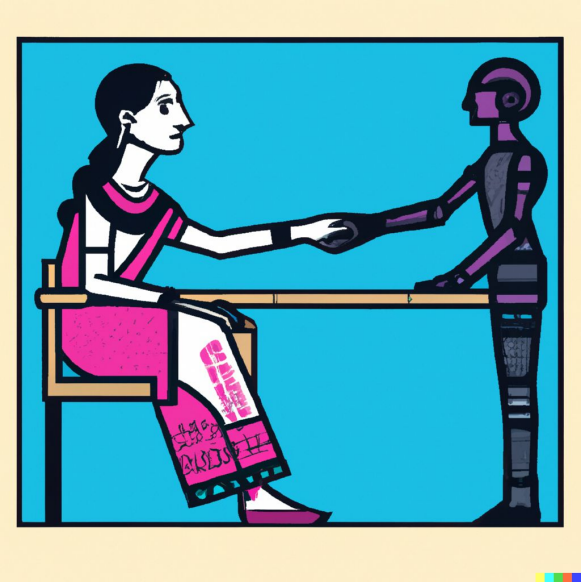
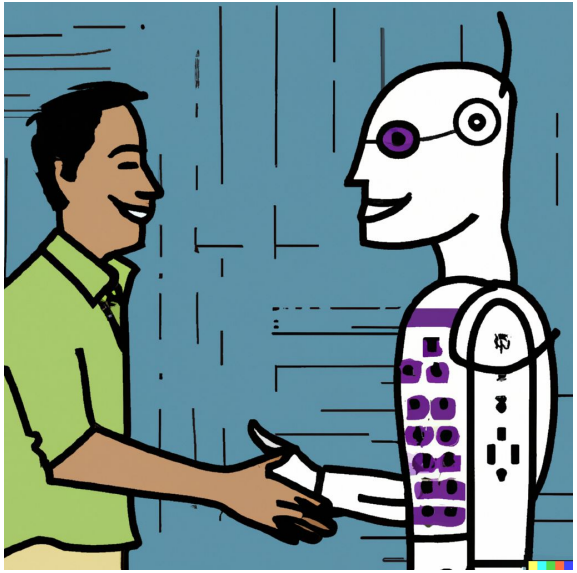
A roadmap for Open Domain NLG

Tuhin Chakrabarty



Committee Members

Smaranda Muresan
Kathy Mckeown
Zhou Yu



A roadmap for Open Domain NLG

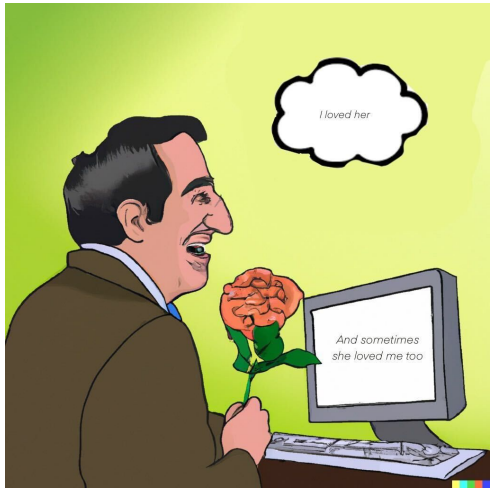
- Open domain NLG with focus on Creativity
- Commonsense in Open Domain NLG
- Open domain NLG Evaluation

Open domain NLG with focus on Creativity

1) What sort of creative open domain NLG problems are we looking at?

   i) Can we generate meaningful sentence level text that is creative and faithful to the input?
   ii) Can we move beyond sentence level creative tasks where global coherence and structure is a challenge ?

2) Can humans benefit from advances in open domain NLG to produce creative and high quality content



Open domain NLG with focus on Creativity

Sentence level Open domain NLG : Figurative Language

- He et al (2019)
- Stowe et al (2021)
- Tian et al (2021)
- Oprea et al (2022)

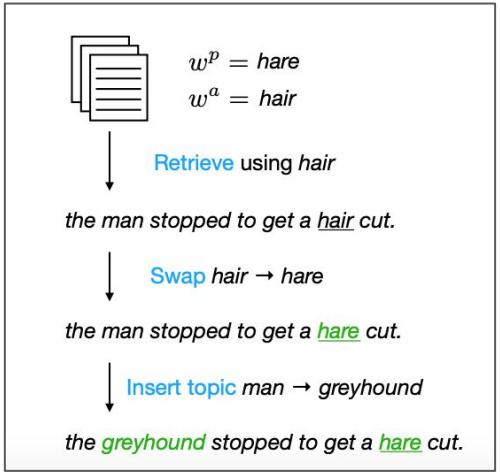
Beyond Sentence level Open domain NLG

- Ghazvininejad et al (2016)
- Van De Cruys (2020)
- Fan et al (2019)
- Fan et al (2019)

Human AI Collaborative Writing

- Lee et al (2022)
- Padmakumar and He (2022)

Pun Generation He et al (2019)



Goal: Unsupervised Pun Generation using Surprisal

Data: SemEval task 7 (Doogan et al., 2017)

Method:

- Proposes Global Local Surprisal Principle and shows it correlates With funniness better than prior metric
- Retrieve seed sentence with alternative word w^a .Replace w^a with w^p [Local Surprisal] and replace first noun/pronoun in sentence with a type consistent related topic word [Global Local Contrast]
- Smooth edited sentence to ensure grammatical correctness

- Local surprisal S_{local}
  - $-log\frac{p(like\ I\ 've\ dyed\ a\ little\ inside)}{p(like\ I\ 've\ died\ a\ little\ inside)}$
- Global surprisal S_{global}
  - $-log\frac{p(Yesterday\ I\ \ldots like\ I\ 've\ dyed\ a\ little\ inside.)}{p(Yesterday\ I\ \ldots like\ I\ 've\ died\ a\ little\ inside.)}$
- Local-global surprisal ratio: $S_{ratio} \stackrel{def}{=} \frac{S_{local}}{S_{global}}$ (the larger the better)

(+) Generation incorporating linguistic theory of humor without requiring training data

(-) Prone to grammatical error/ fluency issues. Pun word doesn't fit in local context due to different POS with alternative word **(wait – weight)**

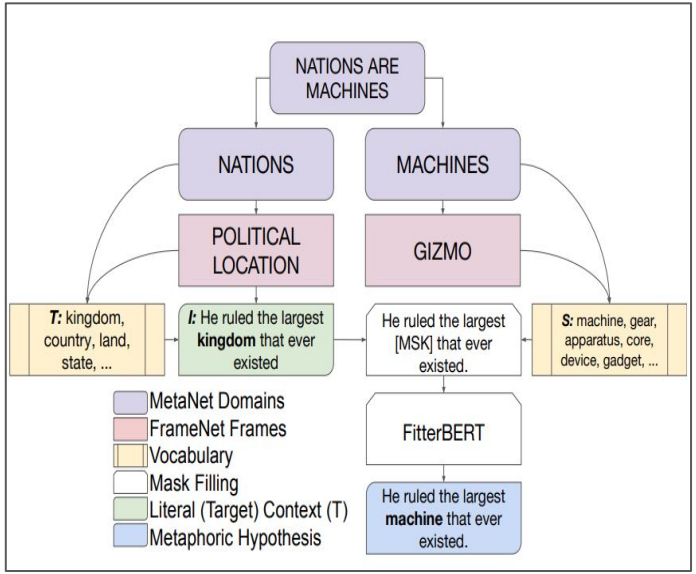
Exploring Metaphoric Paraphrase Generation Stowe et al (2021)

Goal: Metaphor Generation using Conceptual Metaphor Theory

Data: FrameNet corpus & MetaNet

Method:

- Create parallel data in a semi supervised fashion using MetaNet and FrameNet Corpus
- Fine-tuning using T5 with Control Codes



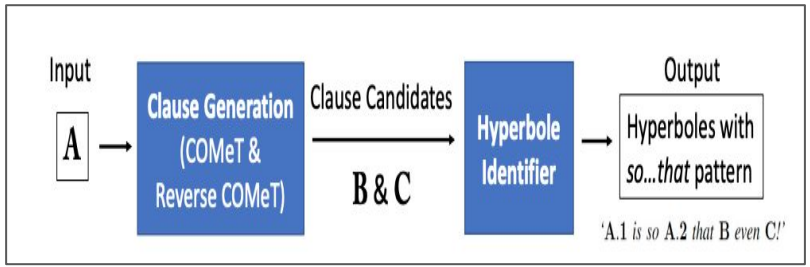
(+) Comparing Free vs Controlled Metaphor Generation, Use of MetaNet, Automatically created Parallel data

(-) Results on Metaphoricity still worse than prior work by Stowe et al (2021)

HypoGen: Hyperbole Generation Tian et al (2021)

Goal: Unsupervised Hyperbole Generation from Literal using Commonsense & Counterfactual

Data: Reddit, Concept Net



Method:

- Create hyperbole of pattern A.1 is so A.2 that B even C where the literal prompt is (A), the subject in the clause is (B) and the predicate (verbal phrase) in the clause is (C)
- Leverages Commonsense Knowledge in particular relation *RelatedTo* and *HasProperty* for Common sense and *NotCapableOf* for Counterfactual

  [The party is so lit] even the [wardrobe] is [dancing]!
- Reranking based on a classifier trained on annotated hyperbole data from Reddit.

(+) Using commonsense and counterfactual in a theoretically principled way McCarthy and Carter (2004) without parallel data

(-) Follows only one pattern and a paraphrase model used to get rid of the pattern leads to additional hallucinations

Should a chatbot be sarcastic ? Oprea et al (2022)

Goal: Unsupervised Sarcasm Generation using Implicit Display Theory

Data: Short Tweets with actions Wilson and Mihalcea (2019)

Method:

- Starts with expectation Q that is part of the ironic environment that negates the action with input U_{in}.

Algorithm 1: Generate sarcastic response

input: utterance U_{in};
ironic environment
  Let $Q := \neg P$ be the failed expectation;
implicit display
  Choose an if-then relation type τ from *xNeed*, *xAttr*, *xReact*, and *xEffect*;
  Let $\alpha = \text{COMET}(U_{\text{in}}, \tau)$;
return response U_{out} that expresses *emotion*$(\neg\alpha)$;

- Uses Pragmatic Insincerity where they infer the relation object α from U_{in} using COMET relations xAttr, xReact xEffect and xNeed and construct an utterance that expresses ¬α

(+) Use of Implicit Display Theory instead of linguistic incongruity and ability to handle all types of input sentiment

(-) Pattern based and lot of heuristics, no controllability in terms of hallucinations, and finally poor sarcasticness measured by human evaluators

Open domain NLG with focus on Creativity

Sentence level Open domain NLG : Figurative Language

- He et al (2019)
- Stowe et al (2021)
- Tian et al (2021)
- Oprea et al (2022)

Beyond Sentence level Open domain NLG

- Ghazvininejad et al (2016)
- Van De Cruys (2020)
- Fan et al (2019)
- Fan et al (2019)

Human AI Collaborative Writing

- Lee et al (2020)
- Padmakumar and He (2022)

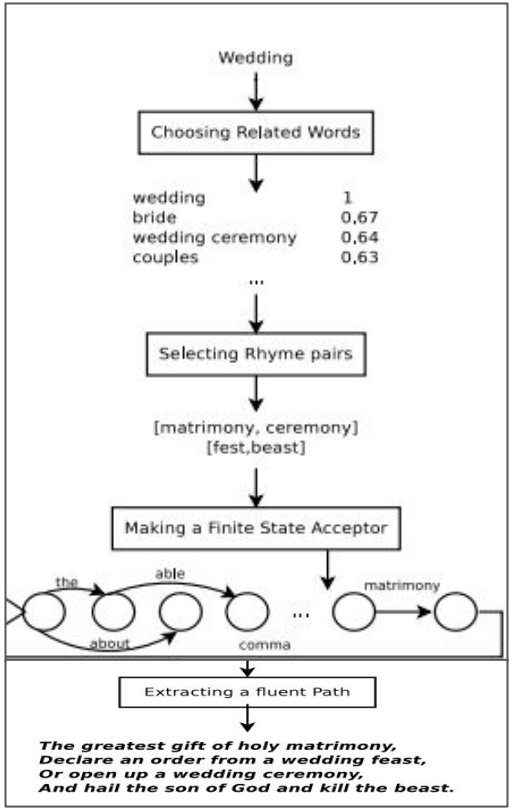
Generating Topical Poetry Ghazvininejad et al (2016)

Goal: Sonnet Generation on any given Topic focusing on rhyme

Data: Wikipedia, Song Corpus (http://www.mldb.org/)

Method:

