle$ and $\langle X1 \rangle$ pulled me on stage. $\langle d \rangle$ $\langle X1 \rangle$ began to sing. $\langle e \rangle$ The concert had started. $\langle sep \rangle$ $\langle e \rangle$ $\langle d \rangle$ $\langle a \rangle$ $\langle b \rangle$ $\langle c \rangle$ $\langle X1 \rangle$ The music artist $\langle X2 \rangle$ her hand $\langle st \rangle$

- **Reorder-1S:** We use examples from trainUnsup for stage 1. It is problematic to train for the forward direction of our task $S, \pi_{i'} \rightarrow S'$ since S' is not known. Approximating S' using S'_{naive} would hurt output fluency. We instead train in the inverse direction $S'_{naive}, \pi_{i'}^{-1} \rightarrow S$, where $\pi_{i'}^{-1}; \pi_{i'}^{-1}(\pi_{i'}) = I_n$ is the inverse permutation of $\pi_{i'}$. To reduce train-test mismatch, we use the

inverse formulation half the time, and an autoencoding one, i.e., $S, I_n \rightarrow S$ the other half.

- **Reorder-2S:** trainSup examples are used to further finetune on reorder-1S. We train in the task direction $S, \pi_{i'} \rightarrow S'$.

9.3.2 Chosen Models

We choose three large, pretrained NLG models: GPT-2, BART, and T5 for our experiments. Note that we’re aware these represent only a particular, recent subsample of the many NLG architectures devised over the last few decades. As such, our task and its utility is not restricted to these particular architectures, and can be used to assess and compare the content ordering abilities of any set of architectures. We choose these three for our experiments as a reasonable initial choice on account of their recently noted good performance across a wide range of typical generation tasks such as abstractive summarization and prompt-based story generation.

We finetune all using both our training methods to produce denoise-1S (d-1S), denoise-2S (d-2S), reorder-1S (r-1S), and reorder-2S (r-2S) versions. GPT-2 [170] is a Transformer-based language model trained on *WebText*. BART [105] and T5 [172] are Transformer seq2seq models. BART is trained as a denoising autoencoder to reconstruct original from noised text. T5 is designed to be effective for transfer learning. We use HuggingFace’s implementations of their base versions.³

9.3.3 Automatic Evaluation Metrics

Reference-Based Metrics assess the similarity between generated text and human-written references. We use **BLEU** [150], **METEOR** [12], and **BERTScore** [253]. We compare generated text with the two references per testSup example.⁴

Target Order Fidelity (TOF) is defined as how closely the reordered text matches the given target narrative order. e.g., given $S = \{s_1, s_2, s_3\}$, $\pi_{i'} = \{3, 2, 1\}$, and $S' = \{s'_1, s'_2, s'_3\}$, we wish to see if s_1 has correctly been translated to s'_3 . We introduce **TOF-METEOR** and **TOF-BERTScore**. These assess the average METEOR and BERTScore values for each aligned pair $\{s_i, s'_{i'}\} \forall i$ (where i' refers to the target index for s_i). Higher values correspond to more content preservation, where each output sentence is more likely in the correct position. Some drop is expected in modulating for $\pi_{i'}$, but the overall content should be faithful. These metrics serve

³See §9.4 for further training/finetuning details.

⁴Correlates well with human evaluation as shown in §9.5.

more as validation, where reasonable values (e.g., > 50 ⁵) are sufficient. Lower values indicate more changing of the text that may be necessary for certain narrative reorderings.

9.4 Experiments

Model Finetuning and Generation

For finetuning our models, we try different combinations of learning rates (LR) for both stages. We look at either the loss (for BART and T5) or perplexity (for GPT-2) on the respective validation splits (devUnsup for 1st stage and devSup for 2nd), and choose the epoch with the lowest.

We evaluate each model on testSup, where we can directly compare results to NAREORC’s human rewritings. We generate a single output per test example. The inputs are the original examples to NAR-r models and the S'_{naive} of the examples to NAR-d models. See §9.3.1 for more details.

We only keep the first five sentences of each output. For BART and T5, we use beam search with a width of 5.⁶ For GPT-2, we use a nucleus sampling budget [77] of 0.9 and output length limit of 500. We try various softmax temperatures and find 0.9 performs best. For GPT-2, during finetuning, it is given the concatenation of the input plus output. During generation, it is only fed the input for which it generates a continuation (the output). We noticed that many GPT-2 generations included trailing exclamation marks, and strip these if more than four occur in a row.⁷

Human Evaluation

Annotators evaluate 100 testSup examples each from the original stories, human rewritings, outputs from our two-stage models, and a subset of one-stage models. Each example is evaluated by two annotators. See §9.12 for more.

They evaluate fluency, coherence, logic, and plot preservation (plot-pres) on 1-5 scales. Fluency is a measure of how fluent and readable a text is. Coherence is how well individual sentences fit together [13]. Logic is the plausibility of described events. Plot-pres is how well reordered text preserves the plot of the original. This includes details about characters, events, and interactions between them, encompassing its semantic and temporal aspects.

We also conduct an interestingness (interest) study on human rewritings and outputs from

⁵Assuming the values are multiplied by 100.

⁶Nucleus sampling did not work as well for BART and T5.

⁷See §9.11 for more finetuning/generation details.

our BART-2S and T5-2S models. Each reordered story’s interestingness w.r.t. suspense and time flow compared to the original are evaluated from 1-5 by two annotators. We ask the following: “*On a scale of 1-5, with 1 being most decrease in interestingness and 3 being same level of interestingness and 5 being most increase in interestingness, how interesting is the suspense and flow of time in the story S, compared to the original story O? How exciting did you find the story as you read through it?*”

9.5 Results and Analysis

We present evaluation results of our 2S and subset of 1S models on testSup compared to human rewritings and original stories. Tables 9.2 and 9.3 contain human evaluation results, and Table 9.4 automatic evaluation results. Correlations between automatic and human metrics are in Table 9.5. Table 9.6 contains qualitative examples, with more in §9.13.

<i>Method\Metric</i>	Fluency	Coherence	Logic	Plot-pres
Original stories	4.209	4.0	3.851	N/A
Human rewritings	3.797	3.723	3.784	3.972
GPT2-d-2S	3.635	3.399	3.399	3.708
GPT2-r-2S	3.595	3.378	3.291	3.375
BART-d-1S	3.628	3.412	3.318	3.847
BART-d-2S	3.818	<u>3.507</u>	3.493	3.722
BART-r-2S	3.757	3.439	3.493	<u>3.861</u>
T5-d-2S	3.764	3.419	<u>3.5</u>	3.889
T5-r-1S	3.655	3.378	3.486	3.847
T5-r-2S	<u>3.784</u>	3.595	3.520	<u>3.861</u>

Table 9.2: Average human evaluation results on testSup (excluding interestingness), rated from 1-5. Bold corresponds to best model performance per metric, and underline second-best model performance.

<i>Method:</i>	Human	BART-d	BART-r	T5-d	T5-r
Interest	3.75	3.367	3.483	<u>3.533</u>	3.3

Table 9.3: Average interestingness results on testSup, rated from 1-5 (3 represents equal to original story). Models are 2S versions. Bold corresponds to best performance, and underline second-best.

<i>Method\Metric</i>	BERTScore	BLEU	METEOR	TOF-BERTScore	TOF-METEOR
Human rewritings	N/A	N/A	N/A	66.85	56.79
GPT2-d-2S	60.75	37.01	45.20	79.23	74.23
GPT2-r-2S	58.03	32.57	40.85	73.04	63.00
BART-d-1S	67.14	44.73	49.88	95.61	93.43
BART-d-2S	<u>67.93</u>	<u>46.03</u>	<u>50.54</u>	93.55	90.81
BART-r-2S	67.16	44.63	49.16	91.32	86.43
T5-d-2S	67.99	46.95	51.12	94.20	91.83
T5-r-1S	66.24	43.40	48.20	89.85	84.26
T5-r-2S	66.62	44.30	49.00	91.61	86.16

Table 9.4: Average automatic evaluation results on testSup (values multiplied by 100). Bold corresponds to best performance per metric, and underline second-best (excluding the TOF metrics that are mainly for validation).

<i>Metric</i>	<i>Correlation</i>	Fluency	Coherence	Logic	Plot-pres	Interest
BERTScore	Pearson	0.130 (4e-04)	0.139 (1e-04)	0.125 (0.001)	0.255 (1e-06)	0.111 (0.226)
	Spearman	0.106 (0.004)	0.124 (0.001)	0.127 (0.001)	0.211 (5e-05)	0.117 (0.201)
BLEU	Pearson	0.144 (9e-05)	0.140 (1e-04)	0.113 (0.002)	0.219 (3e-05)	0.174 (0.047)
	Spearman	0.130 (4e-04)	0.129 (4e-04)	0.123 (0.001)	0.179 (0.001)	0.171 (0.049)
METEOR	Pearson	0.107 (0.003)	0.125 (0.001)	0.108 (0.003)	0.203 (1e-04)	0.120 (0.191)
	Spearman	0.098 (0.008)	0.114 (0.002)	0.122 (0.001)	0.164 (0.002)	0.121 (0.187)

Table 9.5: Pearson and Spearman correlations between automatic and human evaluation metrics, with p-values in brackets. TOF metrics excluded as they are mainly for validation. Bold corresponds to highest correlation per human evaluation metric.

9.5.1 Analysis of Human Evaluation Results

We begin by analyzing human evaluation performance through results in Tables 9.2 and 9.3.

Fluency, Coherence, Logic: Original stories score the highest for all three metrics with human rewritings second for coherence and logic, beating the models by a noticeable degree. BART-d-2S and T5-r-2S are generally the best-performing models here. BART-d-2S slightly outperforms human rewritings on fluency, with T5-r-2S closely behind, demonstrating that these models are quite fluent. These models also outdo their 1S variants. GPT-2 models perform worst on all metrics.

Plot-pres: We see that human rewritings best preserve the plot of the original stories. T5-d-2S is the best performing model on plot-pres, followed by BART-r-2S and T5-r-2S. GPT-2 models perform the worst at preserving the plot of the original stories (which we show qualitatively in §9.5.3).

