PEER: A Collaborative Language Model - Timo Schick et al. (Meta AI, CMU, PSL University, UCL)

Mathieu Ravaut

Nanyang Technological University

Table of content

Motivation

Current language models produce text in a single pass from left to right.

They have several major limitations :

- Not able to retroactively modify or refine their own outputs.
- Hard to control.
- Can hallucinate content.
- Lack the ability to explain their intentions.

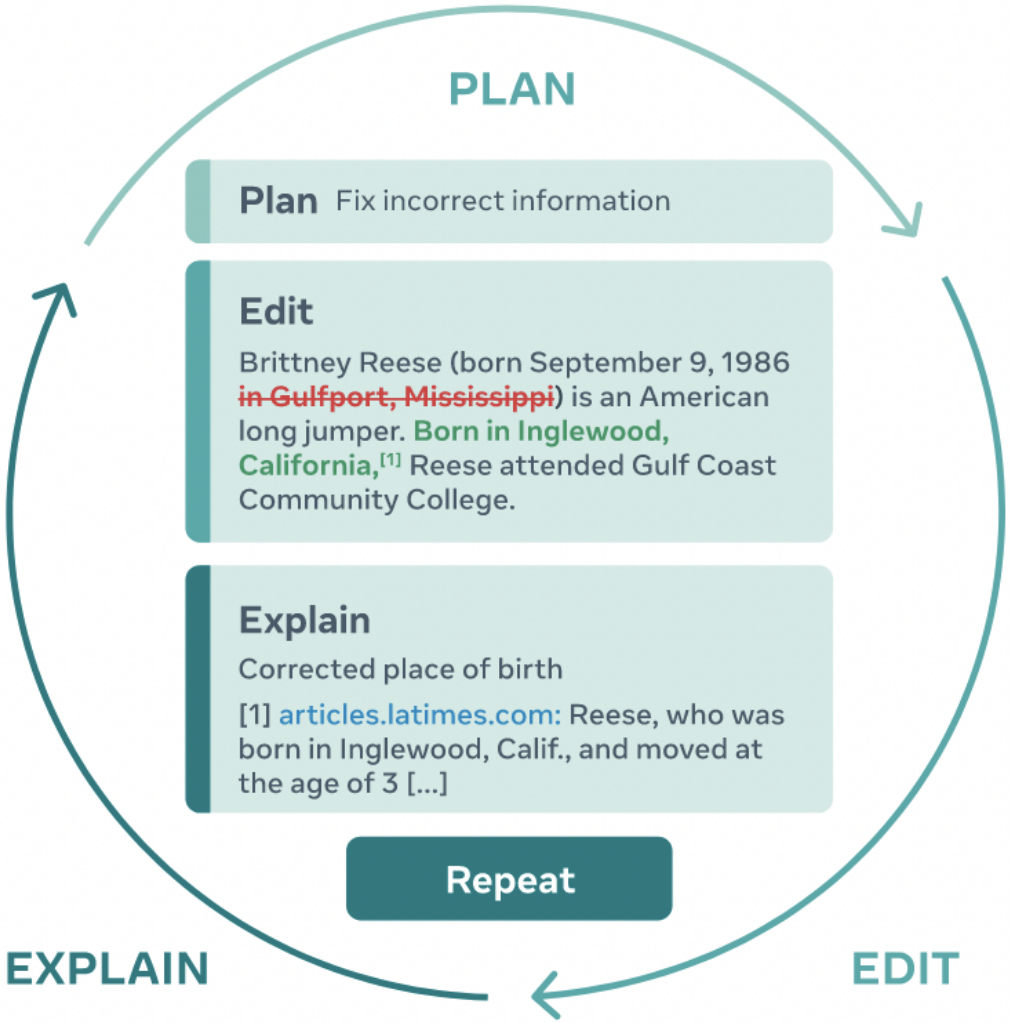
In brief, language models cannot perform any kind of *collaborative* writing.

Introduction

This paper introduces PEER : (**P**lan, **E**dit, **E**xplain, **R**epeat).

- **Plan** an action to be applied to the text.
- This plan is then realized by an **edit**.
- The model can **explain** this edit in form of textual content.
- **Repeat** the process until the text is in satisfactory form.

