

Applications

Applications

- ❖ Text Editing methods:
 - task-specific (e.g., GECToR)
 - general-purpose (e.g., LaserTagger)
 - general with task-specific modifications & tricks (e.g., Seq2Edits, PIE)

Applications

- sentence fusion
- sentence splitting & rephrasing
- text normalization
- text summarization
- machine translation automatic post-editing
- grammatical error correction
- text style transfer
- incomplete utterance rewriting
- text simplification

Defined briefly next

Discussed in more detail

Sentence Fusion

- Joining several independent sentences into a single coherent text.

As a run-blocker, **Zeitler** moves relatively well.

Zeitler too often struggles at the point of contact in space.



As a run-blocker, **Zeitler** moves relatively well. **However, he** too often struggles at the point of contact in space.

anaphora

discourse connective

Example taken from: Geva, M. et al. (2019). *DiscoFuse: A Large-Scale Dataset for Discourse-Based Sentence Fusion* ([pdf](#))

Barzilay, R., & McKeown, K. R. (2005). *Sentence fusion for multidocument news summarization*. ([pdf](#))

Applications of text editing include: EdiT5, [Felix](#), [LaserTagger](#), [Masker](#), [Seq2Edits](#)

Sentence Splitting

- In a sense, a reverse problem to sentence fusion
- Similar to text simplification, but lots of data available

A classic leaf symptom is water-soaked lesions between the veins **which appear as** angular **leaf-spots** where the lesion edge and vein meet.

A classic leaf symptom is **the appearance of angular, water-soaked lesions between the veins.** **The angular appearance results** where the lesion edge and vein meet.

Botha et al. (2018) Learning To Split and Rephrase From Wikipedia Edit History ([pdf](#))

Applications of text editing include: [LaserTagger](#)

Text normalization

- Converting written text into its spoken verbalization
- spelling out numerals, currencies, phone numbers, dates, etc.

A baby giraffe is 6ft tall and weighs 150lb.



A baby giraffe is six feet tall and weighs one hundred and fifty pounds.

Sproat, R. & Jaitly, N. (2016) *RNN Approaches to Text Normalization: A Challenge* ([pdf](#))

Applications of text editing include: [Seq2Edits](#)

Text Summarization / Compression

- See *Dernoncourt et al., 2018* ([pdf](#)) for an overview
- Earlier works on extractive summarization ([Filippova et al., 2015](#)) can be seen as predecessors of text-editing models

Summarize this for a second-grade student:

Jupiter is the fifth planet from the Sun and the largest in the Solar System. It is a gas giant with a mass one-thousandth that of the Sun, but two-and-a-half times that of all the other planets in the Solar System combined. Jupiter is one of the brightest objects visible to the naked eye in the night sky, and has been known to ancient civilizations since before recorded history. It is named after the Roman god Jupiter.[19] When viewed from Earth, Jupiter can be bright enough for its reflected light to cast visible shadows,[20] and is on average the third-brightest natural object in the night sky after the Moon and Venus.

Sample response:

Jupiter is a planet that is bigger than all the other planets in our solar system and is very bright when you see it in the night sky. It is named after the Roman god Jupiter. When viewed from Earth, it is usually one of the three brightest objects in the sky.

Text source: <https://beta.openai.com/examples/default-summarize>

Applications of text editing include: [LaserTagger](#)

Machine Translation Automatic Post Editing (APE)

- Task: Refine the output of an MT system

Source	Does not have a menu bar .
MT	Weist ₁ keine ₂ Menüleiste ₃ angezeigt ₄ . ₅
Reference	Weist ₁ keine ₂ Menüleiste ₃ auf ₄ . ₅
APE prog.	KEEP ₁ KEEP ₂ KEEP ₃ auf ₄ DEL ₄ KEEP ₅ STOP

Source: Vu, T., Haffari, G. (2018) [Automatic Post-Editing of Machine Translation: A Neural Programmer-Interpreter Approach](#). EMNLP 2018

Applications of text editing include: [Felix](#), [LevT](#)



Grammatical Error Correction

Grammatical Error Correction Task

Source: *She no drives to market.*

Target: *She did ~~no~~ not ~~drives~~ drive to market.*

Text Editing for Grammatical Error Correction

- Text Editing for GEC has picked up in the **last couple of years**
- Competitive or **SOTA results** on public benchmarks (CoNLL-14, BEA-19)
- Main highlight: ~10x **inference speedup**
- Allows for task-specific modifications

Applications of text editing include: EdiT5, [Felix](#), [GECToR](#), [LaserTagger](#), [PIE](#), [Seq2Edits](#)

Custom edit operations for GEC

- suffix transformations

PIE [1] uses a total of
58 suffix transformations

ADDSUFFIX(s):

play → plays

CHANGE-d-TO-t:

spend → spent

- help the model generalize
- learnt from the data

[1] Awasthi A. et al. (2019) *Parallel iterative edit models for local sequence transduction*.

Custom edit operations for GEC

- higher-level operations

GECToR [2] introduces
29 "g-transformations"

CASE_LOWER: Medical → **medical**

MERGE_HYPHEN: in depth → in-depth

VERB_FORM_VB_VBZ: make → **makes**

- help the model generalize
- implementation relies on linguistic resources
(e.g., verb conjugation dictionary)

[2] Omelianchuk K. et al. (2020) *GECToR--grammatical error correction: tag, not rewrite*.

Custom edit operations for GEC

- character transformations, e.g.:

Straka et al. [3]

REPLACE_3RD_FROM_END_WITH_v: leafes → lea**v**es

- learnt from the data

Predicting error types

- Seq2Edits:
 - additionally predicts the **type** of grammatical error.
 - task-specific error types (ERRANT [1])
(25 main error types: spelling, punctuation, noun inflection, verb form, etc.)

Model size	Tags	Tuning	Pre-training	Grammar (BEA-dev)			
				P↑	R↑	F _{0.5} ↑	
a Base				23.3	11.0	19.0	
b Base	✓			22.5	13.3	19.8	↪
c Big			✓	50.1	34.4	45.9	↪
d Big	✓		✓	53.7	35.3	48.6	↪
e Big		✓	✓	49.0	38.6	46.5	↪
f Big	✓	✓	✓	50.9	39.1	48.0	↪

Source: Stahlberg F. & Kumar S. (2020)
Seq2edits: Sequence transduction using span-level edit operations.

[1] Bryant C. et al. (2017) *Automatic annotation and evaluation of error types for grammatical error correction.*

Non-specific tricks useful for GEC

- pre-training on synthetic data
- iterative refinement
- model ensembling
- tweaking the probabilities of KEEP at token or sequence-level



Text Style Transfer with Unsupervised Text Editing

Levels of pre-training

- Pre-trained encoder (e.g. GECToR, LaserTagger)
- Pre-trained encoder + insertion component (e.g. Felix, EdiT5)
- Pre-trained encoder + insertion component + tagging component (e.g. Masker)

Style transfer

Setting: Only **non-parallel** source and target examples available.

Goal: Transfer a source sentence to target style.

Example:

Source: *The best place I've visited!*
Target: *The worst place I've visited!*

Often decomposed into two subtasks:

1. Determine which words to delete
2. Determine how to replace them

Applications of text editing include: [LEWIS](#), [Masker](#)

Text-Editing with Masked Language Models

1. Train a Masked Language Model (MLM) on the **target** data.
2. **Mask out each input word** one-by-one to compute their **likelihoods**.
3. **Replace low-likelihood words** with the top prediction by the MLM.