- Starts with choosing related words to the topic using Word2Vec for semantic coherence followed by clustering related words based on rhyming pattern and creating FSA with arcs as words and states that record line number and syllable count
- Uses an RNN based Language model trained on a corpus of songs for syntactic fluency. During decoding with the LM, employ a beam search that is further guided by the FSA for imposing rhyme and meter



(+) Unsupervised approach, clever use of FSA for rhyme and syllables for Sonnet

(-) Computationally pretty expensive , lack of global discourse coherence, figure of speech / imagery

Automatic Poetry Generation from Prosaic Text Van de Cruys (2020)

Goal: Poetry Generation from Prosaic Text

Data: CommonCrawl corpus

Method:

- Uses a GRU to generate next verse in poetry conditioned on an input verse
  Trained to generate the last word first to incorporate rhyme
- Modifies probability distribution of GRU during decoding to incorporate rhyme constraint using Wiktionary and topical constraint using latent semantic model based on nonnegative matrix factorization

At the moment it seems almost impossible
Yet life is neither good nor evil
The divine mind and soul is immortal
In other words, the soul is never ill

So far, it has barely lost its youthful look
But no man is ever too young for the rest
He thought deeply, and yet his heart shook
At that moment he seemed utterly possessed

(+) Incorporation of Rhyme and Topical Constraint in decoding to generate coherent verses in En/Fr

(-) lack of global discourse coherence, figure of speech / imagery, misses the language of poetry due to domain difference in training data

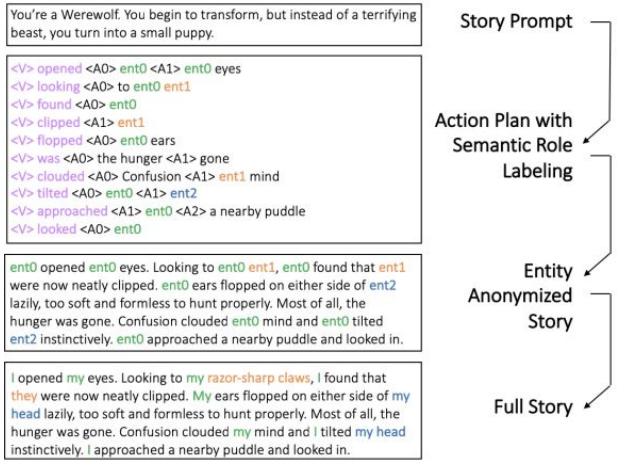
Strategies for Structuring Story Generation Fan et al (2019)

Goal: Story Generation by abstracting over actions and entities

Data: WRITINGPROMPTS (Fan et al., 2018)

Method:

- First generate the predicate-argument structure of the story by generating a sequence of verbs and arguments. This representation is more structured than free text, making it easier for the model learn dependencies across events
- Generates a surface realization of the predicate-argument structure, and finally replaces the entity placeholders with context-sensitive names and references



(+) Generating action sequence makes the model less prone to generate generic events and instead diverse verbs

(-) Still lacking global coherence, heavy lifting for encoder to handle event choice and arrangement , replacing entities and ensuring consistency in coreference is still difficult due to pipelined models

ELI5: Long Form Question Answering Fan et al (2019)

Goal: Long-form generative question answering

Data: r/ELI5 (Fan et al., 2018)

Method:

- Scrape ELI5 subreddit for questions along with long form answers having upvotes and consequently retrieve paragraphs from Common Crawl based on relevance to question called Support Document
- Trains seq2seq models with Question and Support Document going as input to encoder and decoder generating answer auto-regressively

Question: How do Jellyfish function without brains or nervous systems? [...] (60 words)

Answer: Jellyfish may not have a brain, but they have a rough nervous system and innate behaviours. However, they are very simple creatures. They're invertebrate: creatures without a backbone. Most jellyfish have really short life spans. Sometimes just a couple of hours. [...] As their name implies, they are largely composed of basically jelly inside a thin membrane. They're over 95% water. (327 words)

(+) Extremely Challenging task, Good Data Source, Compelling results compared to extractive / retrieval baselines

(-) Poor Evaluation heavily reliant on Rouge Score and no focus on factuality

Open domain NLG with focus on Creativity

Sentence level Open domain NLG : Figurative Language

- He et al (2019)
- Stowe et al (2021)
- Tian et al (2021)
- Oprea et al (2022)

Beyond Sentence level Open domain NLG

- Ghazvininejad et al (2016)
- Van De Cruys (2020)
- Fan et al (2019)
- Fan et al (2019)

Human AI Collaborative Writing

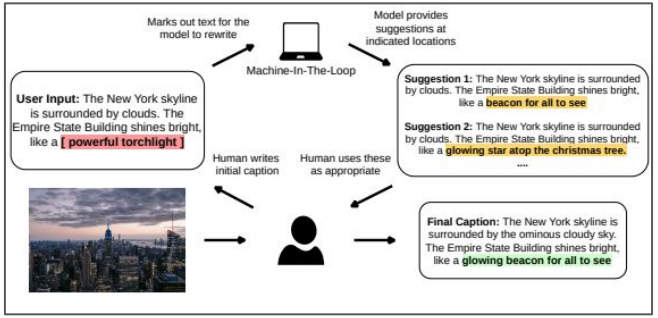
- Lee et al (2020)
- Padmakumar and He (2022)

Machine-in-the-Loop Rewriting for Creative Image Captioning Padmakumar et al (2022)

Goal: Human AI Collaborative Image Captioning

Data: Steen et al. (2010) ; Bostan et al. (2020); Gordon et al. (2015) Mohammed et al (2016) Text from news, fiction, headlines, product reviews

Method:



- Takes target sentences rich in creative imagery such as metaphors and similes and masks the creative part and uses BART to infill it with a replacement that results in a literal source sentence

- Uses this pseudo parallel corpus of 42000 sentences to finetune a BART model which is then used by humans. In particular they are asked to first write a caption from Déjà Captions dataset (Chen et al., 2015) and then iteratively rewrite using fine-tuned model

(+) Nice results showing interaction improves creative writing capabilities for humans

(-) User study shows their system only useful for skilled users
(-) No control for content drift in infilling

CoAuthor: Human-AI Collaborative Writing Dataset Lee et al (2022)

Goal: Reason about GPT-3's language capabilities (ability to generate fluent text), ideation capabilities (ability to generate new ideas), and collaboration capabilities (ability to work jointly with writers)

Data: Prompts from r/WritingPrompt for stories and New York Times for Argumentative text

Method:

- Hires 63 skilled writers on Amazon Mechanical Turk to write creative stories or argumentative text with the help of GPT3. Writers are provided with a UI and a prompt and they can then accept/ edit / delete suggestions from GPT3 to shape up the overall content

- Measuring grammaticality using language tool, diverse vocabulary in terms of distinct ngrams and new ideation capability using % of suggested named entities being used in the final text and also introduces equality and mutuality metric to measure collaboration

(+) One of first studies quantifying human AI collaboration for creative writing using LLMs focusing on longer text
(-) Collaboration measured through automatic metrics mostly and no objective comparison of Human only vs Human and GPT3

A roadmap for Open Domain NLG

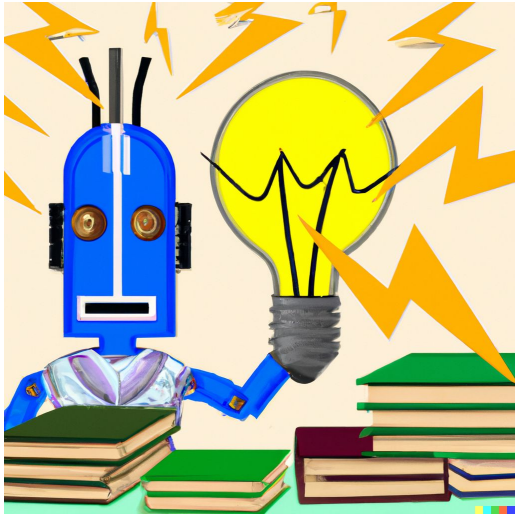
- Open domain NLG with focus on Creativity
- Commonsense in Open Domain NLG
- Open domain NLG Evaluation

Commonsense in Open Domain NLG

1) What is the importance of Common sense in open domain NLG?

   i) What are the tasks that benefit heavily from commonsense ?
   ii) How do these tasks incorporate common sense ?

2) What are some benchmarks designed to measure common sense in open domain NLG ?



Commonsense in Open Domain NLG

Conversation Context

- Ghazvininejad et al (2018)
- Majumder et al (2020)
- Zhou et al (2022)

Beyond Conversation Context

- Ammanabrolu et al (2020)
- Majumder et al (2022)

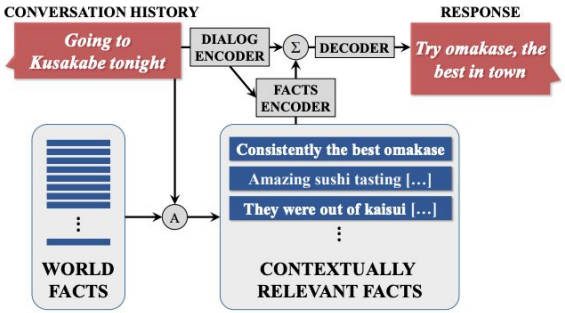
Generative Common sense

- Gabriel et al (2020)

Benchmarks

- Bhagavatula et al (2020)
- Lin et al (2020)

A Knowledge-Grounded Neural Conversation Model Ghazvininejad et al (2018)



Goal: Knowledge Grounded Response Generation

Data: Twitter, FourSquare

Method:

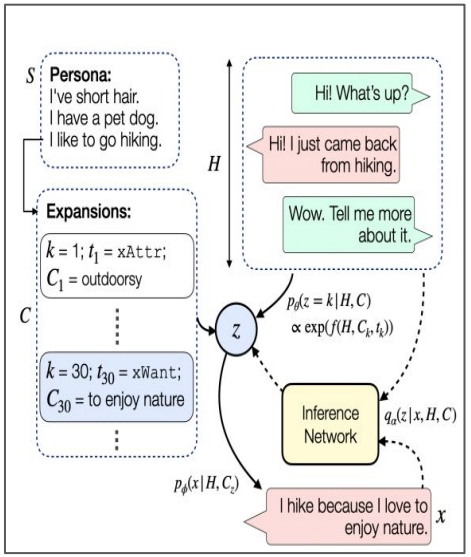
- Retrieves 23M 3 turn tweets and 1.1M Foursquare Tips. Filters approx 1M 2 turn tweets having entity overlap with Foursquare Tips in at least one of the turns.

- Trains two tasks in a multitask fashion NONFACTS[23M tweets P(S,R)] and FACTS [1M grounded tweets P({f_1, ..., f_n, S}, R) where S is dialog input, R is response and f_1, ..., f_n are retrieved facts

(+) Nice approach of multitask learning of non-factual and factual response generation , incorporation of memory network for fact encoding

(-) Dialogue data is limited to 2-3 turn
(-) Mostly restricted to factual knowledge from a single domain without any analysis of overlap of Foursquare tip knowledge in train test
(-) No empirical measure of retrieval quality such as Recall@K and unclear if generations are grounded in retrieval

Like hiking? You probably enjoy nature: Persona-grounded Dialog with Commonsense Expansions Majumder et al (2020)



Goal: Expand persona using existing commonsense knowledge bases and paraphrasing resources to imbue dialog models with access to an expanded and richer set of persona descriptions to improve response generation

Data: PersonaChat (Zhang et al., 2018)

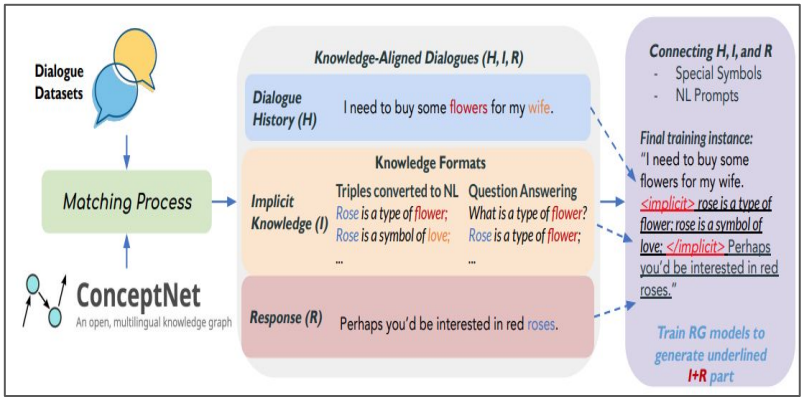
Method:

- Expands persona using commonsense and for an observed dialog history H, a relevant persona is modeled using a latent discrete variable z. Given H a persona sentence C_z is sampled from a prior network $p_\theta(z=k|H,C)$
- A grounded response x is sampled w.r.t C_z and H from a generator network $p_\varphi(x|H, C_z)$.