Method	Text
Original (ex. 1)	Jimmy wandered around town looking for a soda. Before he knew it, he was in a strange area. He was scared of strangers and didn't want to ask anyone. Soon a policeman came by and asked if he was lost. He told him that he was lost. {5,4,2,1,3}
BART-d	Jimmy told a policeman that he was lost. The policeman had come by and asked if he had been lost. Before he knew it, he was in a strange area. Jimmy had wandered around town looking for a soda. He was scared of strangers and didn't want to ask anyone.
BART-r	Jimmy told the policeman that he was lost. The policeman came by and asked if he had lost his wallet . Before Jimmy knew it, he was in a strange area. He had wandered around town looking for a soda. He was scared of strangers and didn't want to ask anyone.
T5-d	Jimmy told a policeman that he was lost. Soon after , a policeman came by and asked if he was lost. Before he knew it, he was in a strange area. He had wandered around town looking for a soda. He was scared of strangers and didn't want to ask anyone.
T5-r	Jimmy told a policeman that he was lost. A policeman came by and asked if he had a soda . Before he knew it, he was in a strange area. He had wandered around town looking for a soda. He was scared of strangers and didn't want to ask anyone.
Original (ex. 2)	Every morning, a bird woke Fred up for work at 6 AM. Fred began to like the bird. And he enjoyed the sound of its singing. One day, the bird no longer sang. He hoped nothing bad had happened to the bird. {4,1,3,5,2}
Human	Fred didn't hear the bird singing this morning which is unusual. The bird had been waking him up every single day at 6 AM for quite some time. He had grown accustomed to the bird being his alarm clock . Now he's worried that something might have happened to the poor bird. He almost considers the bird a friend at this point.
GPT2-r	One day, Fred's bird began to sing. Every morning, Fred woke him up for work at 6 AM. This was because he liked Fred . He hoped nothing bad had happened to Fred. This bird, however, did not like Fred .
BART-r	Fred no longer sang. Every morning, he woke Fred up for work at 6 AM. He enjoyed the sound of his singing. He hoped nothing bad had happened to him. He had begun to like the bird.
T5-d	One day, the bird no longer sang. Every morning, a bird woke Fred up for work at 6 AM. He had enjoyed the sound of its singing. He had hoped nothing bad had happened to the bird. After a while , Fred began to like the bird.
Original (ex. 3)	Eric was walking around a pond at a local park. He saw a turtle on a rock few feet offshore. He stepped into the water to try to catch the turtle. Suddenly, he sank knee deep into the mud and got stuck. Someone walking on the shore ran over to rescue Eric. {5,4,1,3,2}
Human	Someone who had been walking on the shore ultimately ran over and rescued Eric from the mud. Eric's knee had sunk deep into the mud, rendering him stuck. Earlier, Eric had been walking around a pond while at a local park. Trying to catch a turtle in the pond, Eric stepped into the water. Eric did so because he had seen a turtle poached on a rock few feet offshore.
BART-d	Someone walking on the shore ran over to rescue Eric. Earlier , he had sunk knee deep into the mud and got stuck. Eric was walking around a pond at a local park. He stepped into the water to try to catch the turtle. He had seen a turtle on a rock few feet offshore.
BART-r	Someone walking on the shore ran over to rescue Eric. Eric sank knee deep into the mud and got stuck. He was walking around a pond at a local park. He had stepped into the water to try to catch the turtle. He saw the turtle few feet offshore.
T5-r	Someone walking on the shore ran over to rescue Eric. Suddenly , Eric sank knee deep into the mud and got stuck. He was walking around a pond at a local park. He stepped into the water to try to catch the turtle. He saw the turtle few feet offshore.
Original (ex. 4)	Joey went to Disneyland with his family. He was so excited to meet Mickey Mouse. When he met Mickey Mouse he was so excited he couldn't speak! Joey finally said hi to Mickey and it was great! Joey had a great time. {5,2,4,1,3}
GPT2-d	Joey had a great time. He was so excited to meet Mickey Mouse. He had finally said hi to Mickey and it was great! He had gone to Disneyland with his family. When he met Mickey Mouse he was so excited he couldn't speak
GPT2-r	Joey had a great time on the roller coaster at Disneyland. He was so excited to meet Mickey Mouse he couldn't speak! (...) He was so excited that he went to the park with his family. He was so excited when he met Mickey Mouse and he was so excited .
BART-r	Joey had a great time at Disneyland . He was so excited to meet Mickey Mouse. He finally said hi to Mickey and it was great! He had gone to Disneyland with his family. When he met Mickey he was excited he couldn't speak!
T5-d	Joey had a great time at Disneyland . He was so excited to meet Mickey Mouse. He had finally said hi to Mickey and it was great! He had gone to Disneyland with his family. When he met Mickey Mouse he was so excited he couldn't speak!

Table 9.6: Qualitative examples for testSup. Target perms are in brackets beside original stories. *d* refers to denoise, and *r* to reorder (2S models).

Interestingness: Human rewritings score highest on interest. Humans rewrite the text in more creative ways, whereas BART and T5 models are more conservative (see §9.5.2 TOF and §9.5.3). Narrative reorderings for all methods are more interesting, on average, than original stories. NAREOR can indeed be used to generate more interesting story variations.

9.5.2 Analysis of Automatic Evaluation Results

We now analyze the automatic evaluation performance of the different methods in Table 9.4.

BERTScore, BLEU, METEOR: We see from Table 9.5 that these reference-based metrics correlate quite well with human eval metrics, particularly plot-pres. T5-d-2S performs best followed by BART-d-2S. Similar to the human evaluation, 2S models outperform their 1S variants, and GPT-2 models perform worst overall. Denoise outperforms reorder variants and generate more similar text, on avg, to human references.

Target Order Fidelity (TOF): It appears all approaches are reasonable (e.g., > 50 for TOF metrics), and outputs are likely in the correct target orders. Human rewritings have the lowest TOF; humans are less conservative while rewriting (shown in §9.5.3). GPT-2 models modify text second heaviest, but perform worst overall. They introduce more errors, e.g., repeating or hallucinating to degrade text quality and plot-pres (§9.5.3). BART and T5 models are more conservative. It appears they have learned to perform minimal but effective edits (§9.5.3). They lag behind humans and heavier editing may be required to further improve. Lastly, it appears the reorder models modify text more heavily than their denoise variants.

9.5.3 Qualitative Analysis

From Table 9.6, we see that humans modify text heavily to suit the reorderings and are sometimes quite creative, e.g., phrasing Fred as having *grown accustomed to the bird being his alarm clock* (ex. 2). Humans successfully handle necessary coreferences, tenses, time expressions (timexes), etc.

GPT-2 modifies text quite heavily but suffers from incorrect coreference while introducing spurious tokens, repetition, or hallucinations. For ex. 2, GPT2-r changes the plot greatly, stating *Fred woke him up for work and This was because he liked Fred* (likely due to poor coreference), and hallucinating *This bird, however, did not like Fred*. For ex. 4, it repeats Joey’s excitement many times, while hallucinating a *roller coaster* that was absent in the original story.

BART and T5 models are more conservative, but their edits are important and effective. They handle coreference, tense, and timexes quite well. These pinpointed and critical edits are required to maintain plot. For ex. 1, they modify *He told him that he was lost* to **Jimmy told a/the policeman that he was lost** given that sentence is now at the beginning. BART-d impressively modifies tense by converting *Soon a policeman came by and asked if he was lost* to **The policeman had come by and asked if he had been lost**. For ex. 2, T5-d converts *enjoyed* to *had enjoyed* since the bird no longer singing is now prior information, and adds the timex *After a while* to the beginning of the last output sentence. BART-r successfully changes *Fred began*

*to like the bird to He **had begun** to like the bird.* For ex. 3, BART-d inserts the timex *Earlier* at the beginning of the second output sentence, correctly and unambiguously conveying its underlying temporality w.r.t. the first. BART-d correctly changes *saw a turtle* to **had seen a turtle**, while BART-r does so for *stepped* to *had stepped*. For ex. 4, BART and T5 models all resolve the *Disneyland* ellipsis by converting *Joey had a great time* to *Joey had a great time at Disneyland*, while GPT2-d cannot.

However, the BART and T5 models are imperfect. For ex. 1, BART-r hallucinates *lost his wallet* (original story does not involve a wallet), T5-d inserts an incorrect timex of *Soon after* at the beginning of the second output sentence, and T5-r hallucinates *asked if he had a soda* (this is not asked in the original story). For ex. 2, BART-r incorrectly converts **the bird no longer sang** to **Fred no longer sang**, likely due to coreference difficulties. For ex. 3, T5-r does not convert *Suddenly* to *Earlier* like BART-d, giving a false interpretation that Eric slipped after his rescuer’s arrival. BART-r does not mislead with *Suddenly*, but is ambiguous with no timex at all. Further, BART and T5 are more conservative than humans.

9.5.4 Overall Takeaways

Humans modify text greatly while successfully performing NAREOR. BART and T5 models perform decently with minimal but effective edits. GPT-2 models tend to repeat, hallucinate, and reduce text quality and plot preservation.

Based on human (§9.5.1) and automatic (§9.5.2) evaluation, BART-d-2S and T5-d-2S are the best models overall. BART-d-2S outdoes its reorder variant, possibly due to BART’s pretraining as a denoising autoencoder, closer to our denoise training method. For T5, both methods perform quite well and show potential. However, T5-d outperforms on plot-pres (Table 9.2), interest (Table 9.3), and automatic metrics (Table 9.4). The denoise training method appears to be slightly more effective, possibly because it is partially inspired by how humans perform NAREOR (see §9.3.1). These are the first two task-specific training methods for NAREOR which we devise ourselves, each approaching the task differently (see §9.3.1). 2S models also mostly outperform 1S ones, demonstrating that second stage finetuning improves upon the first.

BART and T5 models are quite effective, excelling at fluency, but have further room for improvement in coherence, logic, plot-pres, and interest. §9.5.3 shows they still suffer from several issues. Their conservative tendency may limit their NAREOR ability compared to humans. Overall, these models serve as strong initial baselines for NAREOR while underscoring the task’s difficulty and potential for exploration.

9.6 Applications of NAREOR

Sentence ordering involves reconstructing original sentence order of an unordered sentence set [13]. NAREORC’s reordered stories could serve as a challenge set for sentence reordering models due to their non-linear narrative structure underrepresented in corpora. We use the implementation of Prabhumoye et al. [165] to train i) M_{ext} , an external model on the SIS corpus [79], ii) M_{iid} , an in-domain model on first 20% of ROCStories’ train split. We test each on i) Control set $\{s_i\}_{i=1}^{i=n}$, input stories from testSup, ii) Challenge set $\{s'_i\}_{i=1}^{i=n}$, reordered stories from testSup. Table 9.7 shows drastic drops across metrics (higher is better - see Prabhumoye et al. [165]) for both M_{ext} and M_{iid} from control to challenge set, confirming our hypothesis.