Sentiment Transfer Example

From **negative** (source) to **positive** (target).

price isn't bad while food is the worst in Zürich

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 1$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 2$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 3$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = ???$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 3$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 4$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 5$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 6$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 7$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 8$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = ???$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 8$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the ~~worst~~ in Zürich

best

$i = 8$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$

$$\arg \max P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{"best"}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 9$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{high}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 10$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = ???$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 10$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 10$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$

Source MLM
(negative)

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the worst in Zürich

$i = 10$

$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$

$$P(W_i | W_{\setminus i}, \Theta_{\text{source}}) = \text{low}$$

Source MLM
(negative)

Target MLM
(positive)

Sentiment Transfer Example

price isn't bad while food is the **worst** in Zürich

$i = 8$

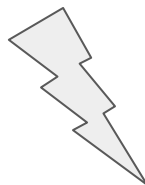
$$P(W_i | W_{\setminus i}, \Theta_{\text{target}}) = \text{low}$$
$$P(W_i | W_{\setminus i}, \Theta_{\text{source}}) = \text{high}$$

Source MLM
(negative)

Target MLM
(positive)

Masker's key idea:

When *Target* and *Source MLM* disagree, we should DELETE.



Masker for Pre-training

Single input requires masking out every n -gram ($n \in \{0,1,2\}$) and running BERT inference → **slow but parallelizable**

Use Masker to create a **silver data to pretrain** a more efficient model.

Masker results: Sentence Fusion

Method	Parallel training examples			
	0	450	4500	45000
LASERTAGGER	0.00	12.32	25.74	38.46
+ MASKER silver data	19.61	25.97	34.20	42.41
FELIXINSERT	0.00	15.34	34.11	46.09
+ MASKER silver data	18.22	25.23	38.43	47.21
BERT2BERT	0.00	0.00	3.35	42.07
+ MASKER silver data	13.05	16.57	30.14	43.03
Average improvement	16.96	13.37	13.19	2.01

Table 2: Low-resource sentence fusion results. Using the predictions of MASKER as silver data to pretrain models improves the Exact score.

LEWIS (Reid & Zhong, 2021) - Model Architecture

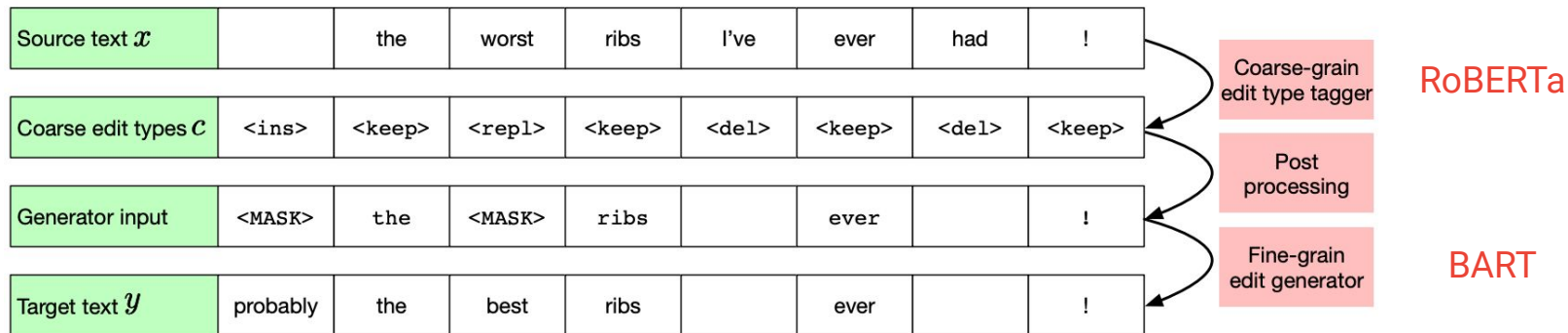
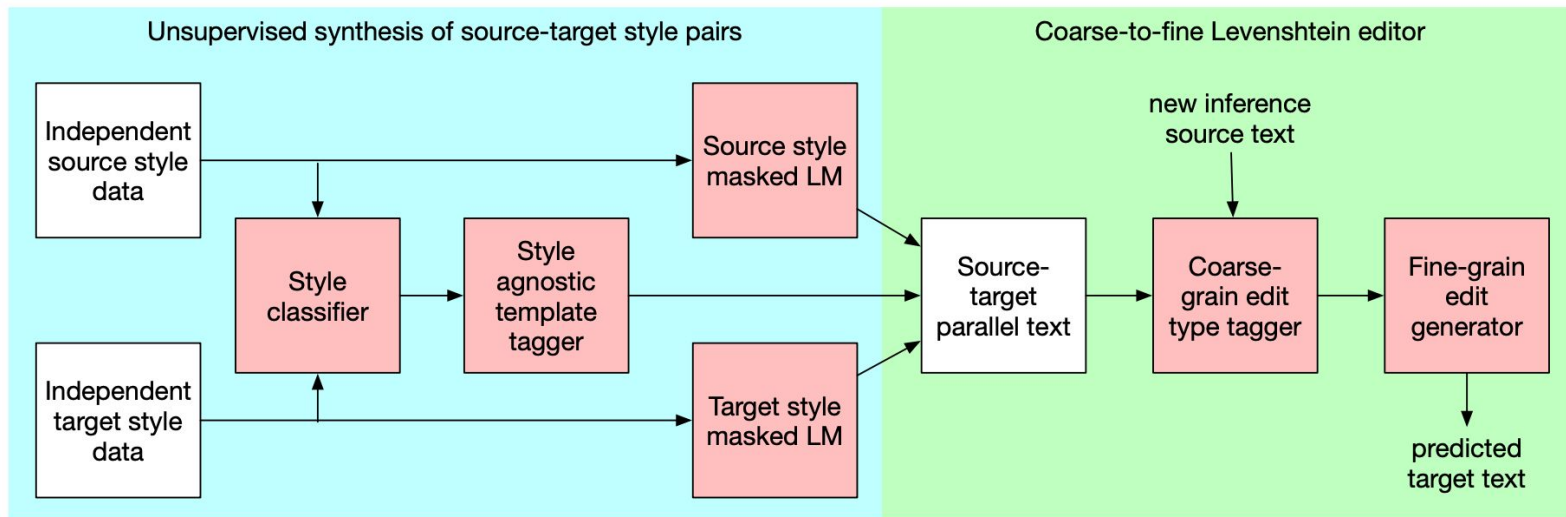


Figure 1: Coarse-to-fine Levenshtein editor. Given the source text, the two-step editor first generates coarse edit types via a tagger. A subsequent generator fills in insertions and replacements while taking into account the source text and the edit types.

Source: Reid, M., Zhong, V. [LEWIS: Levenshtein Editing for Unsupervised Text Style Transfer](#). Findings of ACL-IJCNLP 2021.