PEER Overview



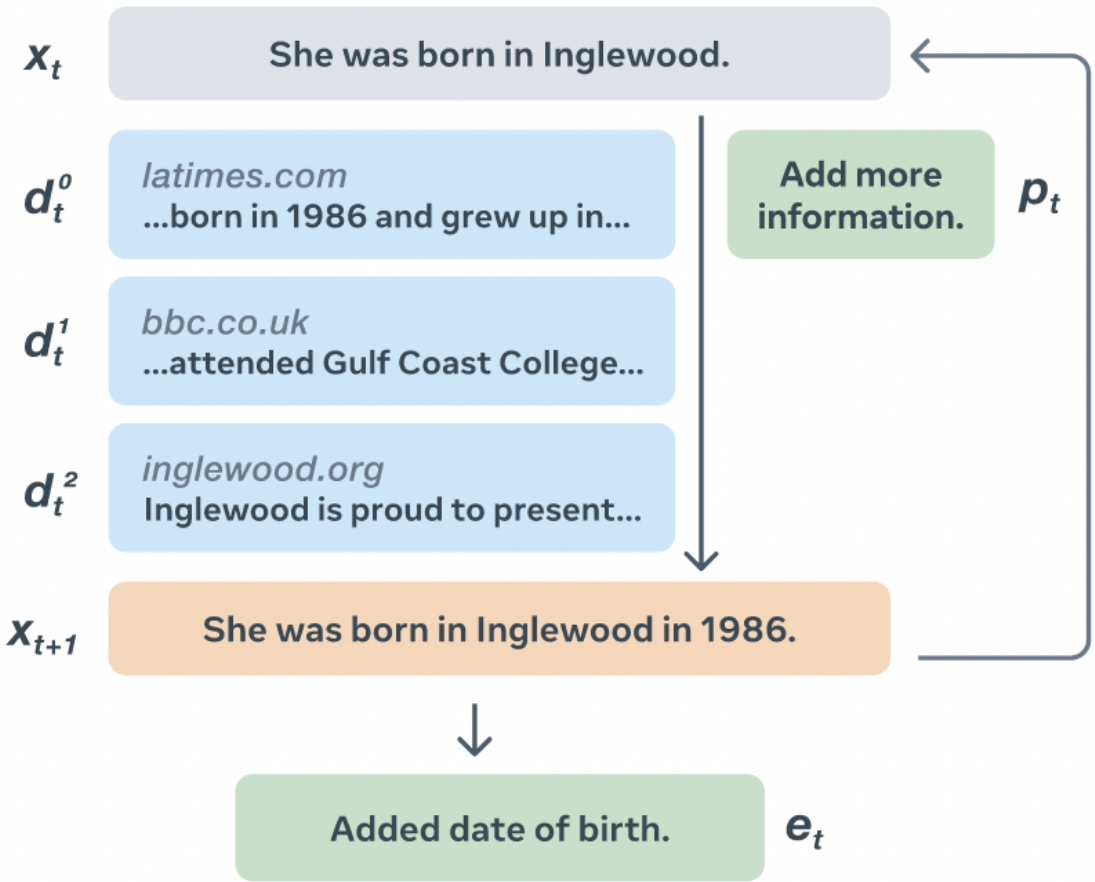
Notation

At each iteration t :

- Edit a text sequence x_t to obtain an updated version x_{t+1}.
- Assume access to a set of k documents $D_t = \{d_t^1, \ldots, d_t^k\}$.
- Formulate a plan p_t.
- Provide a textual explanation e_t.

Can repeat the process to obtain sequence $x_t, x_{t+1}, x_{t+2}, \ldots$ until we have $x_n = x_{n-1}$.

Process


x_t
She was born in Inglewood.
d_t^0
latimes.com
...born in 1986 and grew up in...
d_t^1
bbc.co.uk
...attended Gulf Coast College...
d_t^2
inglewood.org
Inglewood is proud to present...
Add more information.
p_t
x_{t+1}
She was born in Inglewood in 1986.
Added date of birth.
e_t