<i>Model</i>	<i>TestSet</i>	<i>SentAcc</i>	<i>Rouge-S</i>	<i>LCS</i>	<i>Kendall τ</i>
M_{ext}	Control	76.35	48	59.1	0.57
	Challenge	52.4	24.7	29.7	0.12
M_{iid}	Control	66.4	85.3	84.8	0.75
	Challenge	21.9	49.6	58	0.03

Table 9.7: Sentence ordering on control vs. challenge sets.

Systems with ability to manipulate narrative variables like order could be important for automating pedagogical setups, especially for fine-grained language skills such as *argumentation in essay writing*. As Wingate [241] explains, tutor understanding is found deficient and methods of feedback for students are inconsistent or vague. Language in school texts follows a characteristic register, that often differs from registers students handle in everyday conversation [204]. Models (e.g., NAREOR ones) that can control elements of register, e.g narrative order, can be used to tailor such content to intended settings and bridge this gap.

Systems that can generate event timelines for clinical narratives, e.g., admission notes and physical reports, is important for applications like medical document summarization [19, 179] and clinical decision making [37]. Raghavan et al. [174] demonstrate that cross-narrative temporal ordering of medical events is vital to generating a comprehensive timeline over a patient’s history. Aligning multiple medical event sequences using coreference information and temporal relations has a large impact on their presentation and effectiveness. Our NAREOR models may be effective here and improve upon existing systems.

9.7 Related Work

There exists work on the sentence ordering task discussed in §9.6. For example, Chen et al. [30] learn pairwise orderings of sentences using a ranking model. Unlike sentence ordering, NAREOR involves reordering and rewriting a sequence of sentences to fit a new narrative order.

TALESPIN [131] was an early goal-based story generator. There has since been work on related tasks like story cloze test [142, 143] and generation from prompts [46, 207]. Some works explore controllable variants, e.g., with keywords as control [156]. NAREOR is distinct as it aims to preserve the underlying plot while controlling a story-level aspect for an already-complete story.

There is also narrative order visualization work. For example, Kim et al. [92] annotate story order for movie scripts and visualize narrative order as a function of story order.

9.8 Conclusion and Future Work

We devised the macro-level controllable generation task of Narrative Reordering (NAREOR) and introduced a dataset, NAREORC, with task-specific training methods and evaluation metrics, and experimented with T5, BART, and GPT-2. Extensive evaluation and qualitative analysis demonstrated that our models are quite effective but can be further improved, and that NAREOR is challenging with potential for further exploration. We showed that NAREOR can be used to produce more interesting story variations and can also be deployed as a challenge set for tasks like sentence ordering.

Our work serves to address the general dearth of research on discourse-level or macro-level controllable generation in NLG, and closes this gap by formulating NAREOR and collecting the NAREORC benchmark. Our experiments strongly hint towards true sequence-to-sequence pretrained generators, that have an explicit encoder, such as BART and T5 to be relatively superior at macro-level content ordering compared to pure language model based generators such as GPT-2.

Future directions include exploring training ideas better emulating human rewrites. NAREOR can be explored as document-level paraphrasing for applications like data augmentation for document tasks, adversarial sets for more temporal tasks, and applications for education and medicine discussed in §9.6. We also hope our work drives investigation of more challenging task variations (e.g., sub-sentential). Lastly, NAREOR’s controllability aspect can be investigated further in even more challenging and diverse test settings to more exhaustively assess

the macroplanning capabilities than we do here.

9.8.1 Broader Takeaways

This chapter exemplifies how elements of narrativity can provide a viable setting to experiment with discourse-level controllable generation tasks to test the macroplanning skills of NLG models, specifically, their skill at the macroplanning subtask of content ordering. Henceforth, a promising frontier for future exploration could be framing similar macro-level control tasks and benchmarks around other elements of narrativity such as *focus* (character who is depicted as the central figure) and *narrator configuration* (first person vs third person, omniscient vs limited knowledge) etc. Since all of these tasks would have plot-preservation as one of the communicative subgoals, they would, as was the case with narrative reordering in this chapter, largely be expected to hold the content constant, thus isolating out and testing model ability at aspects of macroplanning other than content selection, and particularly, content ordering.

As an example, let us discuss a task involving another element of narrativity. Consider the task of converting from an omniscient narrator to one where the narrator is a specified character in the story and hence only knows what the character knows. The omniscient narrator has access to a global view of the actual story's trajectory, both in terms of time and space. Hence the narrator can include sentences such as "This one decision would in time dramatically alter Odysseus's life and lead him to lands far-flung beyond his conception.". The limited knowledge, character narrator on the other hand, only has access to a egocentric view of time and space depending on the knowledge of the character embodied by the narrator. Naturally, if this character itself would be some kind of omniscient being (e.g a singular God), then this would become equivalent to an omniscient narrator. However, such characters are often non-existent or present only upto a limited extent; typical stories do contain atleast one if not more non-omniscient characters. Even mythological stories and sagas do contain non-omniscient characters such as common citizens, heroic albeit mortal humanoid characters like humans, elves and orcs etc. Hence, re-narrating the story when the specified narrator is a non-omniscient character is likely to require performing non-trivial changes to accomodate the constraint of an egocentric view. For instance, since the story has to be rewritten so that events would now be narrated from the perspective of the character chosen as narrator, their reference times would change, requiring a corresponding change in tense.

Appendices

9.9 Discussion: NAREOR as a Hard Instance of Controllable Generation

In a simple generation task, models have to learn to generate a natural language output o given some input i . Controllable generation tasks [45, 89, 156] differ in that models must generate o given i and an additional “control variable” or goal g . g typically is a desired property of o not already determined by i , such as sentiment, politeness, and so forth. NAREOR is an instance of controllable generation at the story level, where i is the original story S , g is the target order $\pi_{i'}$, and o is the reordered story S' . NAREOR is non-trivial, due to:

1. **Complex Goal, Challenging Invariant:** $g = \pi_{i'}$ can take on many distinct values ($n! - 1$). Typical goal variables for controllable generation are either sparse sets (e.g., keywords) or discrete variables with few possible values (e.g., sentiment/style). Handling a many-valued variable is harder as outputs must be varied over a larger scale with variation in g while maintaining faithfulness to i . The invariant $o_{/g}$ denotes input faithfulness, which should have the same value across outputs o even for different g . For NAREOR, $o_{/g}$ represents whether the output plot matches the input plot (i.e., plot-preservation). This is difficult to maintain as it requires a strong understanding of story event order, characters, interactions between them, and ensuring these factors all stay consistent.
2. **Extra-Sentential, Discourse Sensitive:** Existing controllable generation tasks, such as data-to-text generation with keyword goals [45], generate sentence-level outputs. NAREOR requires generating a full story. As described in §9.2.2, performing well on NAREOR entails learning several types of discourse dependencies such as ellipsis, coreferences, time expressions, etc. Though there are multiple-sentence generation tasks, they do not usually require a model specifically good at discourse dependencies, nor assess these aspects directly.

9.10 NAREORC Annotation Details

We collected NAREORC using AMT over 15 days, in a manner consistent with terms of use of any sources and intellectual property and privacy rights of AMT crowd workers. Almost all source content for NAREORC was based on ROCStories [142], an already publicly available

and widely used dataset in NLP research.

We restricted annotators to be from Anglophone countries with a prior approval of > 97%. A pool of 98 different annotators participated. We manually verified each submitted story, rejecting and republishing it if found unsatisfactory. The hardness distribution was *VeryEasy* (5.90%), *Easy* (19.03%), *Moderate* (38.73%), *Hard* (31.29%), *VeryHard* (5.03%). The specific instructions, one of the examples provided, and a snippet of the question page can be seen in Figure 9.2.

Crowd workers were fairly compensated: \$1.5 per rewritten story, for a roughly 5-7 min task. This is at least 1.5-2 times the minimum wage in the U.S.A. of \$7.25 per hour (\$0.725 per 6 minutes). We neither solicit, record, request, or predict any personal information pertaining to the AMT crowd workers during the annotation process.

9.11 Further Model Finetuning and Generation Details

For GPT-2 models, we use the base version with 117M parameters. We use finetuning and generation seeds of 42, and finetune with a batch size of two for the 1st stage and one for the 2nd stage. GPT2-d-1S reaches lowest validation perplexity (val-PPL) of 5.733 after 4 epochs, and GPT2-d-2S reaches lowest val-PPL of 7.384 after 7 further epochs, using LRs of 5e-5 and 5e-6, respectively. GPT2-r-1S reaches lowest val-PPL of 2.576 after 4 epochs, and GPT2-r-2S reaches lowest val-PPL of 4.486 after 5 further epochs, using an LR of 5e-6 for both.

For BART and T5, we use the base versions with 139M and 220M parameters, respectively. We use a finetuning seed of 42, and set the minimum decoder length to one token. For the NAR-d models, we use a batch size of 32, maximum encoder and decoder lengths of 128, and warmup steps of 1000 and 10 for the first and second stages, respectively. For the NAR-r models, we use a batch size of 16, maximum encoder and decoder lengths of 256, and warmup steps of 2000 and 20 for the first and second stages, respectively.

For BART-d-1S, we find lowest validation loss (val-loss) of 0.1573 after 7 epochs using LR of 3e-05 (\approx 3hrs of train-time). For BART-d-2S, we find lowest val-loss of 0.8828 after 17 further epochs using LR of 1e-05 (\approx 30min of train-time). For BART-r-1S, we find lowest val-loss of 0.1177 after 6 epochs using LR of 1e-06 (\approx 4hrs of train-time). For BART-r-2S, we find lowest val-loss of 0.8949 after 9 further epochs using LR of 1e-05 (\approx 20min of train-time).