LEWIS - Unsupervised Data Generation



Source: ([Reid & Zhong, 2021](#))

LEWIS - Unsupervised Data Generation



Source: ([Reid & Zhong, 2021](#))

LEWIS - Experimental Evaluation

	Acc	SBLEU	BLEU	SBERT	BERT
Editing methods					
Masker (Malmi et al., 2020)	40.9 [†]	—	14.5	—	—
LaserTagger (Malmi et al., 2019) + Masker data	49.6 [†]	—	15.3	—	—
LaserTagger + our data	59.8	71.8	24.8	81.3	51.6
LEWIS	93.1	58.5	24.0	72.2	50.0

Table 2: Results on YELP. Results with [†] are taken from the classifier trained in [Malmi et al. \(2020\)](#) because the outputs for these models are not released.

Source: ([Reid & Zhong, 2021](#))

⇒ distilling Masker to LaserTagger improves quality
(LaserTagger regularizes rewrites)

Incomplete Utterance Rewriting

Incomplete Utterance Rewriting - Motivation






Google AI Blog, <https://ai.googleblog.com/2022/05/contextual-rephrasing-in-google.html>

Task Introduction

Setting: Open-Domain Dialog between User and Agent.




Aka.: *Conversational Query Rewriting, Question-in-context rewriting*

Speaker	Utterance
	Why did Federer withdraw from the tournament?
	He injured his back in yesterday's match.
	Did he have any other injuries?
---	Did Federer have any other injuries besides his back?

Task Introduction

Setting: Open-Domain Dialog between User and Agent.

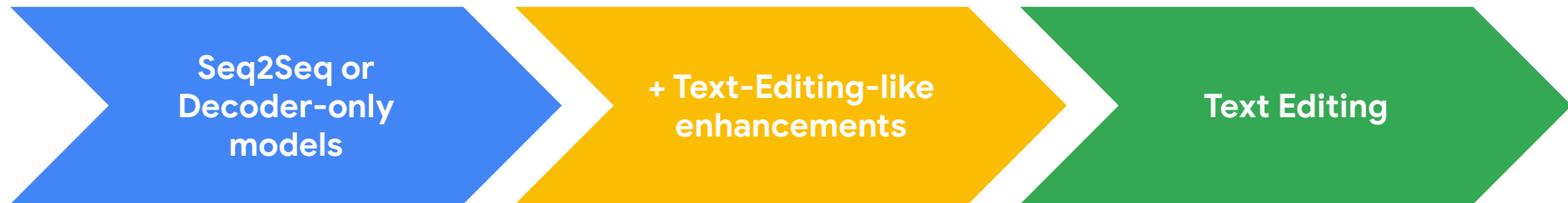
Aka.: *Conversational Query Rewriting, Question-in-context rewriting*

Speaker	Utterance
	Why did Federer withdraw from the tournament?
	He injured his back in yesterday's match.
	Did he have any other injuries?
---	Did Federer have any other injuries besides his back?

Core elements of the task

- **Pass over last utterance** (by generating it, by rewriting it)
 - **KEEP** most words.
 - **REPLACE** some words with phrases from context.
 - **INSERT** some phrases from context.
- **Distribution of anaphora** in corpus of *Regan et al. (2019)* [[pdf](#)]
 - 62% zero anaphora/ellipsis
 - 19% pronoun resolution
 - 10% locative references
 - 8% nominal references
- **Observation:** *Insertion of arbitrary phrases usually not needed!*
 - Pretty ideal scenario for text-editing approaches

Approaches



Ren et al., WWW 2018 [[pdf](#)]

Yu et al., SIGIR 2020 [[pdf](#)]

Vakulenko et al., WSDM 2021 [[pdf](#)]

Rastogi et al., NAACL 2019 ([pdf](#))

Su et al., ACL 2019 ([pdf](#))

Pan et al., EMNLP 2019 ([pdf](#))

Zhou et al., EMNLP 2019 ([pdf](#))

Quan et al., EMNLP 2019 ([pdf](#))

Liu et al., EMNLP 2019 ([pdf](#))

Liu et al., EMNLP 2020 ([pdf](#))

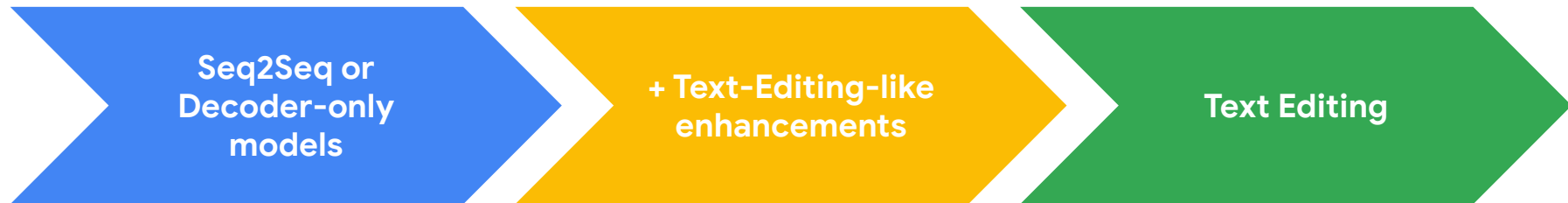
Huang et al., AAAI 2021 ([pdf](#))

Hao et al., EMNLP 2021 ([pdf](#))

Jin et al., AAAI 2022 ([pdf](#))

Zhang et al., ICASSP 2022 ([pdf](#))

Approaches



Ren et al., WWW 2018 [[pdf](#)]

Yu et al., SIGIR 2020 [[pdf](#)]

Vakulenko et al., WSDM 2021 [[pdf](#)]

Rastogi et al., NAACL 2019 ([pdf](#))

Su et al., ACL 2019 ([pdf](#))

Pan et al., EMNLP 2019 ([pdf](#))

Zhou et al., EMNLP 2019 ([pdf](#))

Quan et al., EMNLP 2019 ([pdf](#))

Liu et al., EMNLP 2019 ([pdf](#))

Liu et al., EMNLP 2020 ([pdf](#))

Huang et al., AAAI 2021 ([pdf](#))

Hao et al., EMNLP 2021 ([pdf](#))

Jin et al., AAAI 2022 ([pdf](#))

Zhang et al., ICASSP 2022 ([pdf](#))



Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]

Characteristics:

1. Flexible insertion before each source token from non-contiguous context spans

Reordering possible!

2. "Out-of-context" insertions possible from limited vocabulary

"Slotted rules"