Model Components

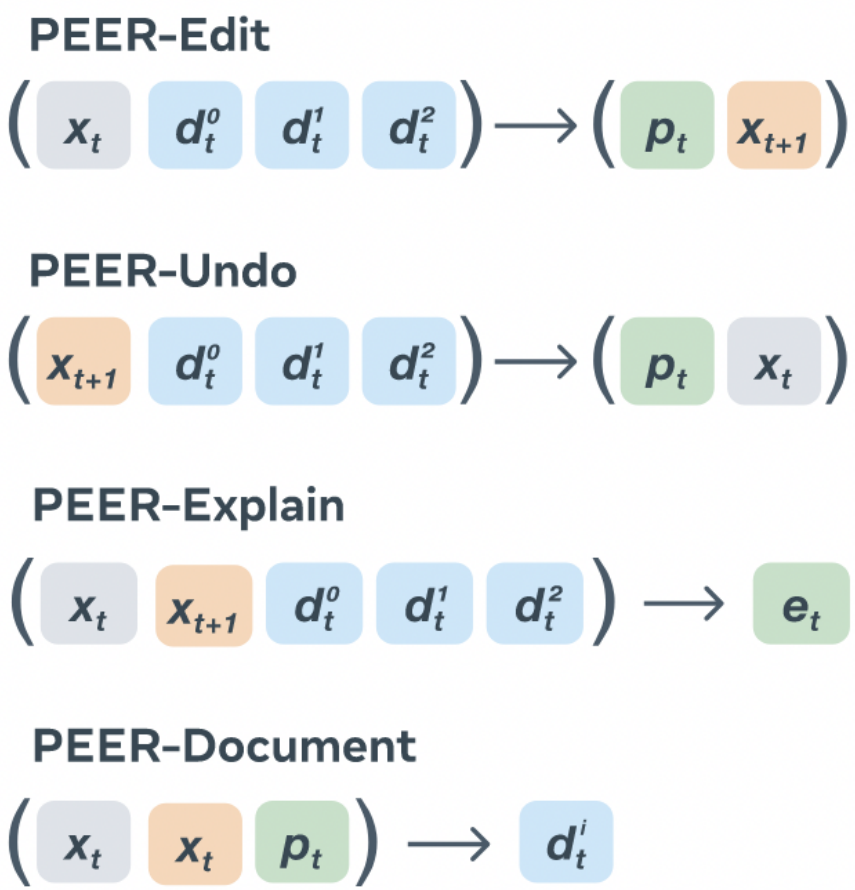
PEER actually comes into several model components :

- **PEER-Edit** : generate plan + edited text x_{t+1}.
- **PEER-Undo** : generate plan + former version of the text x_t (reverse generation from PEER-Edit).
- **PEER-Explain** : generate explanation e_t.
- **PEER-Document** : generate a document $d \in D_t$.

Each component is a separate T5-LM-Adapted 3B model.

Process

Input and output for each PEER component :



Data Generation

PEER can be used for data augmentation :

- Use PEER-Undo to generate "backward" edits until we get dummy text, then train PEER-Edit on the reversed sequence.
  $=>$**This is a form of self-training.**
  $=>$**Extremely similar to the diffusion models ?**
- Use PEER-Explain to get explanations :
  - Given an edit (x_t, x_{t+1}), sample explanations.
  - Keep the one with highest likelihood.
- Use PEER-Document to generate a document if there is no document for an edit.
  - Sample several documents.
  - Keep the one helping PEER-Edit the most.

Controlling Outputs

3 components use control mechanisms on the generated text :

- PEER-Explain :
  - Control output length (=>**how ?**)
  - Force the generated comment to start with a verb.
  - No word overlap between the explanation and the edit.
- PEER-Undo :
  - x_t forced to have strictly less words than x_{t+1}.
- PEER-Document :
  - Control whether the generated document contains a given substring.

Training Data

Data comes from **Wikipedia edit history**, enhanced with a few steps :

- **Filtering** : remove low-quality edits, prevent overlap with the validation sets.
- **Retrieval** :
  - If there are enough, get documents from citations of the edited paragraphs.
  - Otherwise, retrieve from the Sphere corpus.
- **Formatting**
  - Remove all paragraphs not affected by the edit.
  - Remove Wikipedia-specific syntax.
  - Keep title, bold/italic text, links, citations.
  - Linearize each document : [i] d_i # t_i # c_i

Goals

Conduct experiments to check :

- Can PEER **follow plans** and perform meaningful edits in domains for which no edit histories are available ?
- Does the ability to follow plans based on Wikipedia comments **transfer to instructions specified by humans**, and can it be improved by training on synthetic plans generated using PEER-Explain ?
- Can PEER make proper use of **citations and quotes** to explain generated outputs ?
- How does writing text sequences in a single pass compare to an iterative application of PEER ?

Training Details

- Each of the 4 PEER models is trained for 20,000 steps on 64 GPUs with an effective batch size of 256, corresponding to about five million Wikipedia edits.
- Max input length 1024, max output length 384.
- Retrieve $k = 3$ documents.