(+) Neat idea of incorporating ATOMIC for common sense inference as a latent variable in response generation

(-) Automatic evaluation uses BLEU and perplexity none of which correlates with quality of response
(-) Additionally prior network can sample only one persona sentence. An interesting response might combine multiple persona sentences

Think Before You Speak: Explicitly Generating Implicit Commonsense Knowledge for Response Generation Zhou et al (2022)



Goal: Think-Before-Speaking (TBS), a generative approach to first externalize implicit commonsense knowledge (think) and use this knowledge to generate responses (speak)

Data: SocialIQA-prompted Common sense Dialogues (Zhou et al., 2021) Daily Dialog (Li et al., 2017), Empathetic Dialogues (Rashkin et al., 2019), MuTual (Cui et al., 2020)

Method:

- Matches with semantically closest triples in ConceptNet to pair each dialogue with appropriate implicit knowledge

- First train the model to generate just the knowledge I based on H′ , $P_\theta(I|H')$, and then train it to generate R based on both I and H′ , $P_\theta(R|H', I)$

(+) Neat idea of aligning concept centric implicit commonsense to dialogue data during training that leads to generalization during inference and better quality responses

(-) Matching procedures do not concern multi-hop triples that might be needed for complex reasoning chains.ConceptNet mostly contains taxonomic and lexical knowledge limiting the diversity of generated knowledge from TBS models

Commonsense in Open Domain NLG

Conversation Context

- Ghazvininejad et al (2018)
- Majumder et al (2020)
- Zhou et al (2022)

Beyond Conversation Context

- Ammanabrolu et al (2020)
- Majumder et al (2022)

Generative Common sense

- Gabriel et al (2020)

Benchmarks

- Bhagavatula et al (2020)
- Lin et al (2020)

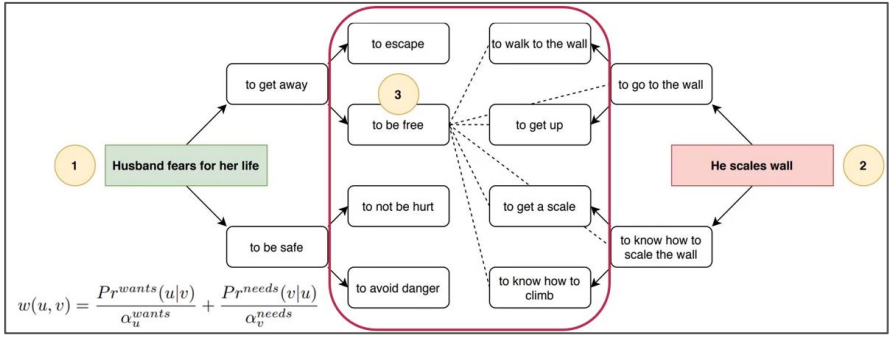
Automated Storytelling via Causal, Commonsense Plot Ordering Ammanabrolu et al (2020)

Goal: Narrative generation that operationalizes Plot Infilling through Causality

Data: Mystery stories and fairy tales Ammanabrolu et al. (2020a)

Method:

- First extracts a set of high level plot points from a given Textual story plot S using Coreference resolution and Information Extraction using Open IE



- Generates a sub-graph of events that go between each high level plot point by recursively querying *xWant* and *xNeed* relations from COMET acting as soft causal relations. The final story is obtained by walking the overall plot graph generated by joining each generated sub-graph

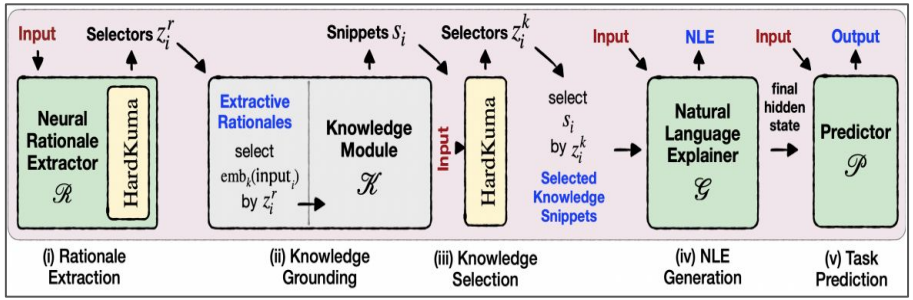
(+) Commonsense Reasoning as a way to incorporate soft causal relations for better event planning in story generation

(-) Pretty much follows a standard template , repetitive, experiments shown on only 5 sentence stories and not compared against more challenging baselines

Knowledge-Grounded Self-Rationalization via Extractive and Natural Language Explanations (Majumder et al 2022)

Goal: Natural Language Explanation Generation

Data: e-SNLI (Camburu et al., 2018), COSe (Rajani et al., 2019) ComVE (Wang et al., 2019)



Method:

- Selection of ERs using a series of latent variables realized by a HardKuma distribution followed by knowledge inference from ER using generative Commonsense models

- Relevant knowledge snippets are selected similarly as previous step. Vector representation of selected knowledge snippets are concatenated and passed along with input to the NLE generator where the explanation is decoded followed by label prediction

(+) Nice incorporation of Commonsense Reasoning incorporated into NLE across various modalities and benchmarks showing its efficacy

(-) Reliance on COMET and hence might not transfer to more challenging tasks or longer contexts where COMET fails

Commonsense in Open Domain NLG

Conversation Context

- Ghazvininejad et al (2018)
- Majumder et al (2020)
- Zhou et al (2022)

Beyond Conversation Context

- Ammanabrolu et al (2020)
- Majumder et al (2022)

Generative Common sense

- Gabriel et al (2020)

Benchmarks

- Bhagavatula et al (2020)
- Lin et al (2020)

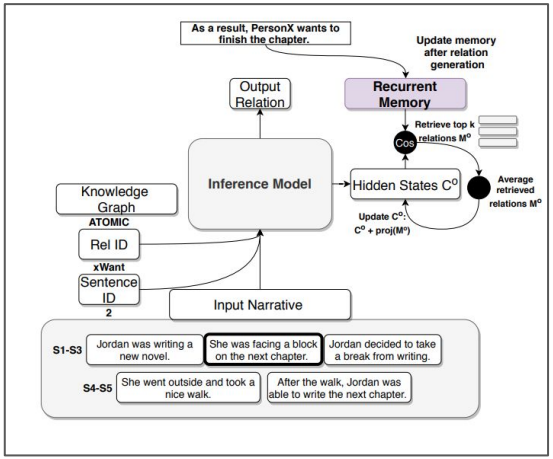
Paragraph-level Commonsense Transformers with Recurrent Memory (Gabriel et al 2020)

Goal: Discourse Aware Commonsense Knowledge Generation

Data: ROC Stories (Mostafazadeh et al. 2016)

Method:

- To create data uses common sense inference from COMET given each sentence S_i in a narrative and filter the inferences that have a low coherence with the overall narrative based on the cross entropy of the story tokens



- Trains a model augmented with recurrent memory that allows them to explicitly incorporate episodic knowledge along with semantic knowledge drawn from the context

(+) Neat incorporation of Recurrent Memory to keep track of previously generated inferences, recalling the model's prior decisions during test time and previously extracted relations from their distant supervision during training

(-) The discourse aware model doesn't work beyond 5 sentences which is a major limitation primarily because for longer discourse the memory size needs to be a lot bigger

Commonsense in Open Domain NLG

Conversation Context

- Ghazvininejad et al (2018)
- Majumder et al (2020)
- Zhou et al (2022)

Beyond Conversation Context

- Ammanabrolu et al (2020)
- Majumder et al (2022)

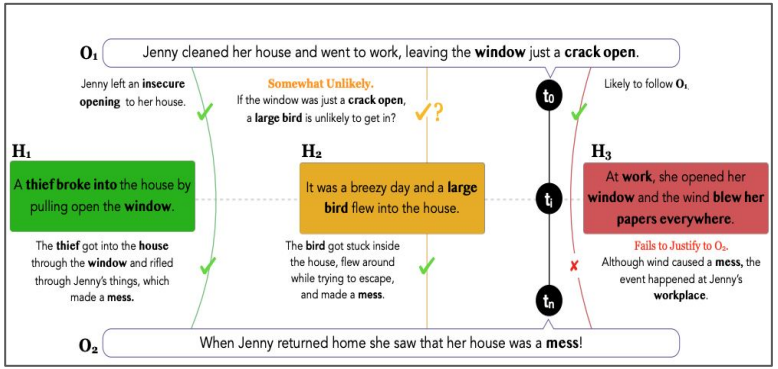
Generative Common sense

- Gabriel et al (2020)

Benchmarks

- Bhagavatula et al (2020)
- Lin et al (2020)

Abductive Common sense Reasoning (Bhagavatula et al 2020)



Goal: αNLG is the task of generating a valid hypothesis h^+ given the two observations O1 and O2.

Data: ROCStories dataset (Mostafazadeh et al., 2016)

Method:

- O1 and O2 are collected from ROCStories and crowdworkers are hired to write multiple plausible and implausible hypotheses for each <O1, O2> pair and then an adversarial filtering algorithm is applied to retain one challenging pair of hypotheses that are hard to distinguish between

- Trains a GPT2 model conditioned on the tokens of the two observations O1 and O2 along with integrating information from COMeT (18 ~ 9 relations from O1,O2)

(+) An important benchmark that focuses on a critical capability in human reasoning

(-) Still works on more easier form of abductive reasoning from narratives. No proof of generalization to more complex forms of abduction such as arguments

CommonGen: A Constrained Text Generation Challenge for Generative Commonsense Reasoning (Lin et al 2020)

Goal: Given a set of common concepts the task is to generate a coherent sentence describing an everyday scenario using these concepts

Data: Flickr30k (Young et al., 2014), MSCOCO (Lin et al., 2014), Conceptual Captions (Sharma et al., 2018), LSMDC (Rohrbach et al., 2017), ActivityNet (Krishna et al., 2017), and VATEX (Wang et al., 2019b)

Method:

- To create data uses visually grounded sentences from image captioning datasets such that words in sentences can be matched to the concept vocabulary of ConceptNet

- Finetunes several generative language models and shows that there is large gap between machine and human performance (31.6% v.s. 63.5% in SPICE metric).

(+) An important benchmark that focuses on relational reasoning with latent commonsense knowledge leading to compositional generalization.; learned generative commonsense reasoning capability can be transferred to improve downstream tasks such as CommonsenseQA

(-) Focuses more on sentence level generation which isn't as hard for language models. Finally training set is still constructed from image captions which might not capture complex reasoning patterns prevalent in text

A roadmap for Open Domain NLG

- Open domain NLG with focus on Creativity
- Commonsense in Open Domain NLG
- Open domain NLG Evaluation

Open domain NLG evaluation

1) What are the challenges in evaluation of open domain NLG?

   i) Are existing evaluation metrics flawed?
   ii) Can humans judge the quality better or its an illusion?

2) What are the reasons why neural NLG models are far from perfect?

Open domain NLG evaluation

Hallucination in Neural NLG models

- Rashkin et al (2022)
- Dziri et al (2022)
- Shuster et al (2021)

Evaluation of Long Text Generation

- Clark et al (2021)
- Dou et al (2022)
- Krishna et al (2021)

Difficulties in Human Evaluation

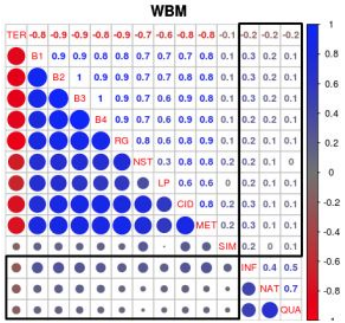
- Karpinska et al (2020)
- Smith et al (2022)

Challenges in Text Generation Evaluation

- Novikova et al (2017)

Why We Need New Evaluation Metrics for NLG (Novikova et al 2017)

MR: inform(name=X, area=X, pricerange=moderate, type=restaurant)
Reference: *"X is a moderately priced restaurant in X."*



Goal: Show how state-of-the-art automatic metrics weakly reflect human judgements of system outputs as generated by data-driven, end-to-end NLG

Data: SFHOTEL & SFREST (Wen et al., 2015) ;BAGEL (Mairesse et al., 2010)

Method:

- Investigate a wide range of metrics, including state-of-the-art word-based and novel grammar-based ones, and demonstrate that they only weakly reflect human judgements of system outputs as generated by data-driven, end-to-end NLG systems.