For T5-d-1S, we find lowest val-loss of 0.1947 after 2 epochs using an LR of 1e-04 (\approx 1hr of train-time). For T5-d-2S, we find lowest val-loss of 0.8610 after 3 further epochs using an LR of 1e-04 (\approx 10min of train-time). For T5-r-1S, we find lowest val-loss of 0.092 after 2 epochs using

Re-narrate the Story In Given Order (Click to expand)

Read the instructions below carefully and proceed to complete the task:

- Below, you are given a five sentence story narrated in a certain order. We would like you to **re-narrate the story** in the given **target order**, keeping the **plot unchanged**.
- (1) Below, we also provide **two examples (Example 1 and 2)** to help you understand the task better.
 - (2) **Don't just repeat** the same sentences in the new order! : You would need to fix the sentences for **smooth flow of time , causal connections , character references** e.g if a "They" comes before what it refers to (e.g. "Jason's friends"), write what it is referring to explicitly (e.g. "Jason's friends")
 - (3) If a new reader reads the re-narrated story, the **characters, sequence of events, cause and effect, conclusion** they understand from the story should be the **same as that in the original story**.
 - (4) Take care that your re-narrated story is **coherent** and makes sense by itself. (Without further explanation)
 - (5) Try to keep your sentences **fluent** and **grammatically correct**.
 - (5) Once you're done writing, check again that the **underlying plot** of your new story is the **same as the original story** i.e the characters, sequence of events, cause and effect, the conclusion etc are all preserved.
 - (6) Also **verify** once again that your re-narrated story uses the respective content of the original sentences in the given **target order**.
 - (7) Before leaving, do answer our **follow-up question** .
 - (8) Thanks in advance for participating in our study :)

(a)

Example 1

Original Story:

1. David asked his mother if he could pour his own drink.
 2. His mother agreed, but warned him to be careful.
 3. David accidentally knocked over his glass which shattered everywhere.
 4. His mother cheerfully forgave David.
 5. David promised to be more careful in the future.
- Target Order: 4 -> 5 -> 2 -> 1 -> 3**

Re-narrated Story

1. (Earlier 4) David's mother cheerfully forgave David.
2. (Earlier 5) This is because he promised to be more careful in the future.
3. (Earlier 2) His mother had agreed to let him pour his own drink, but warned him to be careful.
4. (Earlier 1) This was after David asked her if he could pour his own drink.
5. (Earlier 3) He ended up accidentally knocking over his glass which shattered everywhere.

(b)

Q 1.1 Re-narrate the story in the given target order, without changing the underlying plot.

Original Story:

1. The Moore family raked leaves together every Fall.
2. Every year the trees seemed to drop more and more leaves.
3. This year the Moore family decided to cut down some of the trees.
4. They cut down three of the eight trees in their yard.
5. The Moore family no longer had to spend so much time raking leaves.

Target Order:

4 -> 2 -> 5 -> 3 -> 1

1. (Earlier 4)

2. (Earlier 2)

(c)

Figure 9.2: Snapshots of a) instructions seen by annotator before writing, b) one of the examples provided, and c) part of the page the annotator interacts with while rewriting.

an LR of 5e-06 (\approx 1hr of train-time). For T5-r-2S, we find lowest val-loss of 0.8619 after 2 further epochs using an LR of 1e-04 (\approx 10mins of train-time).

Training was done using single RTX 2080 Ti and Titan Xp GPUs, and Google Colab instances that alternately used a single V100, P100, or Tesla T4 GPU. Hyperparameters such as the learning rate were determined by trying a range of values (for LR, from 5e-8 to 5e-4), and finding ones that led to good convergence behavior (e.g., validation loss or PPL decreases at a decently steady rate and reaches min. after a reasonable number of epochs).

Instructions: Plot Preservation Study (Click to expand)

Read the instructions given below carefully:

Read the two given stories S1 and S2 carefully.

(1) Answer in Q.1 how well you think S2 preserves the plot/storyline of S1

(2) You can answer on a scale of 1, which means least preservation, to 5, which means most preservation.

(3) By plot here, we mean the actual details about the characters, situations and interactions between them, the events taking place etc which you understand on reading the full story.

(4) By plot, we don't refer to the exact way the story is written or presented. Its possible for two stories with the same plot to be written differently - for example, one can be in formal English and the other one in informal English, or one can be written in linear order from start to end while the other one starts from the middle of the plot and does a flashback. Nevertheless, the actual plot a reader understands from both these written stories would still be the same.

(5) We give a few examples (Examples 1-3) to help you get a better feel of what we mean in (3) and (4). Read the examples carefully!

(6) Once you are done answering, we ask you to in Q.2 to write a short reason for your decision. Note that this need not be very long at all - we just need to get a rough idea of why you thought the plot was preserved, or not preserved.

(a)

Story S1

Eddie's dad took him to a wrestling show. Eddie loved the whole show! Afterwards, he got a wrestler's autograph. He decided wrestlers were the best guys ever. Eddie had the best day with his dad!

Story S2

Afterwards, he got a wrestler's autograph. He decided wrestlers were the best guys ever. Eddie had the best day with his dad! Eddie loved the whole show! Eddie's dad took him to a wrestling show.

Q.1 Plot Preservation: On a scale of 1-5, with 1 being least preserving and 5 being most preserving, how much does S2 preserve the actual plot/storyline of S1?

Q.2 Reason: Write a short reason for your decision on plot preservation above

Answer:

(b)

Figure 9.3: Snapshots of Plot-Preservation study: a) instructions seen by annotator and b) part of the page annotators interact with while answering the question.

9.12 Human Evaluation Study Details

The five human evaluation metrics were split into three AMT studies — one for Fluency, Coherence, and Logic, one for Plot Preservation, and one for Interestingness. Annotators were from

Anglophone countries with > 97% approval rate. Evaluating Interestingness and Plot-pres requires more effort and reading two stories rather than one. To better verify whether annotators completed Plot-pres thoughtfully, we requested them to fill in a *Reason* for their Plot-pres score. Specific instructions and a question snippet can be seen in Figure 9.3.

Crowd workers were fairly compensated: \$0.50 per fluency, coherence, and logic evaluation, \$0.50 per Plot-pres evaluation, and \$0.50 per interestingness evaluation. These are all roughly 2 minute tasks, and hence compensation is at least 1.5-2 times the minimum wage in the U.S.A. of \$7.25 per hour (or \$0.25 per 2 minutes). We neither solicit, record, request, or predict any personal information pertaining to the AMT crowd workers during the evaluation studies.

9.13 Further Qualitative Examples of Rewritten Text

See Tables 9.8 and 9.9 for further qualitative examples using various approaches for testSup stories.

Method	Text
Original story	Amy was playing Clue with her friends. She listened to the evidence. Then she reviewed her suspects. Amy decided on her suspect. She won the game by guessing correctly. {2,5,4,3,1}
Human rewriting	Amy listened closely to the evidence while playing Clue. It paid off when she won the game. She had decided carefully on her suspect. Reviewing them one by one. Her careful decision allowed her to beat the friends she was playing with.
GPT2-d-2S	Amy listened to the evidence. She won the game by guessing correctly. She had the idea that she had chosen a suspect. Her latest research revealed a game she had played with her friends. When it came to games, she was actually quite pleased with her choice of suspect.
GPT2-r-2S	Amy listened to the evidence. She was able to win the game by guessing correctly. She decided on her suspect. She had reviewed her suspects' evidence. The evidence of her team was published in a newspaper.
BART-d-2S	Amy listened to the evidence. She won the game by guessing correctly. She decided on her suspect. Then she reviewed her suspects. This was while Amy was playing Clue with her friends.
BART-r-2S	Amy listened to the evidence. She won the game by guessing correctly. She decided on her suspect. Then she reviewed her suspects. She was playing Clue with her friends.
T5-d-2S	Amy listened to the evidence. She won the game by guessing correctly. She had decided on a suspect. Then she reviewed her suspects. Amy was playing Clue with her friends.
T5-r-2S	Amy listened to the evidence. She won the game by guessing correctly. She decided on her suspect. Then she reviewed her suspects. She was playing Clue with her friends.
Original story	My Dad tried to buy a car yesterday. He went to the local dealership. He had some ideas in mind. But the dealer kept pushing other options. Dad got what he wanted, though. {5,3,2,1,4}
Human rewriting	My Dad ultimately got what he wanted from the dealership. My dad had ideas in his mind for buying a car. With these ideas, My Dad had gone to the local dealership. My Dad tried to purchase a car while at the local dealership. The dealer at the dealership, however, tried pushing other options to My Dad.
GPT2-d-2S	My Dad got what he wanted for Christmas. He had some ideas in mind. He went to the local dealership. He ended up trying to buy a car yesterday. But the dealer kept pushing other options.
BART-d-2S	My Dad got what he wanted, though. He had some ideas in mind. He went to the local dealership. He tried to buy a car yesterday. But the dealer kept pushing other options.
T5-d-2S	My Dad got what he wanted, though. He had some ideas in mind. He went to the local dealership. He had tried to buy a car yesterday. The dealer kept pushing other options.
T5-r-2S	My Dad got what he wanted, though. He had some ideas in mind. He went to the local dealership. He had tried to buy a car yesterday. But the dealer kept pushing other options.
Original story	Matt saw a commercial for new shoes. And they had springy looking cushions in the sole. Matt thought it would allow him to jump higher. So he begged his mom to get the shoes for him. Matt's mom eventually got him the shoes. {5,1,3,4,2}
GPT2-d-2S	Matt's mom eventually got him the shoes. Matt had seen a commercial for new shoes. Matt thought it would allow him to jump higher. He was so proud that he begged his mom to get the shoes for him. The shoes looked springy looking cushions in the shoes.
GPT2-r-2S	Matt thought that springy cushions were great for jumping higher. He saw a commercial for new shoes. Matt thought it would allow him to jump higher. Matt begged his mom to get the new shoes for him. The cushions had long suede feet and had sat on the back of Matt's hip.
BART-d-2S	Matt's mom eventually got him the shoes. Matt had seen a commercial for new shoes. He thought it would allow him to jump higher. So he begged his mom to get the shoes for him. The shoes had springy looking cushions in the sole.
T5-d-2S	Matt's mom eventually got him the shoes. Matt had seen a commercial for new shoes. Matt thought it would allow him to jump higher. He begged his mom to get the shoes for him. The shoes had springy looking cushions in the sole.

Table 9.8: Further qualitative examples for testSup. Target permutations are in brackets beside the original stories.