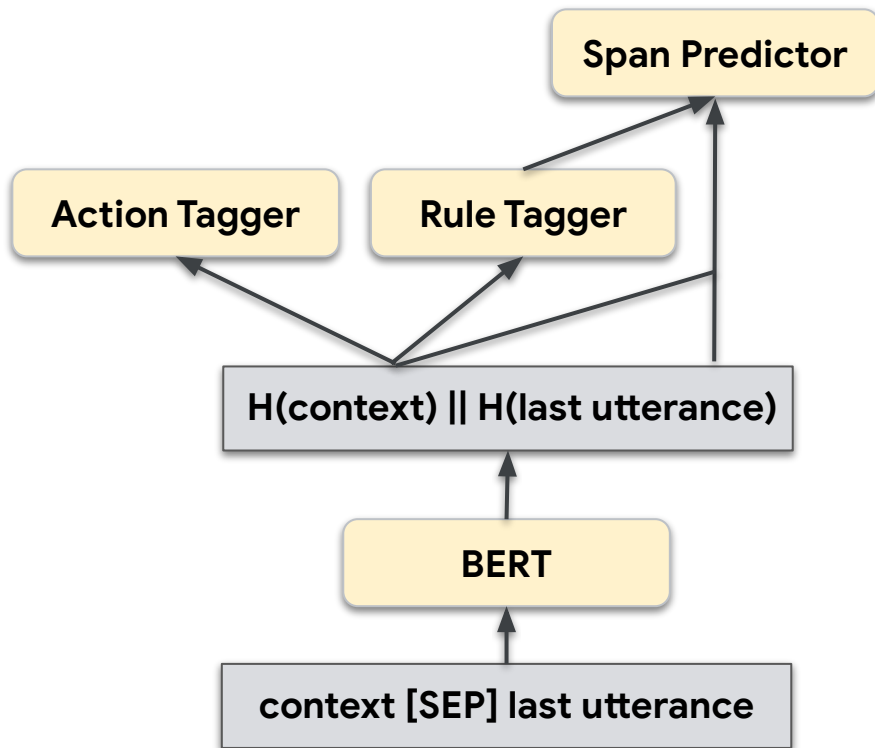
1		—	9		— by —	17		for —
2		— 's	10		— than —	18		in —
3		— —	11		— than — —	19		in — — —
4		— — ,	12		aside from —	20		of —
5		— — 's	13		besides —	21		other than —
6		— — 's —	14		besides — —	22		other than — — —
7		— — —	15		besides — — —	23		the — —
8		— — — 's	16		by —	24		to —

Table 6: HCT rule vocabulary for CANARD.

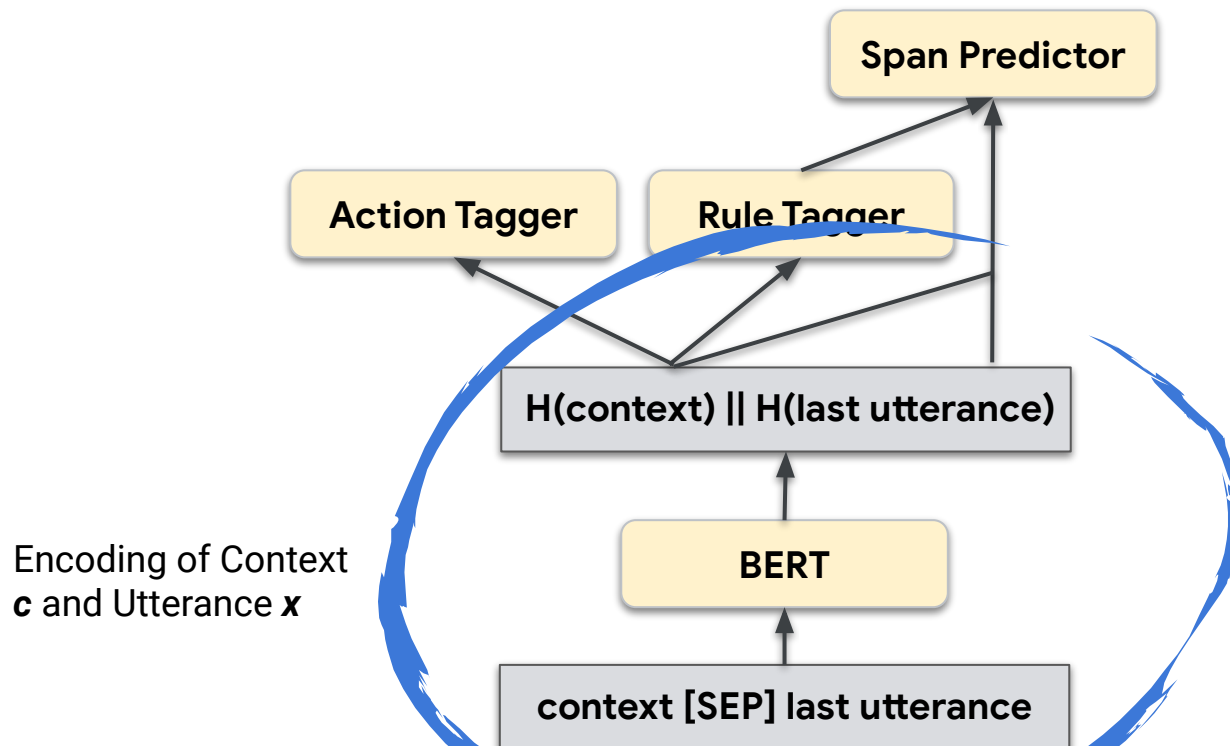
1		—	5		— —	9		okay
2		— 's	6		— — 's	10		song
3		— , —	7		of —	11		the —
4		— , — ,	8		of — —	12		the — —

Table 7: For MuDoCo.

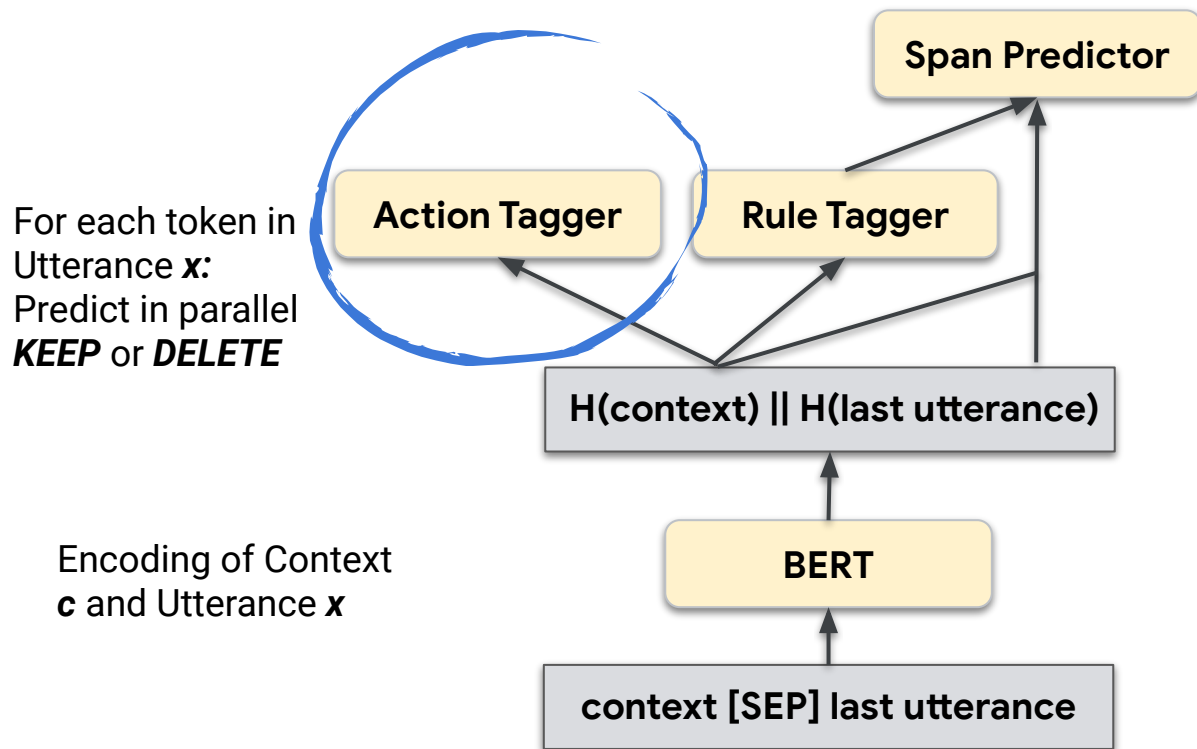
Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]



Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]



Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]



Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]

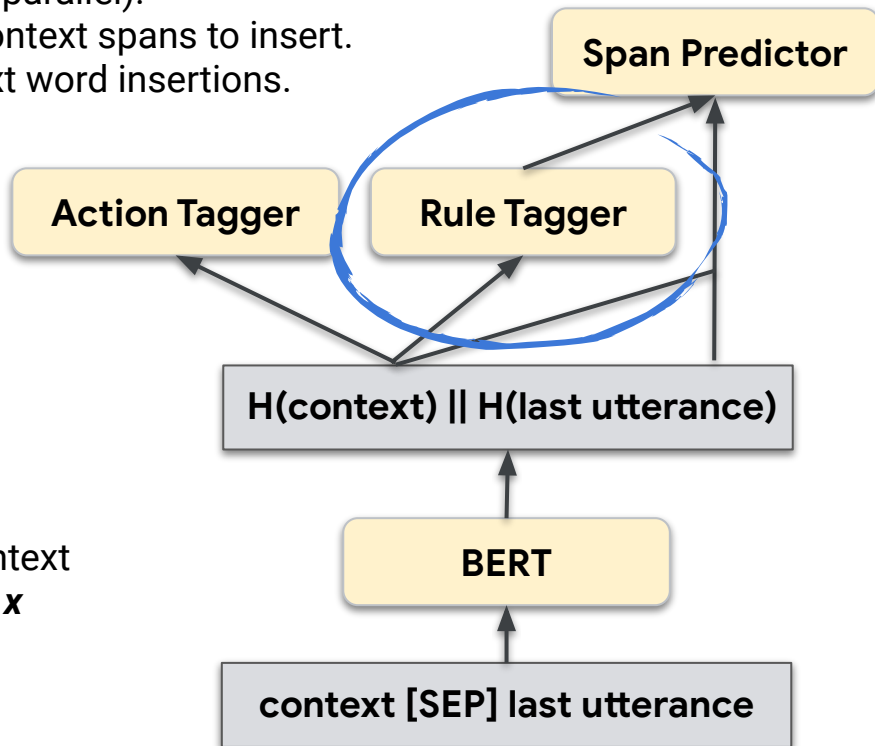
For each token in Utterance x :

Predict a rule (in parallel).

(a) Number of context spans to insert.

(b) Out-of-context word insertions.

For each token in
Utterance x :
Predict in parallel
KEEP or **DELETE**



Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]

For each token in Utterance x :

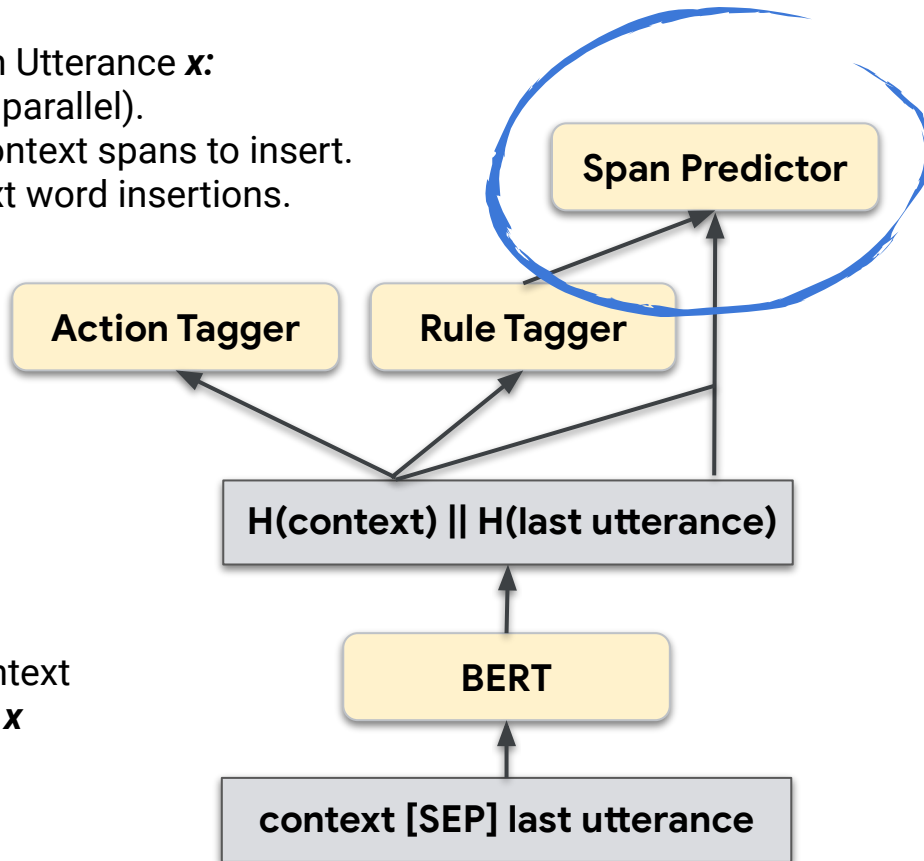
Predict a rule (in parallel).

(a) Number of context spans to insert.

(b) Out-of-context word insertions.

For each token in
Utterance x :
Predict in parallel
KEEP or **DELETE**

Encoding of Context
 c and Utterance x



"_ and _"



[SL0] and [SL1]



[SL0]: start = ..., end = ...
[SL1]: start = ..., end = ...
(auto-regressive)

Hierarchical Context Tagging for Utterance Rewriting [Jin et al. 22]

Context c	why	did	federer	withdraw	...	?	[SEP]	he	injured	his	back	...
	1	2	3	4	5-7	8	9	10	11	12	13	14-18
Source x	1	2		1	1	1	1					3
Rewrite x^*	did	he		have	any	other	injuries					?
	did	<u>federer</u>		have	any	other	injuries	besides	<u>his</u>	<u>back</u>		?
		3							12	13		

Span start	1	2	3	n
Span end	1	2	3	n
Action	K	D		
Rule	1	2	3	n
	∅	—	besides	—



Text Simplification

Text Simplification (TS) Task

The process of transforming/rewriting a text into an **equivalent** which is **simpler** to understand by a target audience

Last year, I read the book
that is authored by Jane.
[Original sentence]



Jane wrote a book.
I read it last year.
[One or several simpler sentences]

Text Simplification (TS) Task: Substitution

Multiple Operations:

1. **Word or phrase substitution**
2. Sentence splitting
3. Deletion
4. Syntactic style transfer

Last year, I read the book
that is **authored** by Jane.

[Original sentence]



TS

Jane **wrote** a book.

I read it last year.

[One or several simpler sentences]

Text Simplification (TS) Task: Splitting

Multiple Operations:

1. Word or phrase substitution
2. **Sentence splitting**
3. Deletion
4. Syntactic style transfer

[S1] Last year, I read the book
that is authored by Jane.

[Original sentence]



TS

[S1] Jane wrote a book.

[S2] I read it last year.

[One or several simpler sentences]

Text Simplification (TS) Task: Deletion

Multiple Operations:

1. Word or phrase substitution
2. Sentence splitting
3. **Deletion**
4. Syntactic style transfer

Last year, I read the book
that is authored by Jane.

[Original sentence]



TS

Jane wrote a book.

I read it last year.