Evaluation Metrics

- **Exact Match (EM)** is the percentage of examples for which the performed edit exactly matches a given target.
- **EM-Diff** is a variant of EM that is computed on the diff level.
- **SARI** averages match scores for the three word-level edit operations add, delete and keep.
- **GLEU** is a variant of BLEU proposed for grammatical error correction tasks.
- **Rouge** is a set of metrics based on n-gram overlap (Rouge-n) or longest common subsequences (Rouge-L).
- **Update-Rouge** is a variant of Rouge that is computed only on sentences updated during an edit.

Custom Evaluation Set

They introduce *Natural Edits*, a collection of naturally occuring edits for different text types and domains obtained from three English web sources :

- **Encyclopedic pages** from Wikipedia
- **News articles** from Wikinews,
- **Questions** from the Cooking, Gardening, Law, Movies, Politics, Travel and Workplace subforums of StackExchange

Natural Edits

Composition of Natural Edits :

Subset	Train (Edit)	Train (PT)	Test	Doc.
Wikipedia	6,960,935	–	4,000	✓
Wikinews	–	125,664	1,000	–
Cooking	–	22,517	500	–
Gardening	–	13,258	500	–
Law	–	16,418	500	–
Movies	–	19,601	500	–
Politics	–	10,676	500	–
Travel	–	38,961	500	–
Workplace	–	18,231	500	–

Table 1: Overview of the number of edits and plain texts (PT) in the train sets and the number of edits in the test sets of Natural Edits. The final column shows whether the subset uses reference documents.

Ablation

Model	EM	EM-Diff	SARI
Copy	0.4	0.0	32.7
PEER	**23.1**	**26.2**	**55.5**
PEER (no plans)	18.0	19.8	52.0
PEER (no documents)	19.8	22.8	51.7
PEER (no plans/documents)	13.5	15.1	45.9

Table 2: Results for variants of PEER on the Wikipedia subset of Natural Edits. Plans and documents provide complementary information and substantially improve performance.

Using documents AND plans does help PEER.

Performance on Natural Edits

	Wiki	News	Cooking	Garden	Law	Movies	Politics	Travel	Workpl.
Copy	0.0 / 32.7	0.1 / 32.8	0.0 / 31.6	0.0 / 32.0	0.0 / 31.1	0.0 / 31.5	0.0 / 31.8	0.0 / 31.2	0.0 / 31.5
PEER (no plans)	16.6 / 50.7	10.8 / 41.3	4.5 / 36.3	1.8 / 35.1	2.6 / 35.8	2.9 / 35.3	2.1 / 36.5	1.6 / 34.8	3.1 / 34.7
PEER	**26.2 / 55.5**	21.3 / 49.3	11.0 / 40.2	4.4 / 37.7	7.5 / 36.4	6.7 / 39.2	6.8 / 38.7	6.7 / 38.1	6.9 / 36.7
PEER (DA)	–	**23.3 / 51.6**	**13.2 / 42.9**	**8.1 / 44.9**	**9.4 / 39.0**	**9.9 / 42.4**	**11.6 / 41.3**	**9.1 / 40.2**	**8.3 / 39.2**

Table 3: EM-Diff / SARI scores on all subsets of Natural Edits. The domain-adapted (DA) variants of PEER clearly outperform regular PEER, demonstrating the usefulness of synthetic edits generated with PEER-Undo.

PEER (DA) is a domain-adapted version : generate synthetic edits by one round of PEER-Undo coupled with PEER-Explain ; fine-tune PEER-Edit on a mixture of original data + synthetic one.

Downstream Tasks

- **JFLEG** is a grammatical error correction dataset with single-sentence inputs written by English language learners.
- **ASSET** is a crowdsourced corpus for single-sentence text simplification.
- **ITERATER** is an editing dataset spanning five edit intentions across three different domains.
- **WNC** is a dataset where the task is to remove or mitigate biased words to make sentences more neutral.
- **FRUIT** contains texts from Wikipedia that need to be updated ; for performing this update, various reference documents from Wikipedia are provided.
- **WAFER-INS** is based on the WAFER dataset the task is to insert a sentence at some position in a Wikipedia paragraph given documents from the Sphere corpus that contain relevant background information.