Method	Text
Original story	Danny bought a new shirt. He wore it to school the next day. As he was eating lunch, he spilled mustard on it. He scrubbed it off very hard. The mustard came out and no one was the wiser. {5,2,4,1,3}
Human rewriting	Danny was able to scrub all the mustard off his shirt before anyone noticed. It was the first time he had worn this shirt to school. The mustard didn't come off easy. The shirt was brand new. Naturally he spilled mustard on it the very first time he wore it!
GPT2-d-2S	The mustard came out and no one was the wiser. Danny wore his shirt to school the next day. He scrubbed it off very hard. Danny bought a new shirt. As he was eating lunch, he spilled mustard on it.
GPT2-r-2S	The mustard came out and no one was the wiser. Danny wore the shirt to school the next day. He scrubbed it off very hard. Danny bought the new shirt. As he was eating lunch, he spilled mustard on the shirt.
BART-d-2S	Danny's mustard came out and no one was the wiser. He wore it to school the next day. He scrubbed it off very hard. Danny bought a new shirt. While eating lunch, he spilled mustard on it.
BART-r-2S	Danny's shirt came out and no one was the wiser. He wore it to school the next day. He scrubbed it off very hard. He had bought a new shirt. As he was eating lunch, he spilled mustard on his shirt.
T5-d-2S	The mustard came out and no one was the wiser. Danny wore it to school the next day. He had scrubbed it off very hard. He had bought a new shirt. As he was eating lunch, he spilled mustard on it.
T5-r-2S	Danny's mustard came out and no one was the wiser. He wore it to school the next day. He scrubbed it off very hard. He had bought a new shirt. As he was eating lunch, he spilled mustard on it.
Original story	Agatha loved to bake cookies. One day she put a batch in to bake. However, after twenty minutes, there was no scent of cookies baking. She went to see what had happened. Agatha found the oven was turned off. {5,1,4,3,2}
GPT2-d-2S	Agatha found the oven was turned off. Agatha loved to bake cookies. She went to see what had happened. However, after twenty minutes, there was no scent of cookies in her oven. It was one day after she had put the batch in to bake cookies.
GPT2-r-2S	Eventually, Agatha found that the oven was turned off and turned off in the bakery. This was a mistake which she had made when she baked a cookie. This happened when she put a batch of cookies in to bake. The bake out of the oven had not been turning out well. The process she had put in the batch wasn't finished.
BART-d-2S	Agatha found out the oven was turned off. Agatha loved to bake cookies. She went to see what had happened. However, after twenty minutes, there was no scent of cookies baking. This was after she put a batch in to bake.
BART-r-2S	Agatha found the oven was turned off. Agatha loved to bake cookies. She went to see what had happened. After twenty minutes, there was no scent of cookies baking. She had put a batch in to bake.
T5-d-2S	Agatha's oven was turned off. Agatha loved to bake cookies. She went to see what had happened. However, after twenty minutes, there was no scent of cookies baking. She had put a batch in the oven to bake.
T5-r-2S	Agatha found the oven was turned off. Agatha had always loved to bake cookies. Agatha went to see what had happened. However, after twenty minutes, there was no scent of cookies baking. One day Agatha put a batch in to bake.

Table 9.9: Further qualitative examples for test⁷⁶. Target permutations are in brackets beside the original stories.

List of Figures

March 25,2022

List of Tables

March 25,2022

Bibliography

- [1] Abubakar Abid, Maheen Farooqi, and James Zou. Persistent anti-muslim bias in large language models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 298–306, 2021. ([document](#))
- [2] Akintunde Akinyemi. Yorùbá oral literature: A source of indigenous education for children. *Journal of African Cultural Studies*, 16(2):161–179, 2003. [4.2](#)
- [3] John Algeo. Blends, a structural and systemic view. *American speech*, 52(1/2):47–64, 1977. [2.1](#)
- [4] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer, 2016. [6.2.3](#)
- [5] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [6.11](#)
- [6] Douglas E Appelt. Planning natural-language utterances to satisfy multiple goals. Technical report, SRI INTERNATIONAL MENLO PARK CA ARTIFICIAL INTELLIGENCE CENTER, 1982. [4](#)
- [7] International Phonetic Association, International Phonetic Association Staff, et al. *Handbook of the International Phonetic Association: A guide to the use of the International Phonetic Alphabet*. Cambridge University Press, 1999. [4.3](#)
- [8] AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 33–40, 2006. [3.10.1](#)
- [9] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by

jointly learning to align and translate. *arXiv:1409.0473*, 2014. [4](#), [1.1.3](#), [2.3.1](#), [3.9](#)

- [10] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(2):423–443, February 2019. ISSN 0162-8828. [6.7](#)
- [11] David Bamman. Natural language processing for the long tail. In *DH*, 2017. [2](#)
- [12] Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005. [5](#), [5.3](#), [9.3.3](#)
- [13] Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008. [9.4](#), [9.6](#)
- [14] Outi Bat-El. Selecting the best of the worst: the grammar of Hebrew blends. *Phonology*, 13(03):283–328, 1996. [2.1](#)
- [15] Ruth Berman. The role of blends in Modern Hebrew word-formation. *Studia linguistica et orientalia memoriae Haim Blanc dedicata*. Wiesbaden: Harrassowitz, pages 45–61, 1989. [2.1](#)
- [16] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008. [7.1](#)
- [17] Gerrit Bos. Jewish Traditions on Strengthening Memory and Leone Modena’s evaluation. *Jewish Studies Quarterly*, 2(1):39–58, 1995. [9.1](#)
- [18] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, and Yejin Choi. COMET: commonsense Transformers for Automatic Knowledge Graph Construction. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *ACL*, 2019. [5](#), [5.2.1](#), [6.1](#)
- [19] Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 189–198, 2006. [9.6](#)
- [20] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996. [2.4](#)
- [21] Roger Brown, Albert Gilman, et al. The pronouns of power and solidarity. *Article*, 1960.

[3.1](#)

- [22] Paweł Budzianowski and Ivan Vulić. Hello, it's gpt-2-how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22, 2019. [1.1.3](#)
- [23] Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Mário Jorge, Célia Nunes, and Adam Jatowt. Yake! collection-independent automatic keyword extractor. In *European Conference on Information Retrieval*, pages 806–810. Springer, 2018. [2](#)
- [24] Yixin Cao, Ruihao Shui, Liangming Pan, Min-Yen Kan, Zhiyuan Liu, and Tat-Seng Chua. Expertise style transfer: A new task towards better communication between experts and laymen. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1061–1071, 2020. [3.10.1](#)
- [25] Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*, 2020. [5.1](#)
- [26] Raman Chandrasekar, Christine Doran, and Srinivas Bangalore. Motivations and methods for text simplification. In *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*, 1996. [3.10.1](#)
- [27] Angel X Chang and Christopher D Manning. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, volume 2012, pages 3735–3740, 2012. [9.2.2](#)
- [28] David L Chen and William B Dolan. Collecting highly parallel data for paraphrase evaluation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 190–200. Association for Computational Linguistics, 2011. [3.7.3](#)
- [29] David L Chen and Raymond J Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135. ACM, 2008. [8.5](#)
- [30] Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Neural sentence ordering. *arXiv preprint arXiv:1607.06952*, 2016. [9.7](#)
- [31] Junyoung Chung, Kyunghyun Cho, and Yoshua Bengio. A character-level decoder without explicit segmentation for neural machine translation. *arXiv:1603.06147*, 2016. [2.2](#)
- [32] Volkan Cirik, Louis-Philippe Morency, and Eduard Hovy. Chess q&a: Question Answering on Chess Games. In *Reasoning, Attention, Memory (RAM) Workshop, Neural Information*

tion Processing Systems, 2015. [8.4.5](#), [8.6.1](#)

- [33] José Coch. Interactive generation and knowledge administration in multimeteo. In *Proc. 9th International Workshop on Natural Language Generation (INLG-98)*, Aug., 1998. [8.5](#)
- [34] Jacob Cohen. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968. [8.4.5](#)
- [35] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008. [1.1.3](#)
- [36] Soenjono Dardjowidjojo. Acronymic Patterns in Indonesian. *Pacific Linguistics Series C*, 45:143–160, 1979. [2.1](#)
- [37] Dina Demner-Fushman, Wendy W. Chapman, and Clement J. McDonald. What can natural language processing do for clinical decision support? *Journal of Biomedical Informatics*, 42(5):760–772, 2009. ISSN 1532-0464. Biomedical Natural Language Processing. [9.6](#)
- [38] Aliya Deri and Kevin Knight. How to make a frenemy: Multitape FSTs for portmanteau generation. In *Proceedings of NAACL-HLT*, pages 206–210, 2015. ([document](#)), [2](#), [2.1](#), [2.5](#), [2.6](#), [2.3](#), [2.7](#), [2.7.1](#), [4](#), [2.8](#)
- [39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. [1.1.3](#)
- [40] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. [6.2.3](#)
- [41] Wenchao Du and Alan W Black. Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 38–43, Florence, Italy, July 2019. Association for Computational Linguistics. [1.2.2](#)
- [42] Wenchao Du and Alan W Black. Boosting dialog response generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 38–43, 2019. [6.3.1](#), [8](#)
- [43] Pablo Ariel Duboue and Kathleen R. McKeown. Statistical acquisition of content selection rules for natural language generation. In *Proceedings of the 2003 Conference on Empirical*

Methods in Natural Language Processing, pages 121–128, 2003. 1.1.3

- [44] Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328, Tilburg University, The Netherlands, November 2018. Association for Computational Linguistics. 6.7
- [45] Henry Elder, Sebastian Gehrman, Alexander O’Connor, and Qun Liu. E2e nlg challenge submission: Towards controllable generation of diverse natural language. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 457–462, 2018. 9.9, 2
- [46] Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*, 2018. 9.7
- [47] Zhihao Fan, Yeyun Gong, Zhongyu Wei, Siyuan Wang, Yameng Huang, Jian Jiao, Xuanjing Huang, Nan Duan, and Ruofei Zhang. An enhanced knowledge injection model for commonsense generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2014–2025, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. ??, 6.6.1, 6.7
- [48] Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*, 2014. 3.1, 3.4.1, 3.10, 3.10.1
- [49] Manaal Faruqui, Yulia Tsvetkov, Graham Neubig, and Chris Dyer. Morphological Inflection Generation using Character Sequence to Sequence Learning. In *Proceedings of NAACL-HLT*, pages 634–643, 2016. 2.1
- [50] Steven Y Feng, Aaron W Li, and Jesse Hoey. Keep Calm and Switch On! Preserving Sentiment and Fluency in Semantic Text Exchange. 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 2701–2711, 2019. 6.7
- [51] Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. Genaug: Data augmentation for finetuning text generators. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 29–42, 2020. 5.5
- [52] Steven Y. Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. A survey of data augmentation approaches for nlp. *ACL*