[One or several simpler sentences]

Text Simplification (TS) Task: Style Transfer

Multiple Operations:

1. Word or phrase substitution
2. Sentence splitting
3. Deletion
4. **Syntactic style transfer**

Last year, I read the book
that is authored by Jane.

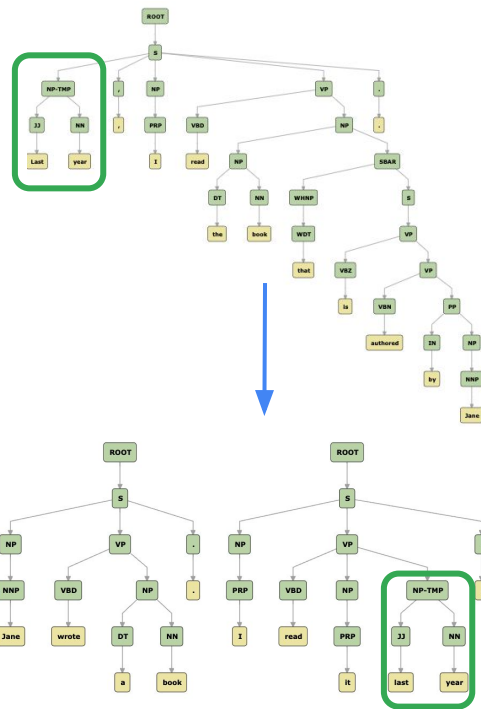
[Original sentence]

TS

Jane wrote a book.

I read it last year.

[One or several simpler sentences]



Text Editing for Text Simplification

- MT-based models have a large degree of generation freedom
- Edit-based model bounds the number of edits
 - **Controlled** generation (e.g. ratio of edits)
 - Competitive or **SOTA results** on the public benchmarks
 - Advantageous in **preserving facts** (Fact-based Text Editing, ACL 2020)
- Papers reporting on TS experiments:
 - SL ([Alva-Manchego et al., 2017](#))
 - EditNTS ([Dong et al., 2019](#))
 - RM ([Kumar et al., 2020](#))
 - Felix ([Mallinson et al., 2020](#))

Experiments for Text Simplification

Datasets

(supervised, document - summary pairs)



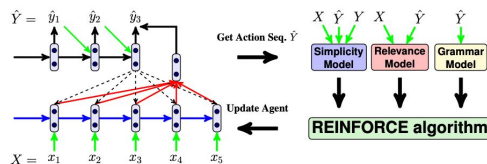
WikiLarge & Small

296,402/2000/359
& 88,837/205/100

newsela

94,208/1129/1076

Models



DELETE (D), REPLACE
(R), MOVE (M), ADD (A),
REWRITE (RW)

Baselines: e.g. [DRESS](#)
(Zhang and Lapata, 2018)

Edit-based models

Evaluation:

- SARI (Xu et al., 2016): Measure similarity to both input and reference sentence
- Human judges rate based on fluency, adequacy, simplicity (a five-point Likert scale)

SL: Sequence Labeling (2017)

Features:

- First neural edit-based model on simplification
- Heavily rely on heuristics and other modules for operations

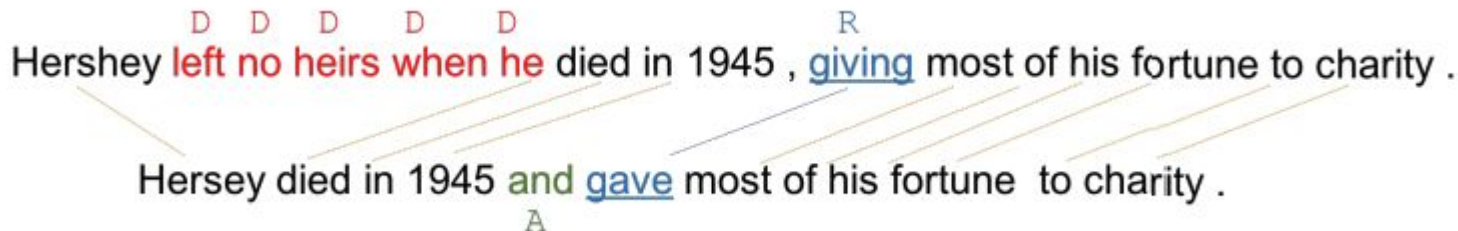
Five operations:

- On original sentence: **DELETE (D)**, **REPLACE (R)**, **MOVE (M)**
- **ADD (A)** in the simplified sentence
- **REWRITE (RW)**

SL Data Construction

Word alignments:

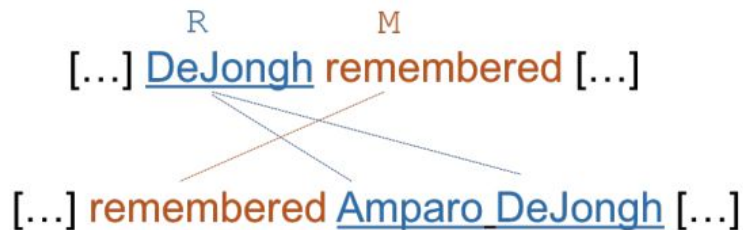
- between the original (x) and simplified sentences (y)
- **DELETE** : in x, not in y
- **REPLACE**: in x and y, but different
- **ADD (A)**: in y, not in x



SL Data Construction (Cont.)

In addition:

- **REWRITE (RW)**: special cases of **REPLACE** (1-N, N-1...)
- **MOVE (M)**: cross align



[Alva-Manchego et al., 2017](#)

Train the model on these **silver labels**:

DELETE (D), **REPLACE (R)**, **MOVE (M)**, **ADD (A)**, **REWRITE (RW)**

SL Inference

1. Predict operation:
 - DELETE (D), REPLACE (R), MOVE (M), ADD (A), REWRITE (RW)
2. Only consider two operations DELETE (D), REPLACE (R)
 - DELETE (D): Directly delete
 - REPLACE: External Simplification Dictionary [Paetzold and Specia (2017)]

	G	M	S
Reference	5.00±0.0	4.45±0.9	2.70±1.3
SL	4.16±1.0	3.91±1.1	1.66±0.9
Nematus	4.49±0.9	3.99±1.2	1.46±0.9
Moses	4.98±0.2	4.99±0.1	1.14±0.4
NTS	4.75±0.6	4.08±1.26	1.53±1.0
Fleiss' Kappa	0.372	0.457	0.342

[Alva-Manchego et al., 2017](#)

Newsela:

Promising human evaluation (higher = better):

- Grammaticality (G)
- Meaning preservation (M)
- Simplicity (S)

Better than NTS (seq2seq) in Simplicity (S)

EditNTS Training (2019)

End-to-end neural model:

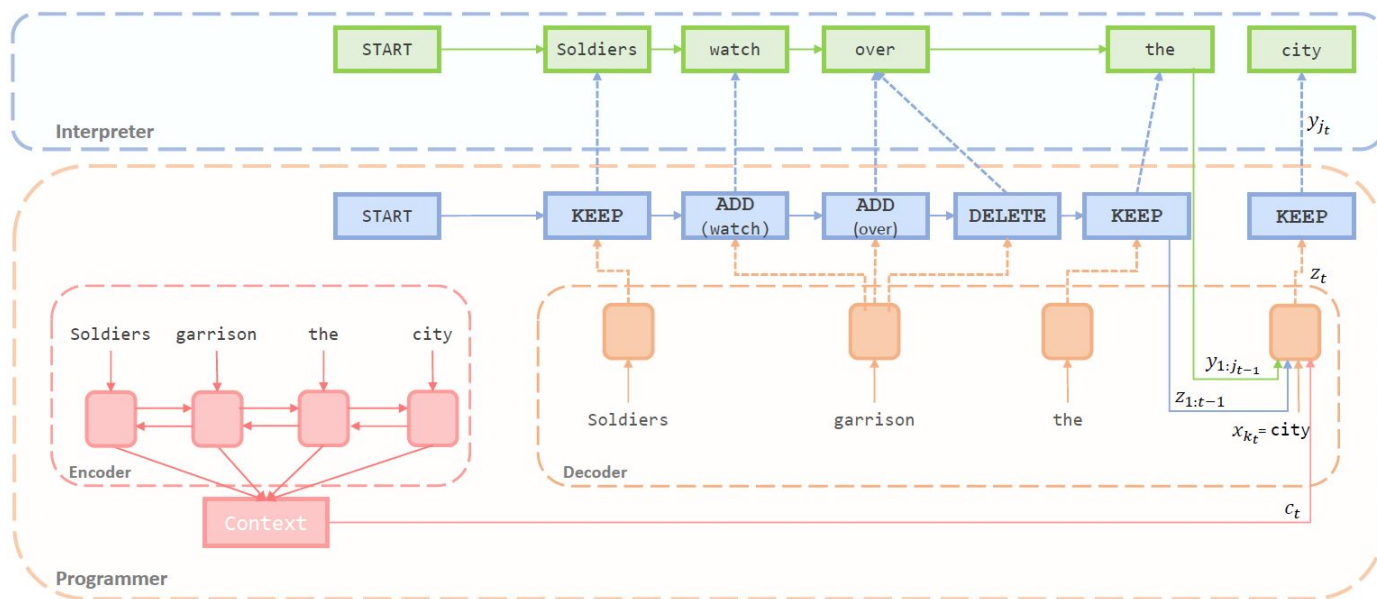
1. Create edit labels explicitly:
 - through three types of edits (z): ADD, DEL, and KEEP
2. New training objective function
 - learn $p(z|x)$



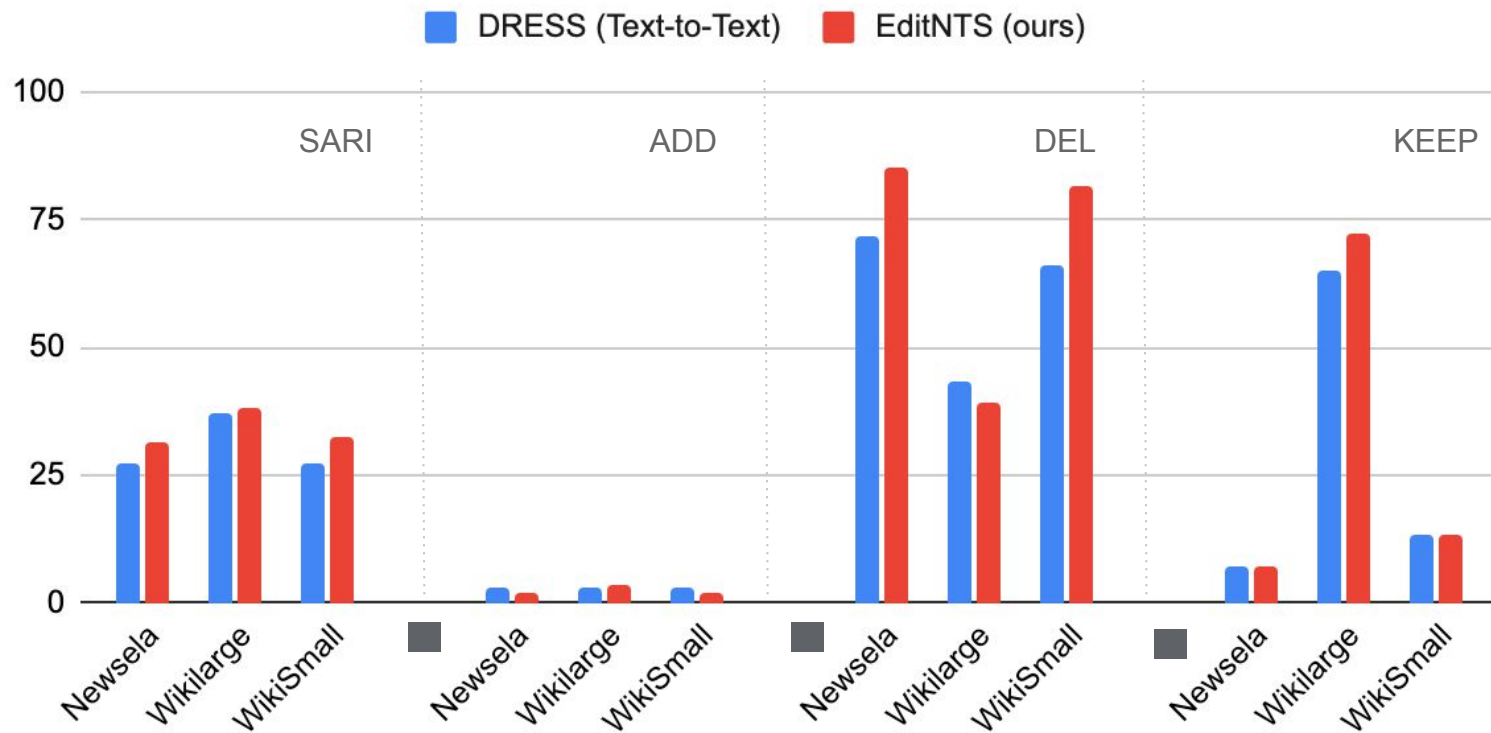
EditNTS: NPI for Modeling $p(\mathbf{z}|\mathbf{x})$

A neural programmer-interpreter (NPI)

$$P(\mathbf{z}|\mathbf{x}) = \prod_{t=1}^{\mathbf{z}} P(z_t | y_{1:j_{t-1}}, z_{1:t-1}, x_{k_t}, \mathbf{x})$$



EditNTS: Results



Benefits #1: Fact preserving by KEEP
Benefits #2: Controlled text generation by edit cost

Iterative Editing (2020)

- Four edit types:
 1. Removal
 2. Extraction
 3. Reordering
 4. Substitution

Optimize scores:

$$f(s) = f_{\text{eslor}}(s)^\alpha \cdot f_{\text{fre}}(s)^\beta \cdot (1/f_{\text{len}}(s))^\gamma \cdot f_{\text{entity}}(s)^\delta \cdot f_{\text{cos}}(s) \quad (2)$$

In 2016 alone, American developers had spent 12 billion dollars on constructing theme parks, ***according to a Seattle based reporter.***

Deletion



In 2016 alone, American developers had spent 12 billion dollars on constructing theme parks.

Reordering



American developers had spent 12 billion dollars ***in 2016 alone*** on constructing theme parks.

Lexical
Simplification



American developers had spent 12 billion dollars in 2016 alone on ***building*** theme parks.