- Show that metric performance is data and system-specific and that automatic metrics are particularly weak in distinguishing outputs of medium and good quality

(+) One of the first kind of a systematic study composed on a wide range of automatic evaluation metrics for NLG paving the way for better metric design and more focus on human evaluation

(-) Done only on one type of task i.e meaning representation to text generation

(-) Human evaluation done on likert scale which has found criticism in literature

Open domain NLG evaluation

Hallucination in Neural NLG models

- Rashkin et al (2022)
- Dziri et al (2022)
- Shuster et al (2021)

Evaluation of Long Text Generation

- Clark et al (2021)
- Dou et al (2022)
- Krishna et al (2021)

Difficulties in Human Evaluation

- Karpinska et al (2020)
- Smith et al (2022)

- Novikova et al (2017)

Challenges in Text Generation Evaluation

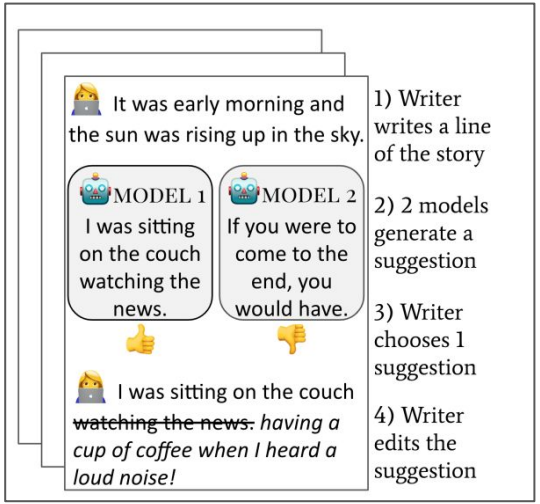
Choose Your Own Adventure: Paired Suggestions in Collaborative Writing for Evaluating Story Generation Models (Clark et al 2021)

Goal: Collaborative writing setup for pairwise model evaluation for story generation

Data: Writing Prompts (Fan et al., 2018)

Method:

- Recruits crowdworkers on AMT to engage in a 5 turn story writing which starts with a human writing a sentence and then getting suggestions from 2 models and then choosing the best one and potentially editing it
- Through this setup tests if a model is better at generating story suggestions than a baseline model as well as usefulness of the models' suggestions and how models' generated texts compare to human-authored text

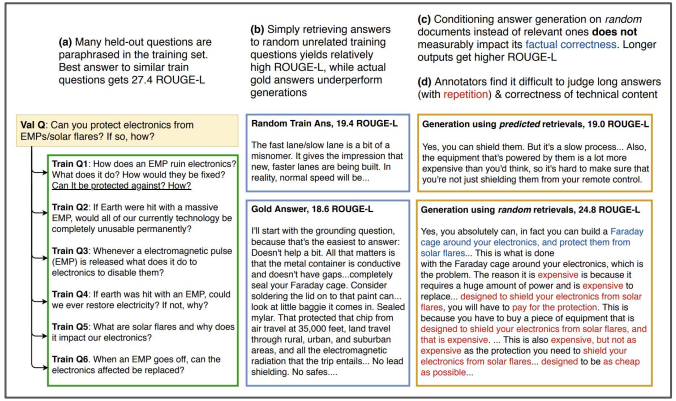


(+) Useful in terms of measuring quality of of autoregressive models in terms how capable they are in handling real world human prompts as well how close the suggestions are to human acceptability

(-) Still involves a lot of subjective judgement, little details about who was hired to collaborate with models (are these experts or untrained workers considering collaborating with a model requires skills?)

(-) Uses average sentence length, distinct n-grams or nouns and verbs to measure closeness to how humans write stories which isn't accurate

Hurdles to Progress in Long-form Question Answering (Krishna et al 2021)



Goal: Highlighting problems in long form question answering dataset and evaluation and how impressive benchmark based on poor metric choices can be misleading

Data: ELI5 (Fan et al., 2019)

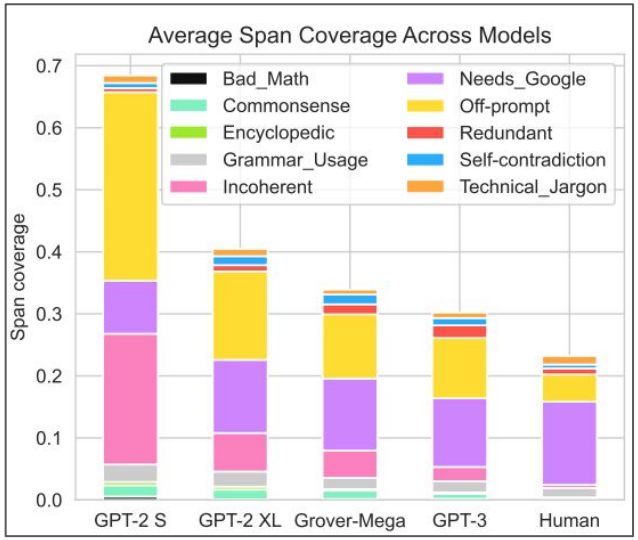
Method:

- Trains a model for long form QA using a retriever based on contrastive REALM and a generator based on Routing Transformer that uses Sparse attention to handle longer sequence for text conditioning
- Models achieves SOTA on KILT benchmark but deeper investigation reveals that generations are not grounded in retrieved data and that simply adding random retrieved passages doesn't hurt ROUGE-L
- Longer generations are favored by ROUGE-L even more than gold answers. Additionally factuality in generation comes from train test overlap in data where paraphrase of training question exist in test

(+) Excellent analysis showing the drawbacks of using ROUGE-L for open domain NLG

(-) While it highlights several pressing issues with evaluation it doesn't propose any new ways to automatically evaluate long form QA

Is GPT-3 Text Indistinguishable from Human Text? SCARECROW: A Framework for Scrutinizing Machine Text (Dao et al 2022)



Goal: Propose a new framework called SCARECROW for scrutinizing machine text via crowd annotation

Data: 13k annotations of 1.3k paragraphs, amassing 41k spans labeled with error type based on news articles found in the Common Crawl

Method:

- Recruits trained crowdworkers to select minimal span in model generated paragraphs that contains error and then label it among one of the 10 categories spanning across *language errors*, *factual errors*, and *reader issues* and provide natural language explanation for it
- Frame this problem as a span classification task using roberta-large and reports per-token precision, recall, and F1 score per error category on test set.

(+) A large scale dataset to study systematic errors in NLG output

(-) Explanations are never used as a part of span classification

(-) Lack of downstream application that shows if knowing these errors in advance can prevent an NLG model to generate these during decoding

Open domain NLG evaluation

- Rashkin et al (2022)
- Dziri et al (2022)
- Shuster et al (2021)

Hallucination in Neural NLG models

- Clark et al (2021)
- Dou et al (2022)
- Krishna et al (2021)

Evaluation of Long Text Generation

- Karpinska et al (2020)
- Smith et al (2022)

Difficulties in Human Evaluation

- Novikova et al (2017)

Challenges in Text Generation Evaluation

The Perils of Using Mechanical Turk to Evaluate Open-Ended Text Generation (Karpinska et al 2022)

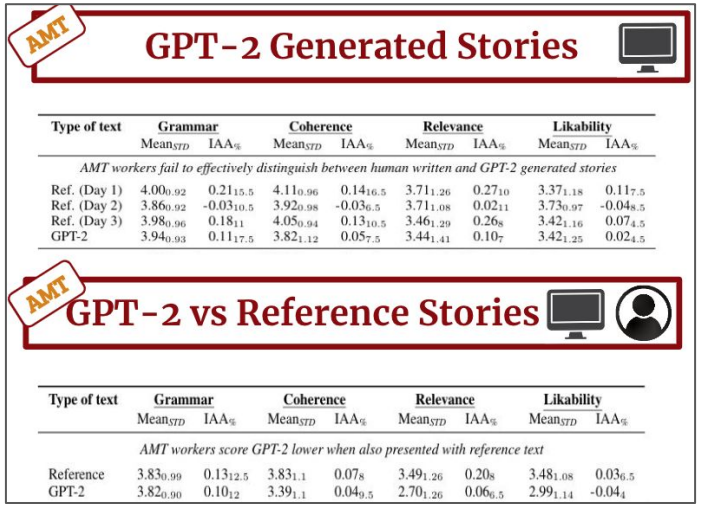
Goal: Exposing problems with AMT when evaluating open ended generation with a focus on story generation

Data: r/WritingPrompts (Fan et al 2019)

Method:

- Recruits workers on AMT for evaluating model & human written stories individually as well as pairwise using likert scores and showcase vulnerabilities in the areas of poor English skills, Short time taken to complete a hit and variance in judgements across days

- Conducts experiments with different qualification groups and finds that workers prefers model generated stories over human generated ones when evaluated separately unlike English teachers. However preference changes when evaluated pairwise



Type of text	Grammar		Coherence		Relevance		Likability	
	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$
AMT workers fail to effectively distinguish between human written and GPT-2 generated stories								
Ref. (Day 1)	$4.00_{0.92}$	$0.21_{15.5}$	$4.11_{0.96}$	$0.14_{16.5}$	$3.71_{1.26}$	0.27_{10}	$3.37_{1.18}$	$0.11_{7.5}$
Ref. (Day 2)	$3.86_{0.92}$	$-0.03_{10.5}$	$3.92_{0.98}$	$-0.03_{6.5}$	$3.71_{1.08}$	0.02_{11}	$3.73_{0.97}$	$-0.04_{8.5}$
Ref. (Day 3)	$3.98_{0.96}$	0.18_{11}	$4.05_{0.94}$	$0.13_{10.5}$	$3.46_{1.29}$	0.26_{8}	$3.42_{1.16}$	$0.07_{4.5}$
GPT-2	$3.94_{0.93}$	$0.11_{17.5}$	$3.82_{1.12}$	$0.05_{7.5}$	$3.44_{1.41}$	0.10_{7}	$3.42_{1.25}$	$0.02_{4.5}$

Type of text	Grammar		Coherence		Relevance		Likability	
	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$	$Mean_{STD}$	$IAA_{\%}$
AMT workers score GPT-2 lower when also presented with reference text								
Reference	$3.83_{0.99}$	$0.13_{12.5}$	$3.83_{1.1}$	0.07_{8}	$3.49_{1.26}$	0.20_{8}	$3.48_{1.08}$	$0.03_{6.5}$
GPT-2	$3.82_{0.90}$	0.10_{12}	$3.39_{1.1}$	$0.04_{9.5}$	$2.70_{1.26}$	$0.06_{6.5}$	$2.99_{1.14}$	-0.04_{4}

(+) Shows the problems with Amazon Mechanical Turk when evaluating open ended text through highlighting several factors

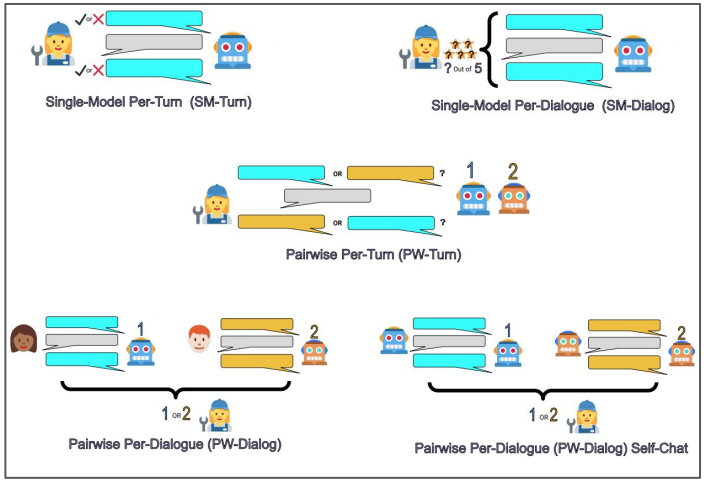
(-) Focuses only on story generation which is itself is a time consuming job and their qualification still doesn't guarantee for better quality

Human Evaluation of Conversations is an Open Problem (Smith et al 2022)

Goal: Comparing the sensitivity of various methods for evaluating dialogue agents

Data: BlenderBot

Method:



- Compare dialog models at a per-turn vs whole conversation level in terms of *Preference, Interestingness* and *Humanlike* metrics by varying them by response length, model size and fine-tuning strategy

- Conducts experiments which shows per-turn evaluation technique may be more suitable for pairs of models that differ in their ability to reply sensibly in a way that is easily detectable by their partner but that a whole-conversation technique may be preferable when differences between models are more sensitive

(+) Empirical study focusing on the pro and con of single vs pairwise evaluation

(-) Focuses still on subjective metrics which might be difficult to discern between two models of comparable performance

(-) Completely exhaustive analysis of the cases in which each technique is most appropriate would require measurement on many more pairs of models than the three studied here

Open domain NLG evaluation

- Dziri et al (2022a)
- Dziri et al (2022b)
- Shuster et al (2021)