2021 Findings, 2021. 5.5

- [53] Steven Y Feng, Kevin Lu, Zhuofu Tao, Malihe Alikhani, Teruko Mitamura, Eduard Hovy, and Varun Gangal. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. *arXiv preprint arXiv:2109.03892*, 2021. 1.2.2
- [54] Charles J Fillmore et al. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech*, volume 280, pages 20–32. New York, 1976. 1.1.3
- [55] Marina Fomicheva, Lucia Specia, and Francisco Guzmán. Multi-hypothesis machine translation evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1218–1232, 2020. 5.5
- [56] Maxwell Forbes, Jena D Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. Social chemistry 101: Learning to reason about social and moral norms. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 653–670, 2020. 5.6
- [57] Neal Gabler. The Weird Science of Naming New Products. *New York Times - http://tinyurl.com/lm1q7ex*, 1 2015. 2.1
- [58] Michel Galley, Chris Brockett, Alessandro Sordoni, Yangfeng Ji, Michael Auli, Chris Quirk, Margaret Mitchell, Jianfeng Gao, and Bill Dolan. \triangle -bleu: A discriminative metric for generation tasks with intrinsically diverse targets. In *ACL (2)*, 2015. 5.2.1, 5.5
- [59] Varun Gangal, Harsh Jhamtani, Graham Neubig, Eduard Hovy, and Eric Nyberg. Charmanteau: Character embedding models for portmanteau creation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2917–2922, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. 1.2.2
- [60] Varun Gangal, Harsh Jhamtani, Graham Neubig, Eduard Hovy, and Eric Nyberg. Charmanteau: Character embedding models for portmanteau creation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Copenhagen, Denmark, September 2017. 3.9
- [61] Varun Gangal, Steven Y Feng, Eduard Hovy, and Teruko Mitamura. Nareor: The narrative reordering problem. *arXiv preprint arXiv:2104.06669*, 2021. 1.2.2
- [62] Varun Gangal, Harsh Jhamtani, Eduard Hovy, and Taylor Berg-Kirkpatrick. Improving automated evaluation of open domain dialog via diverse reference augmentation. In

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4079–4090, Online, August 2021. Association for Computational Linguistics. [1.2.2](#)

- [63] Jing Gao, Peng Li, Zhikui Chen, and Jianing Zhang. A Survey on Deep Learning for Multimodal Data Fusion. *Neural Computation*, 32(5):829–864, 05 2020. ISSN 0899-7667. [6.7](#)
- [64] Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133, 2017. [1.1.2, 6.7](#)
- [65] Samuel Gehman, Suchin Gururangan, Maarten Sap, Yejin Choi, and Noah A Smith. Real-toxicity-prompts: Evaluating neural toxic degeneration in language models. *arXiv preprint arXiv:2009.11462*, 2020. ([document](#))
- [66] Gérard Genette. *Narrative discourse: An essay in method*, volume 3. Cornell University Press, 1983. [9.1](#)
- [67] Jonathan Gordon and Benjamin Van Durme. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 25–30, 2013. [6, 6.1, 6.3.2](#)
- [68] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013. [1.1.3](#)
- [69] Prakhar Gupta, Shikib Mehri, Tiancheng Zhao, Amy Pavel, Maxine Eskenazi, and Jeffrey P Bigham. Investigating evaluation of open-domain dialogue systems with human generated multiple references. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 379–391, 2019. [5.1, ??, 5.3, 1, 5.3, 5.5, ??, 5.10](#)
- [70] M. A. K. Halliday. Systemic functional linguistics: Exploring choice: Meaning as choice. In *Systemic Functional Linguistics: Exploring Choice: Meaning as choice*, 2013. [7](#)
- [71] Stephen Halliwell et al. *Aristotle’s poetics*. University of Chicago Press, 1998. [9.1](#)
- [72] Farzaneh Haratyan. Halliday’s sfl and social meaning. In *2nd International Conference on Humanities, Historical and Social Sciences*, volume 17, pages 260–264, 2011. ([document](#)), [1.1](#)
- [73] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*,

CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, pages 770–778. IEEE Computer Society, 2016. [6.11](#)

- [74] Gaurush Hiranandani, Pranav Maneriker, and Harsh Jhamtani. Generating appealing brand names. *arXiv preprint arXiv:1706.09335*, 2017. [2.2](#), [3.1](#)
- [75] Cong Duy Vu Hoang, Gholamreza Haffari, and Trevor Cohn. Decoding as Continuous Optimization in Neural Machine Translation. *arXiv:1701.02854*, 2017. [2.3.2](#)
- [76] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. [8.3](#)
- [77] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019. [9.4](#)
- [78] Eduard H Hovy. Pragmatics and natural language generation. *Artificial Intelligence*, 43(2):153–197, 1990. ([document](#)), [5](#), [1.1](#)
- [79] Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239, San Diego, California, June 2016. Association for Computational Linguistics. [9.6](#)
- [80] HuggingFace. Neuralcoref, 2020. [Online; accessed 29-September-2020]. [9.3.1](#)
- [81] Harsh Jhamtani, Varun Gangal, Eduard Hovy, and Eric Nyberg. Shakespearizing Modern language using Copy-Enriched Sequence-to-Sequence Models. *arXiv preprint arXiv:1707.01161*, 2017. [1.2.2](#)
- [82] Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1661–1671, 2018. [1.2.2](#)
- [83] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. Google’s multi-lingual neural machine translation system: Enabling zero-shot translation. *arXiv preprint*

arXiv:1611.04558, 2016. [3.10](#)

- [84] Aravind K Joshi. An introduction to tree adjoining grammars. *Mathematics of language*, 1:87–115, 1987. [1.1.3](#)
- [85] Hirotaka Kameko, Shinsuke Mori, and Yoshimasa Tsuruoka. Learning a game commentary generator with grounded move expressions. In *Computational Intelligence and Games (CIG), 2015 IEEE Conference on*, pages 177–184. IEEE, 2015. [8.1](#)
- [86] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015. [8.1](#)
- [87] Heather Kember, Kathryn Connaghan, and Rupal Patel. Inducing speech errors in dysarthria using tongue twisters. *International journal of language & communication disorders*, 52(4):469–478, 2017. [4.2](#)
- [88] Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938. (*document*), [5.1](#), [8.4.5](#), [9.2.1](#)
- [89] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019. [9.9](#)
- [90] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*, 2019. [5.2.1](#)
- [91] Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. Globally Coherent Text Generation with Neural Checklist Models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 329–339, 2016. [1.1.3](#), [8.5](#)
- [92] Nam Wook Kim, Benjamin Bach, Hyejin Im, Sasha Schriber, Markus Gross, and Hanspeter Pfister. Visualizing nonlinear narratives with story curves. *IEEE transactions on visualization and computer graphics*, 24(1):595–604, 2017. [9.7](#)
- [93] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [3.7.4](#), [8.4](#)
- [94] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Anto-

- nio Torralba, and Sanja Fidler. Skip-thought vectors. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3294–3302, 2015. [5.3](#)
- [95] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *Computational linguistics*, 32(4):485–525, 2006. [3.7.1](#)
- [96] Eliza Kitis. Ads—part of our lives: linguistic awareness of powerful advertising. *Word & Image*, 13(3):304–313, 1997. [3.1](#)
- [97] Wei-Jen Ko and Junyi Jessy Li. Assessing discourse relations in language generation from GPT-2. In *Proceedings of the 13th International Conference on Natural Language Generation*, pages 52–59, Dublin, Ireland, December 2020. Association for Computational Linguistics. [\(document\)](#)
- [98] Philipp Koehn. Statistical significance tests for machine translation evaluation. In *EMNLP*, pages 388–395, 2004. [2.7.2](#)
- [99] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007. [3.7.2](#)
- [100] Ranjay Krishna, Yuke Zhu, O. Groth, Justin Johnson, K. Hata, J. Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, D. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32–73, 2016. [6.11](#)
- [101] Sandra Kübler, Ryan McDonald, and Joakim Nivre. Dependency parsing. *Synthesis lectures on human language technologies*, 1(1):1–127, 2009. [2](#)
- [102] Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*, 1998. [1.1.3](#)
- [103] Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. *arXiv preprint arXiv:1603.07771*, 2016. [8.1](#), [8.5](#)

- [104] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019. [1.1.3](#), [6.1](#), [7.1](#)
- [105] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. [9.3.2](#)
- [106] Juncen Li, Robin Jia, He He, and Percy Liang. Delete, retrieve, generate: a simple approach to sentiment and style transfer. 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 1865–1874, 2018. [5.2.1](#)
- [107] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. DailyDialog: A manually labelled multi-turn dialogue dataset. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 986–995, Taipei, Taiwan, November 2017. Asian Federation of Natural Language Processing. ([document](#)), [5](#), [5.1](#), [??](#), [5.1](#), [5.3](#), [??](#), [5.3](#)
- [108] Percy Liang, Michael I Jordan, and Dan Klein. Learning semantic correspondences with less supervision. 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 1-Volume 1*, pages 91–99. Association for Computational Linguistics, 2009. [8.5](#)
- [109] Jen-Wen Liao and Jason S Chang. Computer Generation of Chinese Commentary on Othello Games. In *Proceedings of Rocling III Computational Linguistics Conference III*, pages 393–415, 1990. [8.1](#)
- [110] Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online, November 2020. Association for Computational Linguistics. ([document](#)), [1.1.2](#), [1.2.2](#), [6](#), [6.1](#), [6.3](#), [6.2.2](#), [6.2.3](#), [6.8](#), [6.6.1](#), [6.8](#), [6.12](#), [6.10](#), [6.13](#)
- [111] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text sum-*

marization branches out, pages 74–81, 2004. [5.3](#)

- [112] Chin-Yew Lin and Eduard Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 150–157, 2003. [6.2.2](#)
- [113] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. [8.1](#)
- [114] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1. [6.1](#)
- [115] Wang Ling, Isabel Trancoso, Chris Dyer, and Alan W Black. Character-based neural machine translation. *arXiv:1511.04586*, 2015. [2.2](#)
- [116] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*, 2021. [1.1.3](#)
- [117] Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(7):6418–6425, May 2021. [??](#), [6.6.1](#), [6.7](#)
- [118] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019. [1.1.3](#), [5.5](#)
- [119] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. The Ubuntu Dialogue Corpus: A large Dataset for Research in Unstructured Multi-Turn Dialogue Systems. In *Proceedings of the SIGDIAL 2015 Conference, The 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 2-4 September 2015, Prague, Czech Republic*, pages 285–294. The Association for Computer Linguistics, 2015. [5.3](#)
- [120] Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. Towards an automatic turing test: Learning to evaluate dialogue responses. In *Proceedings of the 55th Annual Meeting of the Association for Compu-*

tational Linguistics (Volume 1: Long Papers), pages 1116–1126, 2017. [5.4](#), [5.5](#), [5.6](#)