Iterative Editing: Score Function

Optimize score $f(s)$:

LM: **fluency** while
preserve rare words

The Flesch Reading
Ease (FRE): **simplicity**

Length: **shorter**
sentences are better

$$f(s) = f_{\text{eslor}}(s)^{\alpha} \cdot f_{\text{fre}}(s)^{\beta} \cdot (1/f_{\text{len}}(s))^{\gamma} \\ \cdot f_{\text{entity}}(s)^{\delta} \cdot f_{\text{cos}}(s)$$

Entity Score:
entity preserving

Cosine Similarity to the input
sentence: **content preserving**

Iterative Editing: Unsupervised Learning

Learning if edited sentence has higher score $f(s)$:

$$f(c)/f(s) > r_{\text{op}}$$

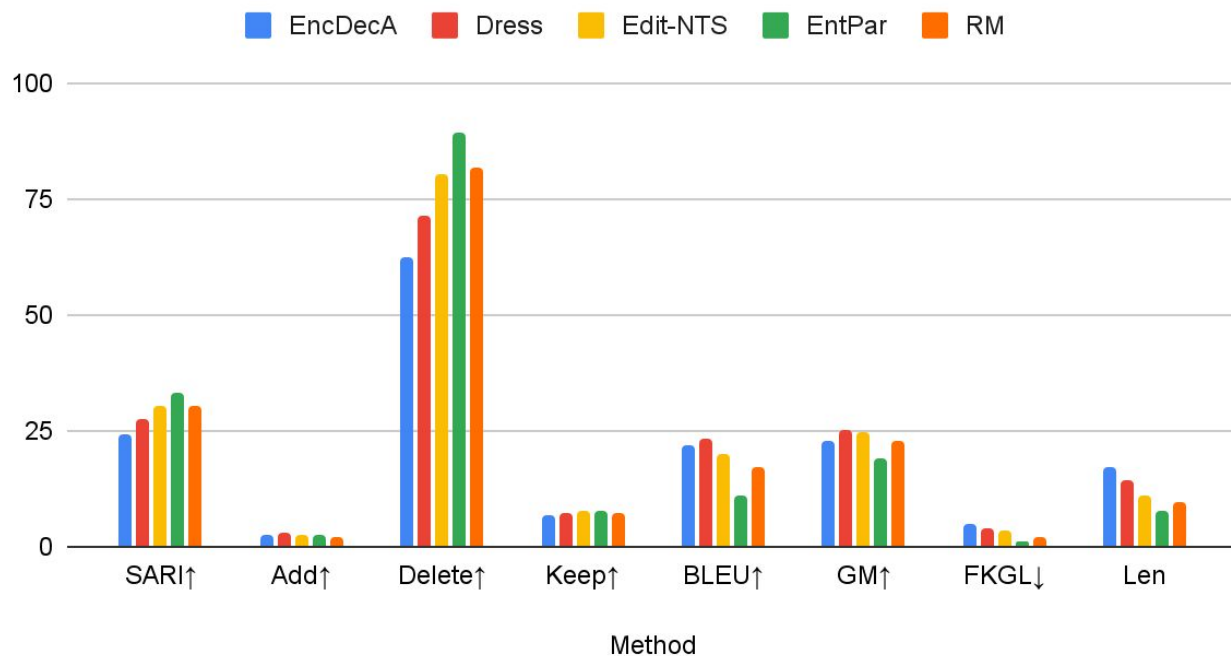
s: sentence given by the previous iteration

c: candidate generated by operator **op** from s

r: thresholds. different thresholds for each operation.

Results: Newsela

RM vs. Other models



Based on results in [Kumar et al., 2020](#)

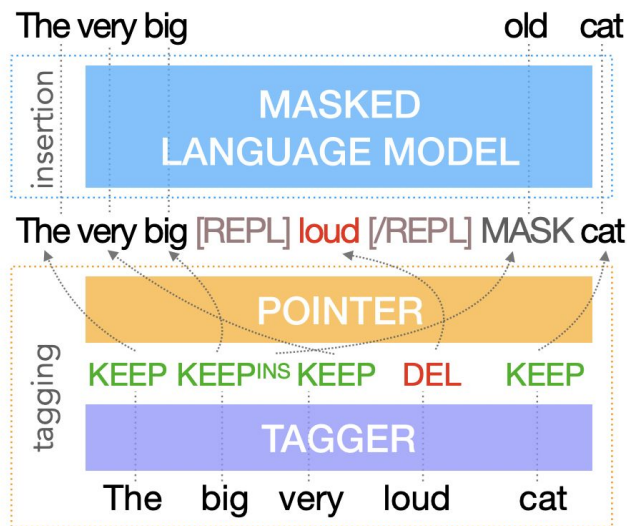
Felix: Flexible Text Editing (2020)

Two-step approach:

1. **Tagger:** Transformer with pointer (Vinyals et al., 2015)
2. **Insertion model:** text infilling with BERT

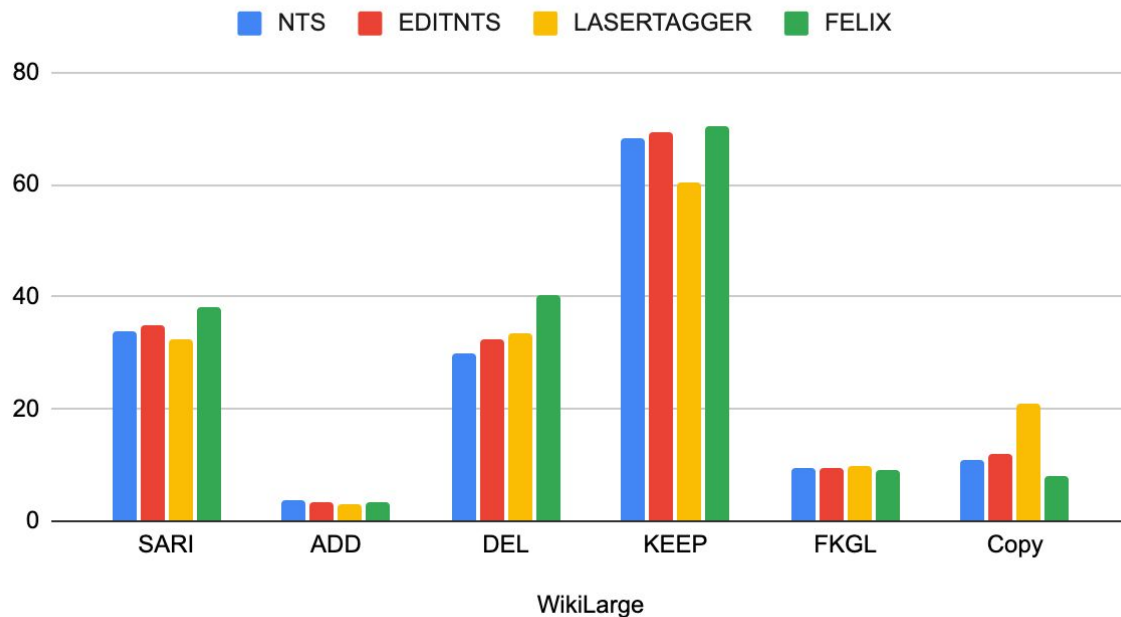
Features:

- Sample efficiency
- Fast inference time
- Flexible text editing



Felix: Results

NTS, EDITNTS, LASERTAGGER and FELIX



Based on results in [Mallinson et al., 2020](#)

Questions?