Hallucination in Neural NLG models

- Clark et al (2021)
- Dou et al (2022)
- Krishna et al (2021)

Evaluation of Long Text Generation

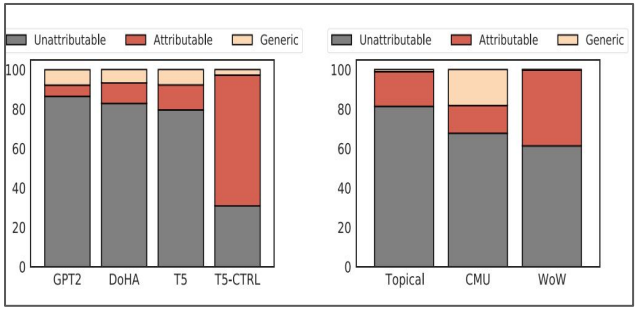
- Karpinska et al (2020)
- Smith et al (2022)

Difficulties in Human Evaluation

- Novikova et al (2017)

Challenges in Text Generation Evaluation

Evaluating Attribution in Dialogue Systems: The BEGIN Benchmark (Dziri et al 2022a)



Goal: Creates a benchmark of 12K dialogue responses across 4 models and 3 knowledge grounded datasets to measure attribution i.e if response is verifiable from source document

Data: Topical Chat (Gopalakrishnan et al., 2019) , CMU-DOG (Zhou et al., 2018), Wow (Dinan et al., 2019)

Method:

- Annotates 12K responses from 4 models fine-tuned on 3 knowledge grounded dialogue datasets into 3 broad classes : *Unattributable , Attributable, Generic*

- Deep dives into evaluating lexical overlap, semantic similarity and QG based metrics between generated response and source and show that they cannot identify between Unattributable, Attributable or Generic responses and are relying on the spurious correlation between attribution and word overlap

(+) Large scale benchmark across 3 knowledge grounded dialogue response with attribution annotation

(+) Detailed analysis on vulnerability of automatic evaluation metrics for measuring attribution

(-) Doesn't propose solutions on how to improve attribution

On the Origin of Hallucinations in Conversational Models: Is it the Datasets or the Models? (Dziri et al 2022b)

Goal: Annotates gold data from knowledge grounded dialog datasets to measure if they contain hallucinations and whether models trained on them amplify it

Data: Topical Chat (Gopalakrishnan et al., 2019) , CMU-DOG (Zhou et al., 2018), Wow (Dinan et al., 2019)

Method:

- Annotates gold responses from 3 knowledge grounded dialogue datasets into BEGIN taxonomy *{Entailment,Hallucination, Partial Hallucination,Generic, Uncooperative}* as well as 6 types of Verbal Response Modes based on speech act
- Expert and Non Expert annotations show that all datasets contain huge amount of hallucinations which is amplified by models fine-tuned on them

VRM Type	Description
Disclosure	Reveal the speaker's subjective opinions, personal experience, thoughts and feelings.
Edification	Concerns information that is objective.
Advisement	Corresponds to guiding the behaviour of the addressee through: commands, requests, suggestions, advice, permission, prohibition.
Confirmation	Compares the speaker's experience with the other's by expressing shared ideas or by agreement, disagreement.
Question	Concerns requesting information or guidance.
Acknowledge	Expresses no content, it conveys only receipt of communication from the other's speaker.

(+) Showing vulnerability of common knowledge grounded benchmark with two taxonomy for response classification

(-) Only 4000 instances for WoW ; for CMU-DOG and TopicalChat only 200 expert evaluations which is still very small

(-) Lack of experiments which shows if a model can predict VRM type from a generated response

Retrieval Augmentation Reduces Hallucination in Conversation (Shuster et al 2021)

Goal: Shows how retrieval augmented modeling reduces hallucination in knowledge grounded dialogues

Seq2Seq Model	Retrieval Mechanism	PPL	F1	Knowledge F1
BlenderBot-400m	None	11.2	19.7	16.3
	RAG DPR-Poly	9.7	21.1	24.2
BART-Large	None	14.7	20.9	17.4
	RAG DPR	12.7	22.4	22.5
	FiD-RAG DPR	11.8	21.1	29.6
T5 Large	None	12.1	19.3	14.6
	RAG DPR	9.8	21.9	25.9
	FiD-RAG DPR	9.5	22.0	27.8

Data:CMU-DOG (Zhou et al., 2018),Wow (Dinan et al., 2019)
Method:

- Extend neural-retriever-in-the-loop generative based architectures with specific improvements in Dense Passage Retrieval (DPR) through late-stage context/ candidate interaction i.e a Poly-encoder re-ranker on top of DPR retrieval
- Introduces RAG-Turn where rather than considering the context as a whole for retrieval, consider each dialogue turn separately. Augments a Fusion in decoder model with a DPR-based retriever that has been trained in a RAG setup

(+) Improving grounding in generation with retrieval augmentation

(-) Limited understanding the interplay between retrieved knowledge and knowledge stored in the model's weights

THANKS

QUESTIONS

APPENDIX

Algorithm 1 Hyperbole Clause Generation

```
function GenHyper(A)
Input: Input prompt A
Output: List of candidate <B, C> pairs cand
    Initialize Bs, cand to empty list
    subject, head_word = parse(A)
    Bs += getPreds(subject, 'RelatedTo')
    Bs += getPreds(head_word, 'HasProperty')
    Bs += subject
    for B in Bs do
        for C in getPreds(A, 'Causal') do
            cand.append(<B,C>)
        end for
        for C in getPreds(B, 'CharacteristicOf') do
            cand.append(<B,C>)
        end for
    end forreturn cand    ▷ Fit into the so...that pattern.
end function
```

Hyperbole

Generating B from A We first parse the input prompt (**A**) into the subject (**A.1**) and the headword (**A.2**). We then generate **B** using the `RelatedTo` and `HasProperty` with the COMeT and reverse COMeT model. Following the COMeT paper (Bosselut et al., 2019), we also compute the conditional log-likelihood of predicting the object tokens X:

$$\mathcal{L} = -\sum_{t=|e1|+|r|}^{|e1|+|r|+|e2|} \log P\left(x_t \mid x_{<t}\right), \qquad (2)$$

where $|e1|$, $|r|$, and $|e2|$ are the number of tokens in e1, relation, and e2, respectively. We denote the likelihood $\mathcal{L}$ as l_{AB} when the likelihood is calculated from generating **B** from **A**.

Generating C from A and from B There are two ways to generate **C**: from **A** and from **B**. Given **A**, we can leverage several causal relationships, such as `CauseDesire`, `Causes`, and `HasSubevent`. Given **B**, we produce i) predicates that **B** is not capable of, using `NotCapableOf` directly available in ConceptNet; and ii) characteristic actions of **B**, from the following relationships `DefinedAs`, `CapableOf`, `IsA`, and `UsedFor`. We also compute the conditional log-likelihoods and call them l_{AC} and l_{BC}.

Finally, we assemble pieces of **A**, **B** and **C** into the 'so...that' pattern. The candidate sentence is: '**A.1** *is so* **A.2** *that* **B** *even* **C**!'.

The IDT first defines the concept of an ironic environment. We say a situation in which an utterance occurs is surrounded by an ironic environment if the discourse context includes the following components: (1) The speaker has expectation Q at time t_0; (2) Q fails at time $t_1 > t_0$; and (3) The speaker has a negative attitude towards the failure of Q. Note that the idea of linking sarcasm to an expectation is not new to Utsumi (1996), rather it is supported by previous work (Kreuz and Glucksberg, 1989; Kumon-Nakamura et al., 1995).

Next, according to the IDT, an utterance is sarcastic if and only if it implicitly displays the ironic environment. Implicit display is realised if the following linguistic devices are present in the utterance: (1) allusion to the speaker's failed expectation Q; (2) pragmatic insincerity, realised by intentionally violating one of the pragmatic principles,

Sarcasm

To generate a line of iambic pentameter poetry, we arrange words to form a sequence of ten syllables alternating between stressed and unstressed. For example:

```
   010      1  0     10       101
Attending on his golden pilgramage
```

Following Ghazvininejad and Knight (2015), we refer to unstressed syllables with 0 and stressed syllables with 1, so that the form of a Shakespearean sonnet is $((01)^5)^{14}$. To get stress patterns for individual words, we use CMU pronunciation dictionary,[2] collapsing primary and secondary stresses. For example:

```
CAFETERIA  K AE2 F AH0 T IH1 R IY0 AH0
```

becomes

```
CAFETERIA 10100
```

Poetry Marjan

After choosing rhyme words, we create a large finite-state acceptor (FSA) that compactly encodes all word sequences that use these rhyme words and also obey formal sonnet constraints:

- Each sonnet contains 14 lines.
- Lines are in iambic pentameter, with stress pattern $(01)^5$. Following poetic convention, we also use $(01)^50$, allowing feminine rhyming.
- Each line ends with the chosen rhyme word/phrase for that line.
- Each line is punctuated with comma or period, except for the 4th, 8th, 12th, and 14th lines, which are punctuated with period.

To implement these constraints, we create FSA states that record line number and syllable count. For example, FSA state L2-S3 (Figure 2) signifies "I am in line 2, and I have seen 3 syllables so far". From each state, we create arcs for each feasible word in the vocabulary. For example, we can move from state L1-S1 to state L1-S3 by consuming any word with stress pattern 10 (such as *table* or *active*). When moving between lines (e.g., from L1-S10 to L2-S1), we employ arcs labeled with punctuation marks.

search that is further guided by the FSA. Each beam state $C_{t,i}$ is a tuple of $(h, s, word, score)$, where h is the hidden states of LSTM at step t in ith state, and s is the FSA state at step t in ith state. The model generates one word at each step.

At the beginning, $h_{0,0}$ is the initial hidden state of LSTM, $s_{0,0}$ is the start state of FSA, $word_{0,0}$ = <START> and $score_{0,0} = 0$. To expand a beam state $C_{t,i}$, we first feed $h_{t,i}$ and $word$ into the LM and get an updated hidden state h_{next}. The LM also returns a probability distribution $P(V)$ over the entire vocabulary V for next word. Then, for each succeeding state s_{suc} of $s_{t,i}$ in the FSA and the word w_{next} over each edge from $s_{t,i}$ to s_{suc}, we form a new state $(h_{next}, s_{suc}, w_{next}, score_{t,i} + log(P(w_{next})))$ and push it into next beam.

Because we fix the rhyme word at the end of each line, when we expand the beam states immediately before the rhyme word, the FSA states in those beam states have only one succeeding state—LN-S10, where $N = [1, 14]$, and only one succeeding word, the fixed rhyme word. For our beam size b = 50, the chance is quite low that in those b words there exists any suitable word to precede that rhyme word. We solve this by generating the whole sonnet *in reverse*, starting from the final rhyme word. Thus, when we expand the state L1-S8, we can choose from almost every word in vocabulary instead of just b possible words. The price to pay is that at the beginning of each line, we need to hope in those b words there exists some that are suitable to succeed comma or period.

The next step then consists in creating a probability distribution for a particular rhyme sound that we want the verse to adhere to:

$$p_{rhyme}(\mathbf{w}) = \frac{1}{Z}\mathbf{x} \text{ with } \begin{cases} x_i = 1 & \text{if } i \in R \\ x_i = \epsilon & \text{otherwise} \end{cases} \quad (10)$$

where R is the set of words that contain the required rhyme sound, ϵ is a small value close to zero, used for numerical stability, and Z is a normalization factor in order to ensure a probability distribution. We can now use $p_{rhyme}(\mathbf{w})$ as a prior probability distribution in order to reweight the neural network's standard output probability distribution—according to Equation 11—each time the rhyme scheme demands it:

$$p_{out}(\mathbf{w}) = \frac{1}{Z}(p(\mathbf{w}^t|w^{<t}, S_i) \odot p_{rhyme}(\mathbf{w})) \quad (11)$$

where $\odot$ represents pointwise multiplication.[2] As we noted before, each verse is generated in reverse; the reweighting of rhyme words is applied at the first step of the decoding process, and the rhyme word is generated first. This prevents the generation of an ill-chosen rhyme word that does not fit well with the rest of the verse.