- [121] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. [6.7](#)
- [122] Ruotian Luo, Brian Price, Scott Cohen, and Gregory Shakhnarovich. Discriminability objective for training descriptive captions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. [6.4.2](#), [6.11](#)
- [123] Pranava Swaroop Madhyastha, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Mapping unseen words to task-trained embedding spaces. *arXiv preprint arXiv:1510.02387*, 2015. [3.4.2](#)
- [124] Saad Mahamood and Ehud Reiter. Working with clinicians to improve a patient-information NLG system. In *Proceedings of the Seventh International Natural Language Generation Conference*, pages 100–104. Association for Computational Linguistics, 2012. [8.3.1](#)
- [125] Bodhisattwa Prasad Majumder, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Julian McAuley. Like hiking? you probably enjoy nature: Persona-grounded dialog with commonsense expansions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020. [5.5](#)
- [126] Bodhisattwa Prasad Majumder, Taylor Berg-Kirkpatrick, Julian McAuley, and Harsh Jhamtani. Unsupervised enrichment of persona-grounded dialog with background stories. In *ACL*, 2021. [5.5](#)
- [127] Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993. [3.4.1](#)
- [128] R Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. How much do language models copy from their training data? evaluating linguistic novelty in text generation using raven. *arXiv preprint arXiv:2111.09509*, 2021. [2.8](#)
- [129] Jonathan McGovern. Three Tudor Tongue-Twisters. *Notes and Queries*, 68(4):392–393, 12 2021. ISSN 0029-3970. [4.2](#)
- [130] Kathleen R McKeown. Discourse strategies for generating natural-language text. *Artifi-*

cial intelligence, 27(1):1–41, 1985. [1.1.3](#), [1.1.3](#)

- [131] James R Meehan. Tale-spin, an interactive program that writes stories. In *Ijcai*, volume 77, page 9198, 1977. [9.7](#)
- [132] Shikib Mehri and Maxine Eskenazi. Unsupervised evaluation of interactive dialog with dialogpt. *arXiv preprint arXiv:2006.12719*, 2020. [5.5](#)
- [133] Shikib Mehri and Maxine Eskenazi. Usr: An unsupervised and reference free evaluation metric for dialog generation. *arXiv preprint arXiv:2005.00456*, 2020. [5.5](#)
- [134] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. What to talk about and how? selective generation using LSTMs with coarse-to-fine alignment. *arXiv preprint arXiv:1509.00838*, 2015. [1.1.3](#), [8.5](#)
- [135] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer Sentinel Mixture Models. *arXiv preprint arXiv:1609.07843*, 2016. [3.5.3](#), [3.6](#), [3.9](#)
- [136] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842, 2019. [6.7](#)
- [137] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. [3.4.1](#)
- [138] Sebastien Montella, Betty Fabre, Tanguy Urvoy, Johannes Heinecke, and Lina Rojas-Barahona. Denoising pre-training and data augmentation strategies for enhanced RDF verbalization with transformers. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 89–99, Dublin, Ireland (Virtual), 12 2020. Association for Computational Linguistics. [6.7](#)
- [139] Nick Montfort. Ordering events in interactive fiction narratives. In *AAAI Fall Symposium: Intelligent Narrative Technologies*, pages 87–94, 2007. [9.1](#)
- [140] Mary S Morgan. Narrative ordering and explanation. *Studies in History and Philosophy of Science Part A*, 62:86–97, 2017. [9.1](#)
- [141] Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. A corpus and cloze evaluation for deeper understanding of commonsense stories. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 839–849, 2016. [9.1](#)

- [142] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany, August 2016. Association for Computational Linguistics. [9.7](#), [9.10](#)
- [143] Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. LSDSem 2017 shared task: The story cloze test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-level Semantics*, pages 46–51, Valencia, Spain, April 2017. Association for Computational Linguistics. [9.7](#)
- [144] Aakanksha Naik. *Tackling the Long Tail in Language Understanding*. PhD thesis, Carnegie Mellon University Pittsburgh, PA, 2020. [2](#)
- [145] Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*, 2016. [1.1.3](#)
- [146] Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. [8.4.2](#), [8.4.5](#)
- [147] Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003. [3.7.2](#)
- [148] Gözde Özbal and Carlo Strapparava. A computational approach to the automation of creative naming. In *Proceedings of ACL*, pages 703–711. Association for Computational Linguistics, 2012. [2.2](#), [2.8](#)
- [149] Lalchand Pandia, Yan Cong, and Allyson Ettinger. Pragmatic competence of pre-trained language models through the lens of discourse connectives. *arXiv preprint arXiv:2109.12951*, 2021. ([document](#))
- [150] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. [9.3.3](#)
- [151] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for

automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002. [3.7.3](#), [5](#), [5.1](#), [5.3](#), [5.6](#), [6.2.3](#), [8.4](#)

- [152] Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. Pytorch, 2017. [8.4](#)
- [153] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [8.2](#)
- [154] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12(Oct):2825–2830, 2011. [8.4.1](#)
- [155] Hao Peng, Ankur Parikh, Manaal Faruqui, Bhuwan Dhingra, and Dipanjan Das. Text generation with exemplar-based adaptive decoding. 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 2555–2565, 2019. [5.2.1](#)
- [156] Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49, 2018. [1.1.2](#), [9.7](#), [9.9](#)
- [157] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global Vectors for Word Representation. In *EMNLP*, volume 14, pages 1532–1543, 2014. [3.4.1](#)
- [158] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237, 2018. [1.1.3](#)
- [159] Alexandra Petri. Say No to Portmanteaus. *Washington Post - <http://tinyurl.com/kvmep2t>*, 6 2012. [2.1](#)
- [160] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? 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 2463–2473, 2019. [7.1](#)

- [161] Carlos-Eduardo Piñeros. The creation of portmanteaus in the extragrammatical morphology of Spanish. *Probus*, 16(2):203–240, 2004. [2.1](#)
- [162] Andrew Piper, Richard Jean So, and David Bamman. Narrative theory for computational narrative understanding. *arXiv*, 2021. [9](#)
- [163] Edwin JG Pitman. Significance tests which may be applied to samples from any populations. *Supplement to the Journal of the Royal Statistical Society*, 4(1):119–130, 1937. ([document](#)), [6.6](#), [6.7](#)
- [164] Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. Exploring controllable text generation techniques. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 1–14, 2020. [1.1.2](#)
- [165] Shrimai Prabhumoye, Ruslan Salakhutdinov, and Alan W Black. Topological sort for sentence ordering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2783–2792, 2020. [9.6](#)
- [166] Vladimir Propp. *Morphology of the Folktale*, volume 9. University of Texas Press, 2010. [9.1](#)
- [167] Danijela Prošić-Santovac. The use of tongue twisters in efl teaching. *Annual Review of the Faculty of Philosophy/Godisnjak Filozofskog Fakulteta*, 34, 2009. [4.2](#)
- [168] Lianhui Qin, Vered Shwartz, Peter West, Chandra Bhagavatula, Jena D. Hwang, Ronan Le Bras, Antoine Bosselut, and Yejin Choi. Back to the future: Unsupervised backprop-based decoding for counterfactual and abductive commonsense reasoning. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16–20, 2020*, pages 794–805. Association for Computational Linguistics, 2020. [5.2.2](#), [5.8.2](#)
- [169] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018. [1.1.3](#)
- [170] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019. [1.1.3](#), [5.2.1](#), [6.2.3](#), [9.3.2](#)
- [171] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint*

arXiv:2103.00020, 2021. [6.8.1](#)

- [172] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. [6.1](#), [9.3.2](#)
- [173] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. ([document](#)), [1.1.3](#), [7.1](#)
- [174] Preethi Raghavan, Eric Fosler-Lussier, Noémie Elhadad, and Albert M Lai. Cross-narrative temporal ordering of medical events. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 998–1008, 2014. [9.6](#)
- [175] Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, 2019. [9.1](#)
- [176] Attipate Krishnaswami Ramanujan. Three hundred ramayanas: Five examples and three thoughts on translation. *Many Ramayanas: The Diversity of a Narrative Tradition in South Asia*, pages 22–49, 1991. [9.1](#)
- [177] Paul Rayson, Dawn Archer, Alistair Baron, Jonathan Culpeper, and Nicholas Smith. Tagging the Bard: Evaluating the accuracy of a modern POS tagger on Early Modern English corpora. *Article*, 2007. [3.1](#)
- [178] Hans Reichenbach. Elements of symbolic logic. *Dover Publications*, 1947. [9.1](#)
- [179] Daniel Reichert, D. Kaufman, Benjamin Bloxham, H. Chase, and Noémie Elhadad. Cognitive analysis of the summarization of longitudinal patient records. *AMIA ... Annual Symposium proceedings. AMIA Symposium*, 2010:667–71, 2010. [9.6](#)
- [180] Ehud Reiter. Has a consensus nl generation architecture appeared, and is it psycholinguistically plausible? *arXiv preprint cmp-lg/9411032*, 1994. [1.1.3](#), [1.1.3](#)
- [181] Ehud Reiter. NLG vs. templates. *arXiv preprint cmp-lg/9504013*, 1995. [8.4.1](#)
- [182] Ehud Reiter and Anja Belz. An investigation into the validity of some metrics for auto-

- matically evaluating natural language generation systems. *Computational Linguistics*, 35(4):529–558, 2009. [8.4.5](#)
- [183] Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997. [1.1.3](#), [1.1.3](#)
- [184] Ehud Reiter and Robert Dale. Building natural generation systems. *Studies in Natural Language Processing*. Cambridge University Press, 2000. [1.1.3](#), [1.1.3](#)
- [185] Ehud Reiter, Roma Robertson, and Liesl M Osman. Lessons from a failure: Generating tailored smoking cessation letters. *Artificial Intelligence*, 144(1-2):41–58, 2003. [8.5](#)
- [186] Ehud Reiter, Somayajulu G Sripada, and Roma Robertson. Acquiring correct knowledge for natural language generation. *Journal of Artificial Intelligence Research*, 18:491–516, 2003. [8.3.1](#)
- [187] Ehud Reiter, Somayajulu Sripada, Jim Hunter, Jin Yu, and Ian Davy. Choosing words in computer-generated weather forecasts. *Artificial Intelligence*, 167(1-2):137–169, 2005. [8.5](#)
- [188] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. [6.11](#)
- [189] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. [6.4.2](#), [6.11](#)
- [190] M Revathy and K Ravindran. Enhancing effective speaking skills through role play and tongue twisters. *Language in India*, 16(9), 2016. [4.2](#)
- [191] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995. [5.9](#)
- [192] David E Rogers. The influence of panini on leonard bloomfield. *Historiographia linguistica*, 14(1-2):89–138, 1987. [1](#)
- [193] Ronald Rosenfeld. A whole sentence maximum entropy language model. In *1997 IEEE workshop on automatic speech recognition and understanding proceedings*, pages 230–237. IEEE, 1997. [3](#)
- [194] Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685*, 2015. [3.1](#), [3.9](#)