Downstream Tasks

	Model	Params	Without Documents JFLEG	ASSET	ITERATER	WNC	With Documents FRUIT	WAFER-INS	Avg
	Copy	–	26.7 / 40.5	20.7	30.5	31.9 / 0.0	29.8 / 0.0	33.6 / –	28.9
	T*k*-Instruct	3B	31.7 / 38.7	28.3	36.2	30.3 / 0.0	12.7 / 3.9	1.6 / –	23.5
	T0	3B	42.9 / 38.6	28.6	28.1	17.8 / 0.0	13.1 / 5.7	6.1 / –	22.8
(a)	T0++	11B	35.9 / 43.8	25.8	36.1	27.0 / 0.0	16.1 / 3.7	3.9 / –	24.1
	PEER	3B	54.8 / 55.1	29.9	36.5	56.4 / 31.9	39.4 / 28.3	35.2 / 33.6	42.0
	PEER (SP)	3B	59.0 / 57.2	**33.2**	37.1	56.6 / 32.7	40.3 / **33.9**	35.5 / 37.6	43.6
	PEER (SP)	11B	**59.9 / 58.6**	32.4	**37.8**	**58.8** / **34.7**	**40.7** / 33.5	**35.9** / **38.4**	**44.3**
(b)	PEER (SP, i3)	3B	63.3 / 59.6	36.1	37.1	45.2 / 12.4	41.6 / 34.6	35.2 / 37.0	43.1
	PEER (SP, s-i3)	3B	57.4 / 49.7	40.7	35.8	38.4 / 3.9	41.6 / 38.7	32.9 / 34.3	41.1
	OPT	175B	49.2 / 49.4	25.8	31.4	25.1 / 0.0	35.6 / 27.4	21.1 / –	31.4
(c)	GPT3	175B	50.6 / 51.8	25.0	30.7	26.0 / 0.5	33.6 / 25.9	22.9 / –	31.5
	InstructGPT	175B	62.3 / 60.0	35.4	38.2	33.9 / 0.7	37.5 / 23.4	29.2 / –	39.4
(d)	Sup. SotA	–	– / *62.4*	*44.2*	*37.2*	– / *45.8*	– / *47.4*	–	–

Table 4: Downstream task results for PEER and various baselines, divided into four groups: **(a)** T5-based models and a copy baseline, **(b)** PEER with different sampling strategies, **(c)** 175B parameter decoder-only models, **(d)** supervised state of the art. The first numbers for each task are SARI scores; additional metrics are GLEU for JFLEG, EM for WNC, Update-R1 for FRUIT and SARI scores obtained if the model is told exactly where to insert a new sentence for WAFER-INS. Supervised scores from left to right are from Ge et al. (2018), Martin et al. (2020), Du et al. (2022b), Pryzant et al. (2020) and Logan IV et al. (2021), respectively. The best result for models based on *LM Adapted* T5 is shown in bold, the best zero-shot performance overall is underlined. On average, PEER (SP) clearly outperforms all baselines.

Citations and Quotes

Two datasets derived from Natural Edits for each task : NE-Cite and NE-Quote, corresponding to adding a citation or a quotation, respectively.

Model	NE-Cite	NE-Quote	NE-Quote (con.)
Random	– / 33.3	–	40.1 / 31.7 / 36.5
Unigram	– / 34.2	–	–
Side	– / **91.1**	–	–
Lead	– / –	–	50.6 / 44.0 / 46.0
PEER	74.1 / 88.1	0.0 / 0.0 / 0.0	49.3 / 44.3 / 48.1
PEER (SP)	74.5 / 88.9	0.2 / 0.1 / 0.1	49.8 / 44.8 / 48.7
PEER (SQ)	**74.9** / 87.9	**13.6 / 11.9 / 12.9**	**58.1 / 54.6 / 57.3**

Table 5: Accuracy on NE-Cite (without/with gold positions) and R1/R2/RL scores on both NE-Quote and constrained NE-Quote. When given the correct position, PEER (SP) almost matches the performance of the supervised Side model on NE-Cite, demonstrating its strong citing abilities. Training on synthetic documents substantially improves PEER's ability to quote relevant passages.