The factorization that comes out of th NMF model can be interpreted probabilisticall (Gaussier and Goutte, 2005; Ding et al., 2008 matrix $\mathbf{W}$ can be considered as $p(\mathbf{w}|k)$, i.e. th probability of a word given a latent dimension In order to constrain the network's output to a ce tain topic, it would be straightforward to simply us $p(\mathbf{w}|k)$ as another prior probability distribution a plied to each output. Initial experiments, howeve indicated that such a blind modification of the ou put probability distribution for every word of th output sequence is detrimental to syntactic fluenc In order to combine syntactic fluency with topica consistency, we therefore condition the weightin of the output probability distribution on the entrop of that distribution: when the output distribution' entropy is low, the neural network is certain c its choice for the next word in order to generat a well-formed sentence, so we will not change i On the other hand, when the entropy is high, w will modify the distribution by using the topica distribution $p(\mathbf{w}|k)$ for a particular latent dime sion as prior probability distribution—analogou to Equation 11—in order to inject the desired topi The entropy threshold, above which the modifie distribution is used, is set experimentally.

- the log-probability score of the generated verse, according to the encoder-decoder architecture (section 3.1);
- compliance with the rhyme constraint (section 3.2.1); additionally, the extraction of the preceding group of consonants (cf. Table 1) allows us to give a higher score to rhyme words with disparate preceding consonant groups, which elicits more interesting rhymes;
- compliance with the topical constraint (section 3.2.2); the score is modeled as the sum of the probabilities of all words for the defined dimension;
- the optimal number of syllables, modeled as a Gaussian distribution with mean μ and standard deviation σ;[4]
- the log-probability score of a standard n-gram model.

3.2.2 Topical constraint

For the modeling of topical coherence, we make use of a latent semantic model based on non-negative matrix factorization (NMF; Lee & Seung, 2001). Previous research has shown that non-negative factorization methods are able to induce clear-cut, interpretable topical dimensions (Murphy et al., 2012). As input to the method, we construct a frequency matrix $\mathbf{A}$, which captures co-occurrence frequencies of vocabulary words and context words.[3] This matrix is then factorized into two non-negative matrices $\mathbf{W}$ and $\mathbf{H}$,

$$\mathbf{A}_{i\times j} \approx \mathbf{W}_{i\times k}\mathbf{H}_{k\times j} \quad (12)$$

where k is much smaller than i, j so that both instances and features are expressed in terms of a few components. Non-negative matrix factorization enforces the constraint that all three matrices must be non-negative, so all elements must be greater than or equal to zero. Using the minimization of the Kullback-Leibler divergence as an objective function, we want to find the matrices $\mathbf{W}$ and $\mathbf{H}$ for which the divergence between $\mathbf{A}$ and $\mathbf{WH}$ (the multiplication of $\mathbf{W}$ and $\mathbf{H}$) is the smallest. The factorization is carried out through the iterative application of update rules. Matrices $\mathbf{W}$ and $\mathbf{H}$ are randomly initialized, and the rules in 13 and 14 are iteratively applied—alternating between them. In each iteration, each vector is adequately normalized, so that all dimension values sum to 1.

$$\mathbf{H}_{a\mu} \leftarrow \mathbf{H}_{a\mu}\frac{\sum_i \mathbf{W}_{ia}\frac{\mathbf{A}_{i\mu}}{(\mathbf{WH})_{i\mu}}}{\sum_k \mathbf{W}_{ka}} \quad (13)$$

$$\mathbf{W}_{ia} \leftarrow \mathbf{W}_{ia}\frac{\sum_\mu \mathbf{H}_{a\mu}\frac{\mathbf{A}_{i\mu}}{(\mathbf{WH})_{i\mu}}}{\sum_v \mathbf{H}_{av}} \quad (14)$$

In general, we can decompose the generation process by converting a story x into a more abstract representation z. The negative log likelihood of the decomposed problem is given by

$$\mathcal{L} = -\log \sum_z p(x|z)p(z). \quad (1)$$

of certain tokens). Instead, we minimize a variational upper bound of the loss by constructing a deterministic posterior $q(z|x) = 1_{z=z^*}$, where z^* can be given by running semantic role labeller or coreference resolution system on x. Put together, we optimize the following loss:

$$z^* = \arg\max_z p(z|x) \quad (2)$$

$$\mathcal{L} \leq -\log p(x|z^*) - \log p(z^*) \quad (3)$$

This approach allows models $p(z^*)$ and $p(x|z^*)$ to be trained tractably and separately.

Figure 2: Verb-Attention. To improve the model's ability to condition upon past verbs, one head of the decoder's self-attention mechanism is specialized to only attend to previously generated verbs.

consider sequences of verbs, we designate one of the heads of the decoder's multihead self-attention to be a *verb-attention* head (see Figure 2). By masking the self-attention appropriately, this verb-attention head can only attend to previously generated verbs. When the text does not yet have a verb, the model attends to a zero vector. We show that focusing on verbs with a specific attention head generates a more diverse array of verbs and reduces repetition in generation.

identify all people, organizations, and locations. We replace these spans with placeholder tokens (e.g. *ent0*). If any two entity mentions have an identical string, we replace them with the same placeholder. For example, all mentions of *Bilbo Baggins* will be abstracted to the same entity token, but *Bilbo* would be a separate abstract entity.

- **Coreference-based Entity Anonymization**: The above approach cannot detect different mentions of an entity that use different

5.1.3 Collaboration Capabilities: Ability to Work Jointly. To investigate the extent that writers interact with GPT-3 in the writing process, we adapted the notions of equality and mutuality from Storch [52]. We redefined *equality* as how evenly a writer and GPT-3 distributed turns (e.g. the degree of deviation from even division of work) and *mutuality* as the level of interaction a writer has with GPT-3 (e.g. querying multiple times, choosing suggestions, reopening suggestions, and navigating through suggestions). Concretely, we computed equality and mutuality scores for a writing session by counting the number of event blocks as follows. Let

$$\mathcal{H} = \{\text{insert}\},\ \mathcal{M} = \{\text{choose}\}$$

be the sets representing human-generated event blocks and machine-generated event blocks, respectively. Also, let

$$\mathcal{I} = \{\text{insert}, \text{choose}, \text{reopen}, \text{navigate}\},$$
$$\mathcal{A} = \{\text{dismiss}, \text{insert}, \text{delete}\}$$

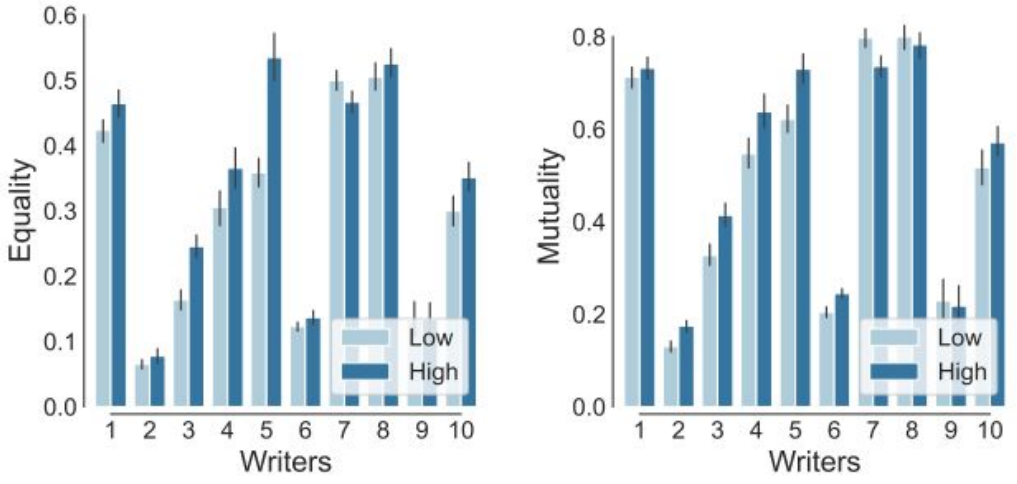
denote the sets of event blocks corresponding to human-machine interactions and event blocks corresponding to writing alone, respectively. Given a set of events $\{e_i\}$, we define:

$$\text{equality} = 1 - \frac{\sum_i [e_i \in \mathcal{H}] - \sum_i [e_i \in \mathcal{M}]}{\sum_i [e_i \in \mathcal{H}] + \sum_i [e_i \in \mathcal{M}]},$$

where $[P] = 1$ if P is true and 0 if not. Moreover, define:

$$\text{mutuality} = \frac{\sum_i [e_i \in \mathcal{I}]}{\sum_i [e_i \in \mathcal{I}] + \sum_i [e_i \in \mathcal{A}]}.$$

An equality score of 1 means perfect division of turns for writing and 0 means that there was no turn change (i.e. it was written entirely by a writer or GPT-3). For mutuality, score 1 means writers interacted with GPT-3 the entire time (without writing on their own as a result) and 0 means writers never interacted with GPT-3. Note that these scores are not based on final texts, but event blocks. This accounts for text written (by either writers or GPT-3) during the writing session, that may not be present in final texts.



(a) Equality **(b) Mutuality**

Figure 8: Equality (a) and mutuality (b) (y-axis) vary across writers (x-axis) greatly. Also, some writers had more equal and mutual collaboration with GPT-3 with high randomness (dark blue), whereas others had such collaboration with GPT-3 with low randomness (light blue).

S	Set of original persona sentences
C	Set of expanded persona sentences (includes S and a null persona $\varnothing$)
H	Dialog history with alternative turns from each speaker
x	Target utterance
z	Discrete latent random variable $\in \{1, 2, \ldots, \|C\|\}$
e	Mean of RoBERTa subword embeddings as an encoder
t_k	Expansion type for k-th expansion
f_i	i-th feature function for prior network; $i \in \{1, 2, 3\}$
θ	Parameters for prior network $p_\theta(z\|H, C)$
ϕ	Parameters for generator network $p_\phi(x\|H, C_z)$
α	Parameters for inference network $p_\alpha(z\|x, H, C)$

Table 1: Summary of notation used in the paper

The dimension of the expansion type embedding was set to 5. Finally, the prior model can be represented concisely as $p_\theta(z = k|H, C) \propto \exp(f(H, C_k, t_k))$, where $f(H, C_k, t_k)$ is the sum $\lambda_1 * f_1(H, C_k) + \lambda_2 * f_3(t_k) + \lambda_3 * f_3(t_k, H)$ with λ_i's as trainable parameters.

Majumder et al (2020)

Our training data $\mathcal{D}$ consists of instances of dialog history H and ground truth dialog responses x. We train our model parameters θ and ϕ to maximize the likelihood of the target dialog response x given the dialog history: $\log p(x|H, C; \theta, \phi)$ totalled over $\mathcal{D}$. Since the discrete random variable z is unobserved in the training data, we must marginalize over z to compute the desired likelihood $p(x|H; \theta, \phi)$:

$$\log p(x|H; \theta, \phi) = \log \mathbb{E}_{z \sim p_\theta(z|H)}[p_\phi(x|z, H)];$$

where we drop C from the conditionals for simplicity.

We obtain *f1(H, C_k)*: a scalar feature using a bilinear product e(H), e(C_k) to align the persona sentences with the dialog history. f2(t_k) that represents a global preference over the relation type embedded via a type embedding layer; and (b) f3(t_k, H) that appends the expansion type embedding with dialog history encoding e(H), followed by a linear layer to obtain a real-valued score for history-specific preference over the expansion type

4.3 Learning and Inference

Our training data $\mathcal{D}$ consists of instances of dialog history H and ground truth dialog responses x. We train our model parameters θ and ϕ to maximize the likelihood of the target dialog response x given the dialog history: $\log p(x|H, C; \theta, \phi)$ totalled over $\mathcal{D}$. Since the discrete random variable z is unobserved in the training data, we must marginalize over z to compute the desired likelihood $p(x|H; \theta, \phi)$:

$$\log p(x|H; \theta, \phi) = \log \mathbb{E}_{z \sim p_\theta(z|H)}[p_\phi(x|z, H)];$$

where we drop C from the conditionals for simplicity.

Inference Network Note that the number of persona expansions is typically in the range 150-250, and thus it is computationally expensive to marginalize over the entire selection space of z during training. We instead optimize a variational lower bound (ELBO) of $\log p(x|H; \theta, \phi)$ given as

$$\mathbb{E}_{z \sim q_\alpha(z|H)}[\log p_\phi(x|z, H)] - KL(q_\alpha(z|x, H)||p_\theta(z|H)),$$

where we use the inference network $q_\alpha(z|x, H)$ to compute the approximate posterior (Kingma and Welling, 2014). In our initial experiments, we observe that using an inference network leads to better perplexity values than using samples from the prior.

The architecture of the inference network is similar to that of the prior network, a log-linear model. Along with the features related to dialog history and expansion types, we additionally include another scalar feature: a bilinear product $\langle x, C_k \rangle$ between the encoded persona and ground truth response x encoded with RoBERTa embeddings to align the persona choice according to the target utterance.