- [195] Aleksander Sadikov, Martin Moina, Matej Guid, Jana Krivec, and Ivan Bratko. Automated chess tutor. In *International Conference on Computers and Games*, pages 13–25. Springer, 2006. [8.1](#)
- [196] Rishiraj Saha Roy, Aishwarya Padmakumar, Guna Prasaad Jeganathan, and Ponnurangam Kumaraguru. Automated Linguistic Personalization of Targeted Marketing Messages Mining User-Generated Text on Social Media. In *16th International Conference on Intelligent Text Processing and Computational Linguistics 2015 (CICLING ’15)*, pages 203–224. Springer International Publishing, 2015. [3.1](#), [3.1](#), [3.9](#)
- [197] Ananya B Sai, Mithun Das Gupta, Mitesh M Khapra, and Mukundhan Srinivasan. Re-evaluating adem: A deeper look at scoring dialogue responses. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6220–6227, 2019. [5.5](#)
- [198] Ananya B Sai, Akash Kumar Mohankumar, Siddhartha Arora, and Mitesh M Khapra. Improving dialog evaluation with a multi-reference adversarial dataset and large scale pre-training. *Transactions of the Association for Computational Linguistics*, 8:810–827, 2020. [5.5](#), [5.7.2](#)
- [199] Ananya B Sai, Akash Kumar Mohankumar, and Mitesh M Khapra. A survey of evaluation metrics used for nlg systems. *arXiv preprint arXiv:2008.12009*, 2020. [5.1](#)
- [200] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A. Smith, and Yejin Choi. ATOMIC: an atlas of machine commonsense for if-then reasoning. In *AAAI*, 2019. [5.2.1](#)
- [201] Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social iqa: Commonsense reasoning about social interactions. 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 4453–4463, 2019. [5.6](#)
- [202] Shiki Sato, Reina Akama, Hiroki Ouchi, Jun Suzuki, and Kentaro Inui. Evaluating dialogue generation systems via response selection. *arXiv preprint arXiv:2004.14302*, 2020. [5.5](#)
- [203] Roger C Schank. Conceptual dependency: A theory of natural language understanding. *Cognitive psychology*, 3(4):552–631, 1972. [1.1.3](#), [5.1](#), [9.1](#)
- [204] Mary J Schleppegrell. Linguistic features of the language of schooling. *Linguistics and education*, 12(4):431–459, 2001. [9.6](#)
- [205] H Andrew Schwartz, Johannes C Eichstaedt, Margaret L Kern, Lukasz Dziurzynski,

Stephanie M Ramones, Megha Agrawal, Achal Shah, Michal Kosinski, David Stillwell, Martin EP Seligman, et al. Personality, gender, and age in the language of social media: The open-vocabulary approach. *PLoS one*, 8(9):e73791, 2013. [3.1](#)

- [206] Abigail See, Peter J Liu, and Christopher D Manning. Get To The Point: Summarization with Pointer-Generator Networks. *arXiv preprint arXiv:1704.04368*, 2017. [1.1.3](#), [2](#), [3.9](#)
- [207] Abigail See, Aneesh Pappu, Rohun Saxena, Akhila Yerukola, and Christopher D. Manning. Do massively pretrained language models make better storytellers? In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 843–861, Hong Kong, China, November 2019. Association for Computational Linguistics. [1.1.3](#), [9.7](#)
- [208] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany, August 2016. Association for Computational Linguistics. [5.3](#)
- [209] Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In *Proceedings of NAACL-HLT*, pages 35–40, 2016. [3.9](#)
- [210] I. Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, 2016. [5.3](#)
- [211] Claude E Shannon. Prediction and entropy of printed English. *Bell Labs Technical Journal*, 30(1):50–64, 1951. [8.4.4](#)
- [212] Katherine E Shaw, Andrew M White, Elliott Moreton, and Fabian Monroe. Emergent faithfulness to morphological and semantic heads in lexical blends. In *Proceedings of the Annual Meetings on Phonology*, volume 1, 2014. [2.1](#)
- [213] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*, 2020. [7.1](#)
- [214] Vered Shwartz and Yejin Choi. Do neural language models overcome reporting bias? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6863–6870, 2020. [6.3.2](#)

- [215] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of Go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016. [8.6](#)
- [216] Aastha Singhal. Would You Interact with a Chatbot that’s Unfriendly? 67% users say no! <https://www.entrepreneur.com/article/339248>, 2021. [3](#)
- [217] Michael R Smith, Ryan S Hintze, and Dan Ventura. Nehovah: A neologism creator nomen ipsum. In *Proceedings of the International Conference on Computational Creativity*, pages 173–181, 2014. [2.2](#)
- [218] Yiping Song, Rui Yan, Xiang Li, Dongyan Zhao, and Ming Zhang. Two are better than one: An ensemble of retrieval-and generation-based dialog systems. *arXiv preprint arXiv:1610.07149*, 2016. [5.5](#)
- [219] Charles Spearman. The proof and measurement of association between two things. *Appleton-Century-Crofts*, 1961. ([document](#)), [5.1](#), [5.3](#)
- [220] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017. [5.2.1](#)
- [221] Balaji Vasan Srinivasan, Rishiraj Saha Roy, Harsh Jhamtani, Natwar Modani, and Niyati Chhaya. Corpus-based automatic text expansion. In *CICLING*, 2017. [3.1](#)
- [222] VentureBeat Staff. People, not tech companies should pick their AI Assistant’s personality. <https://venturebeat.com/2017/10/>, 2021. [3](#)
- [223] Hiroaki Sugiyama, Toyomi Meguro, and Ryuichiro Higashinaka. Automatic evaluation of chat-oriented dialogue systems using large-scale multi-references. In *Advanced Social Interaction with Agents*, pages 15–25. Springer, 2019. [5.1](#)
- [224] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Neural information processing systems*, pages 3104–3112, 2014. [1.1.3](#), [2.2](#), [3.9](#)
- [225] Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. oLMpics-on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758, 2020. [6.1](#)
- [226] Chongyang Tao, Lili Mou, Dongyan Zhao, and Rui Yan. Ruber: An unsupervised method for automatic evaluation of open-domain dialog systems. *arXiv preprint arXiv:1701.03079*,

2017. [5.4](#), [5.5](#), [5.6](#), [5.7.2](#)

- [227] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017. ([document](#)), [1.1.3](#)
- [228] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015. [6.2.3](#), [8.4](#)
- [229] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016. [6.3.1](#), [8](#)
- [230] Oriol Vinyals and Quoc V. Le. A neural Conversational Model. *CoRR*, abs/1506.05869, 2015. [5.3](#)
- [231] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015. [3.9](#)
- [232] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018. [1.1.3](#)
- [233] Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. Retrieval enhanced model for commonsense generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3056–3062, Online, August 2021. Association for Computational Linguistics. [??](#), [6.6.1](#)
- [234] Shuohang Wang and Jing Jiang. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*, 2016. [3.9](#)
- [235] Jason Wei and Kai Zou. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6382–6388, Hong Kong, China, November 2019. Association for Computational Linguistics. [5.5](#)
- [236] R Weide. The CMU pronunciation dictionary, release 0.6. *Carnegie Mellon University*, 1998. [2](#)

- [237] Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. In *International Conference on Learning Representations*, 2019. [1.1.3](#)
- [238] Sarah Wiegreffe and Ana Marasovic. Teach me to explain: A review of datasets for explainable natural language processing. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. [9.1](#)
- [239] John Wieting and Kevin Gimpel. Paranmt-50m: Pushing the limits of paraphrastic sentence embeddings with millions of machine translations. *arXiv preprint arXiv:1711.05732*, 2017. [5.3](#)
- [240] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016. [5.3](#)
- [241] Ursula Wingate. ‘argument!’helping students understand what essay writing is about. *Journal of English for academic purposes*, 11(2):145–154, 2012. [9.6](#)
- [242] Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Challenges in Data-to-Document Generation. *arXiv preprint arXiv:1707.08052*, 2017. [8.4.5](#), [8.5](#)
- [243] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*, 2019. [5.3](#)
- [244] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. *Advances in Neural Information Processing Systems*, 33, 2020. [5.3](#)
- [245] Wei Xu. *Data-driven approaches for paraphrasing across language variations*. PhD thesis, New York University, 2014. [3.2](#), [3.4.1](#)
- [246] Wei Xu, Alan Ritter, William B Dolan, Ralph Grishman, and Colin Cherry. Paraphrasing for style. In *24th International Conference on Computational Linguistics, COLING 2012*, 2012. [3.1](#), [3.2](#), [3.4.1](#), [3.7.2](#), [3.7.2](#), [3.9](#)
- [247] Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. Reference-Aware Language Models. *arXiv preprint arXiv:1611.01628*, 2016. [8.5](#)
- [248] Lei Yu, Phil Blunsom, Chris Dyer, Edward Grefenstette, and Tomas Kociský. The Neural

Noisy Channel. *arXiv:1611.02554*, 2016. [2.3.2](#)

- [249] Tsuta Yuma, Naoki Yoshinaga, and Masashi Toyoda. bleu: Uncertainty-aware automatic evaluation method for open-domain dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 199–206, 2020. [5.5](#)
- [250] Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. [6.7](#)
- [251] Wenyuan Zeng, Wenjie Luo, Sanja Fidler, and Raquel Urtasun. Efficient Summarization with Read-Again and Copy Mechanism. *arXiv preprint arXiv:1611.03382*, 2016. [3.9](#)
- [252] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020. [1.1.3](#)
- [253] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019. [5](#), [5.6](#), [6.2.3](#), [9.3.3](#)
- [254] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with BERT. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. [5.3](#)
- [255] TIANCHENG ZHAO, RAN ZHAO, and MAXINE ESKÉNAZI. Learning Discourse-level Diversity for Neural Dialog Models using Conditional Variational Autoencoders. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 654–664. Association for Computational Linguistics, 2017. [5.3](#)
- [256] TIANCHENG ZHAO, RAN ZHAO, and MAXINE ESKÉNAZI. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 654–664. Association for Computational Linguistics, 2017. [5.1](#)
- [257] Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. Texxygen: A benchmarking platform for text generation models. In *The 41st International*

ACM SIGIR Conference on Research & Development in Information Retrieval, pages 1097–1100, 2018. [5.7.3](#)

- [258] Barret Zoph and Kevin Knight. Multi-source neural translation. *arXiv:1601.00710*, 2016. [2.2](#), [3.9](#)