Iterative Editing for Text Generation

Models :

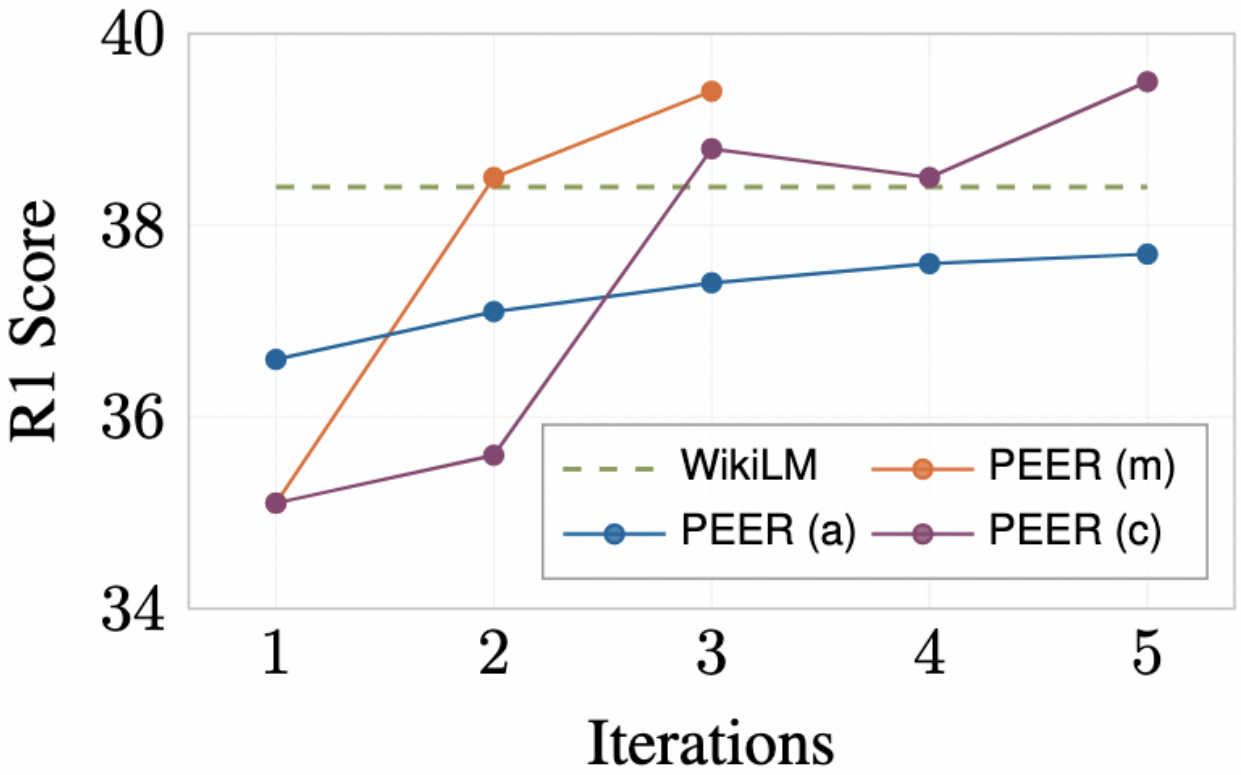
- **WikiLM** : T5-LM-Adapted fine-tuned as conditional language model.
- **PEER (autonomous)** : the model continuously writes and realizes its own plans without human involvement.
- **PEER (manual)** : give the model a series of human-written plans.
- **PEER (collaborative)** : human-written plans are interleaved with plans proposed by PEER.

Iterative Editing for Text Generation

Model	LP	R1 / R2 / RL	QE
Wiki-LM	5.0	38.4 / 16.9 / 27.3	38.7
PEER (autonomous)	5.0	37.7 / 15.8 / 26.2	40.6
PEER (manual)	2.0	39.4 / 17.0 / 28.1	**41.1**
PEER (collaborative)	2.0	**39.5 / 17.2 / 28.4**	41.0

Table 6: Results for various approaches on our Wikipedia intro generation test set. Length penalty (LP) is optimized on the dev set; scores shown are Rouge-1/2/L and QuestEval (QE). WikiLM performs better than autonomous PEER in terms of Rouge scores, but is outperformed by PEER in manual and collaborative mode; all PEER models perform better in terms of QuestEval.

Iterative Editing for Text Generation



PEER benefits from several rounds.

Conclusion

- A system enabling collaborative writing through 4 models, each with a specific task : edit, undo, explain, generate a document.
  - Can perform edits in different domains.
  - Better at following instructions.
  - Good citing and quoting performance.
- Limitations :
  - A major limitation is training 4 3B-params models.
  - Retrieval assumes access to the target.
- Very clever to use PEER-Undo for data augmentation and self-training.
  Inspiration for other NLP tasks ?