Facts Encoder

The Facts Encoder of Fig. 3 is similar to the Memory Network model first proposed by (Weston, Chopra, and Bordes 2015; Sukhbaatar et al. 2015). It uses an associative memory for modeling the facts relevant to a particular problem—in our case, an entity mentioned in a conversation–then retrieves and weights these facts based on the user input and conversation history to generate an answer. Memory network models have been successfully used in Question Answering to make inferences based on the facts saved in the memory (Weston et al. 2016).

In our adaptation of memory networks, we use an RNN encoder to turn the input sequence (conversation history) into a vector, instead of a bag of words representation as used in the original memory network models. This enables us to better exploit interlexical dependencies between different parts of the input, and makes this memory network model (facts encoder) more directly comparable to a SEQ2SEQ model.

More formally, we are given an input sentence $S = \{s_1, s_2, ..., s_n\}$, and a fact set $F = \{f_1, f_2, ..., f_k\}$ that are relevant to the conversation history. The RNN encoder reads the input string word by word and updates its hidden state. After reading the whole input sentence the hidden state of the RNN encoder, u is the summary of the input sentence. By using an RNN encoder, we have a rich representation for a source sentence.

Let us assume u is a d dimensional vector and r_i is the bag of words representation of f_i with dimension v. Based on (Sukhbaatar et al. 2015) we have:

$$m_i = Ar_i \tag{1}$$

$$c_i = Cr_i \tag{2}$$

$$p_i = \text{softmax}(u^T m_i) \tag{3}$$

$$o = \sum_{i=1}^{k} p_i c_i \tag{4}$$

$$\hat{u} = o + u \tag{5}$$

Where $A, C \in \mathbb{R}^{d \times v}$ are the parameters of the memory network. Then, unlike the original version of the memory network, we use an RNN decoder that is good for generating the response. The hidden state of the RNN is initialized with $\hat{u}$ which is a symmetrization of input sentence and the external facts, to predict the response sentence R word by word.

As alternatives to summing up facts and dialog encodings in equation 5, we also experimented with other operations such as concatenation, but summation seemed to yield the best results. The memory network model of (Weston, Chopra, and Bordes 2015) can be defined as a multi-layer structure. In this task, however, 1-layer memory network was used, since multi-hop induction was not needed

Ghazvininejad et al (2018)

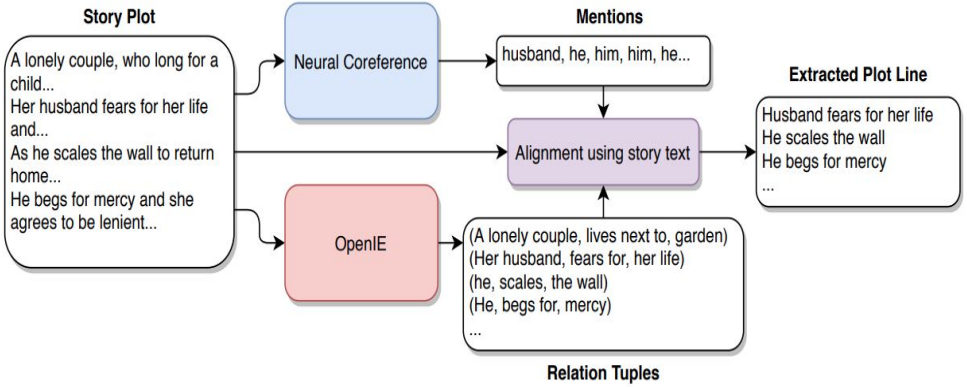


Figure 1: An illustration of high level plot point extraction.

Ammanabrolu et al 2020

	C2PO
Mystery	**Holmes decides go.** Holmes wants to go. Holmes begins to see something. Holmes begins to look around. **Holmes notices a pair of trouser knees.** Holmes wants to clean up. Holmes begins take a shower. Holmes wants to get ready. Holmes wants to walk to the store. **Holmes taps in front of Wilson's shop.** Holmes tries say hello. Holmes wants start the car. Holmes tries to drive to the scene. **He calls Police Inspector Jones.**
Fairy Tale	**Queen asks her mirror.** Queen wants to look better. Queen wants to try on clothes. Queen starts to be mad. **Queen is furious.** Queen tries to relax. Queen wants to take a nap. Queen starts to get up. Queen begins to approach someone. **She appears at a dwarfs'.** Queen starts to surprise everyone. Queen starts to have a party. queen wants to have money. Queen tries to buy poison comb. **She brushes with poisoned comb.** Queen tries to wash her hair. Queen starts dry it. Queen wants to be hungry. Queen wants to get the knife. **Queen cuts the apple in half.**

4.2 Plot Graph Generation

Once we have established a series of plot points $\mathcal{P} = \{p_1, p_2, ..., p_n\}$, we move on to plot graph generation as illustrated in Figure 2. A plot graph is generated for each pair of adjacent plot points $(p_i, p_{i+1}), i \in \{1, .., n-1\}$ and then linked together in the order the plot points first appear in P to form a plot graph for an entire story.

The process to generate a plot graph between adjacent plot points p_1, p_2 is as follows. Starting from p_1, we use COMET (Bosselut et al. 2019) to generate candidate next events in the story. The *wants* relation indicates a direct forward cause—a character has a want and therefore performs an action. We recursively query COMET to generate k event candidates n times going forward starting with p_1; let this be $\mathcal{G}^f$. The *needs* relation indicates backward enablement—a character needed something to be true to do an action. We recursively query COMET to generate k event candidate n times going backward from p_2; let this be $\mathcal{G}^b$. This gives us two directed acyclic graphs as seen in Figure 2. The relations in $\mathcal{G}^f$ and $\mathcal{G}^b$ are weighted proportional to the likelihood score produced by COMET for each inference.

The next step is to look for the optimal way to link $\mathcal{G}^f$ and $\mathcal{G}^b$ and computing the probability of reaching a node $u \in \mathcal{G}^f$ looking at all nodes $\forall v \in \mathcal{G}^b$. Let $Pr^{needs}(u|v)$ be the probability of generating event e_2 as determined by COMET under the $needs$ relation, conditioned on e_1, and $Pr^{wants}(v|u)$ be the same but under the $wants$ relation. We define this link's weight as:

$$w(u,v) = \frac{Pr^{wants}(u|v)}{\alpha_u^{wants}} + \frac{Pr^{needs}(v|u)}{\alpha_v^{needs}} \quad (1)$$

were α_u^{wants} and α_v^{needs} are normalization constants. Here we set them equal to the probability of generating the word "to", a word in ATOMIC common to both relation types. This process is repeated for all nodes until we have found a set of optimal links.[2]

Majumder et al (2022)

2.5. Training

The parameters for $\mathcal{R}$, $\mathcal{G}$, $\mathcal{P}$, and the knowledge selector can be jointly trained end-to-end with backpropagation by summing up the negative log-likelihoods for the predictions and NLEs. We found that updating parameters for the knowledge resource $\mathcal{K}$ led to a minimal improvement; hence, $\mathcal{K}$ is fixed for computational ease.

However, due to the presence of z_i^rs in $\mathcal{R}$, we instead have to optimize a lower bound $\mathcal{E}$ of the original log-likelihood. We follow Bastings et al. (2019) and optimize $\min_{\theta^r,\theta^g,\theta^{ks},\theta^p} \mathcal{L}_1$ with

$$\mathcal{L}_1 = -\mathcal{E}(\theta^r, \theta^k, \theta^{ks}, \theta^g, \theta^p) + \lambda_0^r \sum_{i=1}^{N} z_i^r + \lambda_1^r \sum_{i=1}^{N-1} |z_i^r - z_{i+1}^r|, \quad (1)$$

where the second term is the L_1 penalty, the third term is a fused Lasso to control the total number of transitions for compactness (Lei et al., 2016), and λ_0^r and λ_1^r are hyperparameters. Similarly, we have another lower bound for the z_i^k variables in the knowledge selection step, for which we optimize $min_{\theta^{ks},\theta^g,\theta^p} \mathcal{L}_2$ with

$$\mathcal{L}_2 = -\mathcal{E}(\theta^{ks}, \theta^g, \theta^p) + \lambda_0^k \sum_{i=1}^{M} z_i^k, \quad (2)$$

where the second term denotes L_1 regularization for sparse knowledge selection. Finally, we combine the lower bounds as $\alpha \times \mathcal{L}_1 + (1 - \alpha) \times \mathcal{L}_2$, where $\alpha \in [0, 1]$ is a hyperparameter. We estimate the gradient of $\mathcal{E}$ via Monte-Carlo sampling from the reparameterized HardKuma variables (Kingma & Welling, 2014). All hyperparameters are chosen based on a greedy search over the task prediction accuracy (more in Appendix A).

2.1. Extractive Rationales via Binary Latent Variables

We define a neural module $\mathcal{R}$ that selects an ER from the input. An ER is a minimal sufficient subset of input parts (e.g., tokens for text or super-pixels for images) most responsible for the model's prediction (Lei et al., 2016). In Fig. 1a, we see an example from the natural language inference task (Bowman et al., 2015) (details in Section 3), where the ER is {"men", "people", "bicycle race", "riding bikes"}, the most responsible units for the prediction (*entailment*).

We model the selection of ERs using a series of latent variables ranging from $[0, 1]$ ($z_i^r \in \mathcal{Z}^r$) over the N input units. A unit becomes a part of the ER *iff* its associated variable takes value 1. Following (Bastings et al., 2019), we use the Hard Kumaraswamy distribution (referred to as HardKuma) as the reparameterization strategy to learn these latent selectors using backpropagation. The parameters of the neural module $\mathcal{R}$ are denoted by θ^r, which estimate the HardKuma variables for the input units. We also encourage the ERs to be terse, and we control the sparsity using an L_1 relaxation defined by the tractable Kumaraswamy CDF.

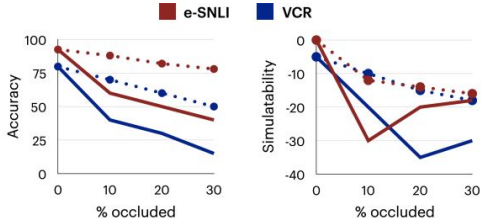
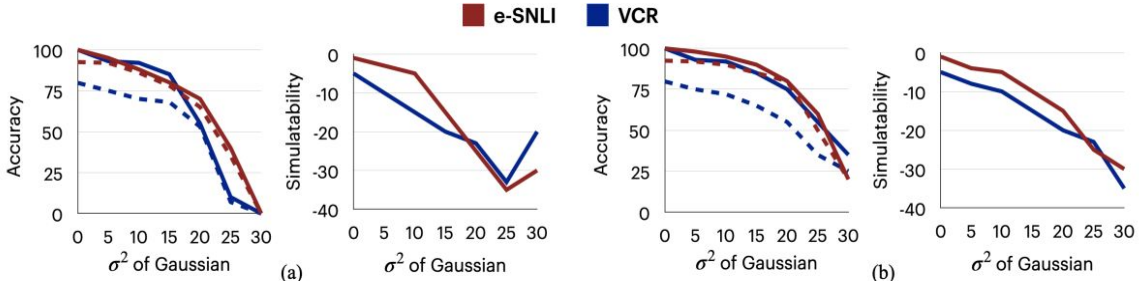


Figure 4. **Feature importance agreement.** Left: Solid lines indicate the prediction accuracy when features important for NLEs are occluded. The dotted lines indicate the prediction accuracy when random features are dropped. Right: solid lines indicate the simulatabilities when features important for prediction are occluded. Dotted lines indicates simulatabilities for random occlusions. In both, solid lines should be lower (meaning higher changes) than dotted lines for better label-NLE association.

(a) input and (b) selected knowledge snippets



Gabriel et al (2020)

The inferences obtained by both heuristic and model-based methods (Section 4.2) only consider one sentence at a time. To improve coherence with the rest of the narrative, we filter the inferences that have a low *coherence* with the given narrative. Specifically, inspired by information theory (Shannon 1948; Hale 2001), we define coherence as a measure based on the cross entropy of the story tokens conditioned on a particular candidate knowledge inference. For a tuple $< e_1, d, e_2 > \in R_i$ matched to a sentence S_i, and a language model Θ, we compute the cross entropy loss of the tokens in the story, where $< d, e_2 >$ follow S_i: $CE(S_1, ...S_i, < d, e_2 >, ...S_5)$.[5] We use a transformer-based language model, and convert $< d, e_2 >$ to natural language using hand-crafted templates shown in Table 2.

In practice, we divide the dimensions into causes (`xNeed`, `xIntent`, `xAttr`) and effects (`xWant`, `xEffect`, `xReact`, `oWant`, `oEffect`, `oReact`). For cause inferences, we compute coherence with the previous and current sentences in the story. For effect inferences we use the full story. This allows us to effectively measure how well the extracted inferences may follow from past or predict future story events.

Memory-augmented Model. To incorporate inferences generated for other sentences in the story while generating inferences for a given sentence, we extend the model with a recurrent memory component, inspired by episodic memory. $M^m \in \mathbb{R}^{R^m \times L^r \times H}$ is the external memory, where R^m is either the maximum number of inferences per instance to store in memory (during training time) or the current number of instances (during decoding time), L^r is the maximum inference sequence length,[9] and H is the hidden state dimension.

The memory-augmented model takes as input a memory update matrix $M^u \in \mathbb{R}^{R^u \times L^r \times H}$, where R^u is the number of inferences used to update memory, and incorporates it into the memory matrix:

$$M^m = M^m \oplus f_{emb}(M^u) \quad (2)$$

$\oplus$ stands for matrix concatenation, and f_{emb} is an embedding layer trained jointly with the rest of the model. After the memory is updated, we average M^m across the token dimension to get $\theta^{mem} \in \mathbb{R}^{R^m \times H}$:

$$\theta^{mem} = \frac{1}{L^r} \cdot \sum_{l=1}^{L^r} M^{ml} \quad (3)$$

We denote the context representation obtained from GPT or GPT2's hidden state as $C^o \in \mathbb{R}^{L^c \times H}$, where L^c is the context sequence length. We average it across all tokens, obtaining $\theta^{ctx} \in \mathbb{R}^H$. We then prune the memory to the top-k most relevant inferences, measured by cosine similarity between the memory θ^{mem} and context vectors θ^{ctx}. The memory output $M^o \in \mathbb{R}^H$ is the average of the top-k inferences.

Finally, we reweigh the context representation C^o to consider the memory:

$$C^o = C^o + \text{proj}(M^o) \quad (4)$$

Where proj is a linear projection layer used to project the memory output into the same hidden dimensional space as the context representation.

At training time, the memory consists of previously extracted relations from our distant supervision, while at test time, it consists of previously generated inferences, recalling the model's prior decisions. For both PARA-COMET model variants, we minimize the cross entropy loss of the entire sequence (input and output).

Lin et al (2020)

$$\texttt{score}(x) = |S(x)| \frac{|\bigcup_{s_i \in S(x)} \{w | w \in s_i\}|}{\sum_{s_i \in S(x)} \text{len}(s_i)} \rho(x),$$

where $\rho(x) = \frac{|\mathcal{X}|}{\max_{c_i \in x} |\{x' \mid c_i \in x' \text{ and } x' \in \mathcal{X}\}|}$. The

We therefore design a function to score a concept-set x based on scene diversity and inverse frequency penalty. We denote S(x) as the set of unique sentences that contain all given concepts {c1, c2, . . . , ck}, and then we have The first term in score is the number of unique sentences covering all given concepts in x, and the second term is to represent the diversity of the scenes described in these sentences. Th last term ρ(x) is the penalty of inverse frequency. Specifically, we find the concept in x that has the maximum “set frequency” (i.e., the number of unique concept sets containing a particular concept), then we take the inverse with the number of all concept-sets for normalization. This penalty based on inverse set-frequency effectively controls the bias towards highly frequent concepts. With the distribution of such scores of concept-sets, we sample our candidate examples for the next steps.

A Probabilistic Framework for αNLI: A distinct feature of the αNLI task is that it requires jointly considering all available observations and their commonsense implications, to identify the correct hypothesis. Formally, the αNLI task is to select the hypothesis h^* that is most probable given the observations.

$$h^* = \arg\max_{h^i} P(H = h^i | O_1, O_2) \quad (1)$$

Rewriting the objective using Bayes Rule conditioned on O_1, we have:

$$P(h^i | O_1, O_2) \propto P(O_2 | h^i, O_1) P(h^i | O_1) \quad (2)$$

We formulate a set of probabilistic models for αNLI that make various independence assumptions on Equation 2 – starting from a simple baseline that ignores the observations entirely, and building up to a fully joint model. These models are depicted as Bayesian Networks in Figure 2.

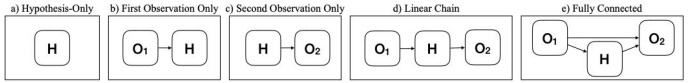


Figure 2: Illustration of the graphical models described in the probabilistic framework. The "Fully Connected" model can, in theory, combine information from both available observations.

Hypothesis Only: Our simplest model makes the strong assumption that the hypothesis is entirely independent of both observations, i.e. $(H \perp O_1, O_2)$, in which case we simply aim to maximize the marginal $P(H)$.

First (or Second) Observation Only: Our next two models make weaker assumptions: that the hypothesis depends on only one of the first O_1 or second O_2 observation.

Linear Chain: Our next model uses both observations, but considers each observation's influence on the hypothesis *independently*, i.e. it does not combine information across the observations. Formally, the model assumes that the three variables $\langle O_1, H, O_2 \rangle$ form a linear Markov chain, where the second observation is conditionally independent of the first, given the hypothesis (i.e. $(O_1 \perp O_2 | H)$). Under this assumption, we aim to maximize a somewhat simpler objective than Equation 2:

$$h^* = \arg\max_{h^i} P(O_2 | h^i) P(h^i | O_1) \text{ where } (O_1 \perp O_2 | H) \quad (3)$$

3.2 Abductive Natural Language Generation

Given $h^+ = \{w_1^h \dots w_l^h\}$, $O_1 = \{w_1^{o1} \dots w_m^{o1}\}$ and $O_2 = \{w_1^{o2} \dots w_n^{o2}\}$ as sequences of tokens, the αNLG task can be modeled as $P(h^+ | O_1, O_2) = \prod P(w_i^h | w_{<i}^h, w_1^{o1} \dots w_m^{o1}, w_1^{o2} \dots w_n^{o2})$ Optionally, the model can also be conditioned on background knowledge $\mathcal{K}$. Parameterized models can then be trained to minimize the negative log-likelihood over instances in $\mathcal{ART}$:

$$\mathcal{L} = -\sum_{i=1}^{N} \log P(w_i^h | w_{<i}^h, w_1^{o1} \dots w_m^{o1}, w_1^{o2} \dots w_n^{o2}, \mathcal{K}) \quad (4)$$

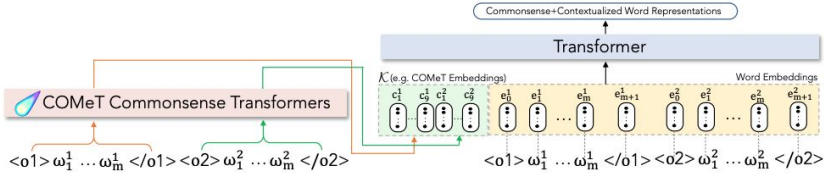


Figure 3: Overview of an αNLG model that integrates commonsense representations obtained from COMeT (Bosselut et al., 2019) with GPT2. Each observation is input to the COMeT model to obtain nine embeddings, each associated with one commonsense inference type.

Bhagavatula et al (2020)

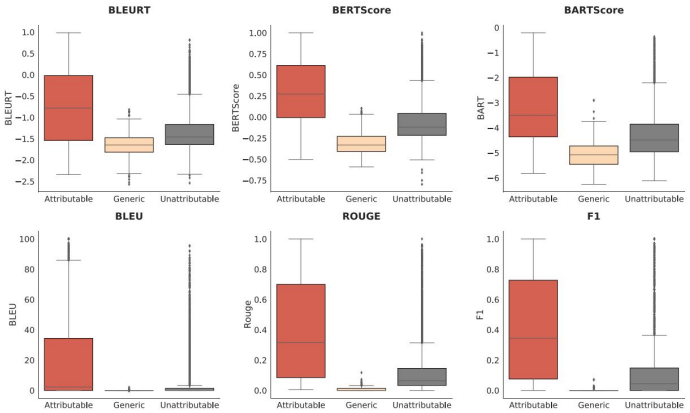


Figure 3: The distribution of scores assigned by semantic similarity metrics (upper row) and lexical overlap scores metrics (lower row) to the BEGIN test set.

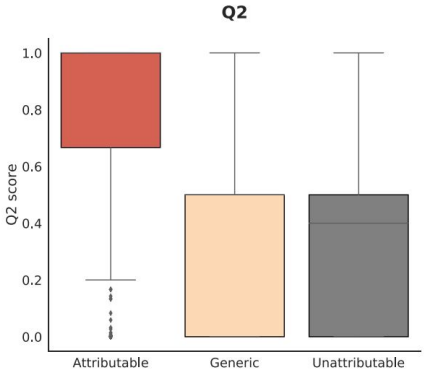


Figure 4: The distribution of Q^2 scores for each of the three example categories in the BEGIN test set.

Dziri et al (2022)

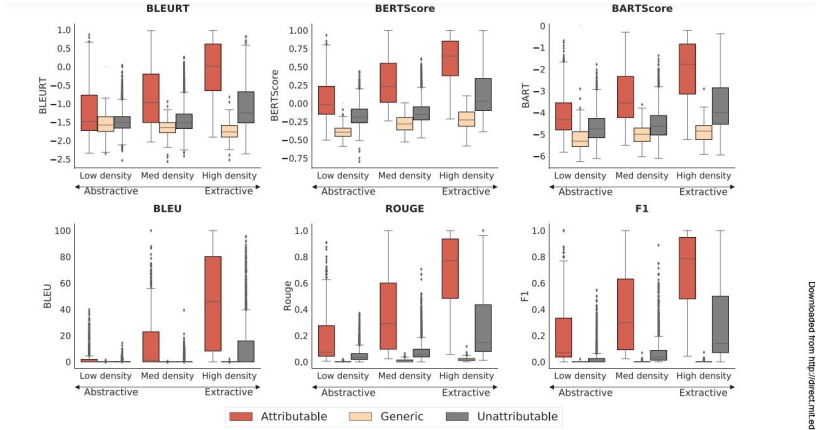
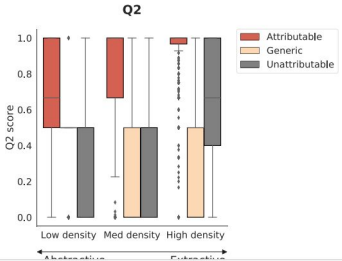


Figure 5: Scores assigned to each of the three BEGIN categories by semantic similarity metrics (upper row) and lexical overlap metrics (lower row), broken down by extractivity of the response (the extent to which it copies verbatim from the knowledge).



is likely to be scored much lower by all of these metrics if it is abstractive. Even more strikingly, unattributable extractive responses score higher on average than attributable abstractive responses in all metrics.

We observe similar trends for the classifiers (Figure 7). The performance on classifying attributable responses is much higher in extractive cases than in abstractive ones. In contrast, the performance on unattributable responses is typically worse in the extractive cases. This pattern of results suggests that a response that is unattributable

RAG : https://ai.facebook.com/blog/retrieval-augmented-generation-streamlining-the-creation-of-intelligent-natural-language-processing-models/

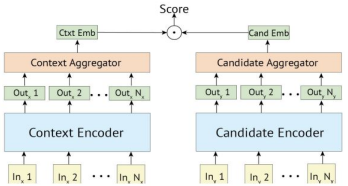
Neural-Retrieval-in-the-loop Question Answering

- **RAG**: Retrieval-Augmented Generation [1]
  - Combine a neural Dense Passage Retriever (DPR) with a seq2seq generative model (BART)
  - Train the retriever end-to-end on generation tasks by incorporating retrieval scores into the generations
  - **RAG Token**: attend/marginalize over all documents jointly
  - **RAG Sequence:** attend/marginalize over documents separately
- **FiD**: Fusion in Decoder [2]
  - Retrieve relevant documents with a retriever
  - Encode all documents independently and attend jointly in the decoder.
  - Retriever is **not trained** during generator training.

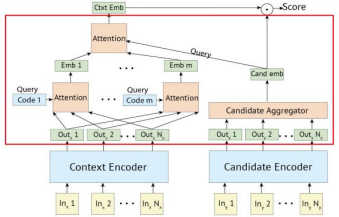
Neural-Retrieval-in-the-loop Improvements for Dialogue

- **Improving Augmented Generation**
  - **RAG Turn**
    - Rather than consider the context *as a whole* for retrieval, consider each dialogue turn separately.
    - Consider documents within a turn jointly; consider turns either separately or jointly.
  - **FiD-RAG**
    - Augment a FiD model with a DPR-based retriever that has been trained in a RAG setup
    - Allows FiD to benefit from the signal RAG receives during training.

- **Improving Retrieval**
  - **Dense Passage Retrieval (DPR)**: retrieve via dot-product similarity of independently encoded context/documents
  - **Poly-encoders**: late-stage context/candidate interaction
    - We apply a Poly-encoder re-ranker on top of DPR retrieva



Bi-encoder (DPR Model)



Poly-encoder