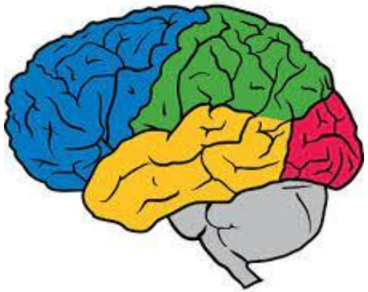
Editing + Diffusion for Text Generation

Machel Reid

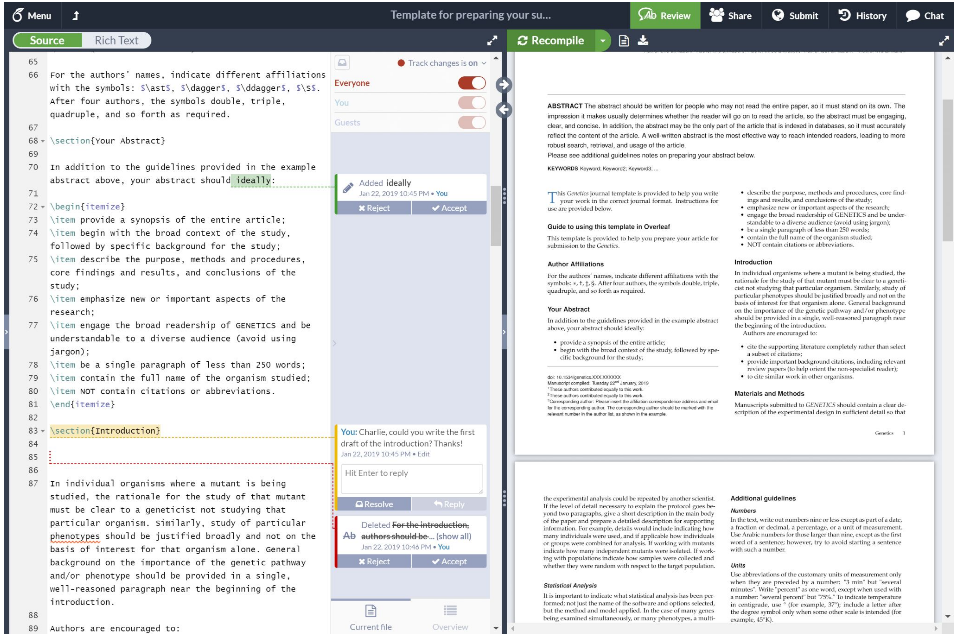
Who am I?

- Currently RS at Google Brain in London
- Working on multilinguality research + edit-models research

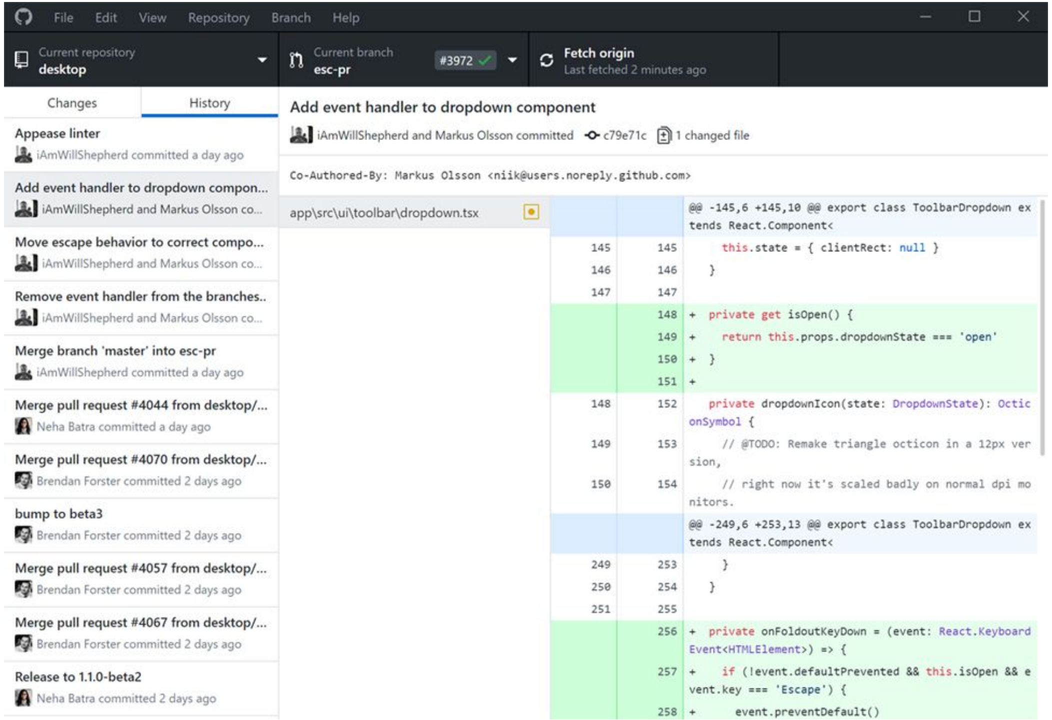


How do we create content?

Writing papers

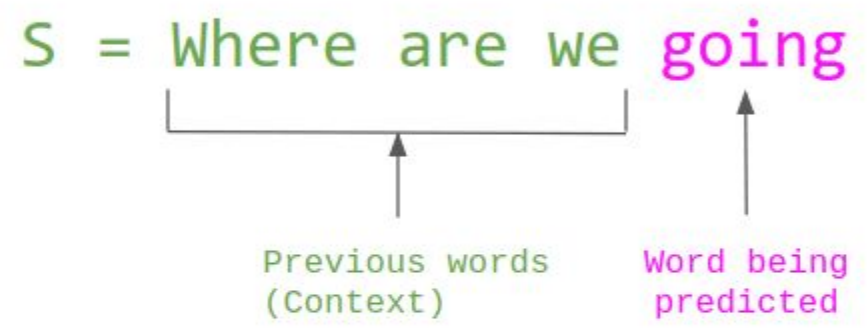


Writing code



Contrast this with current language models...

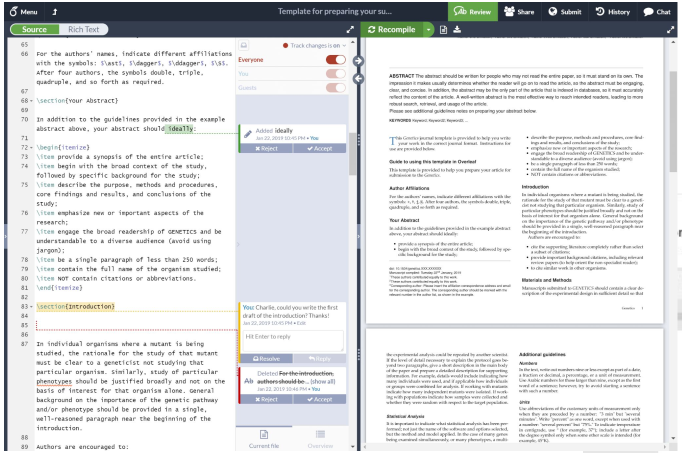
Autoregressive generation

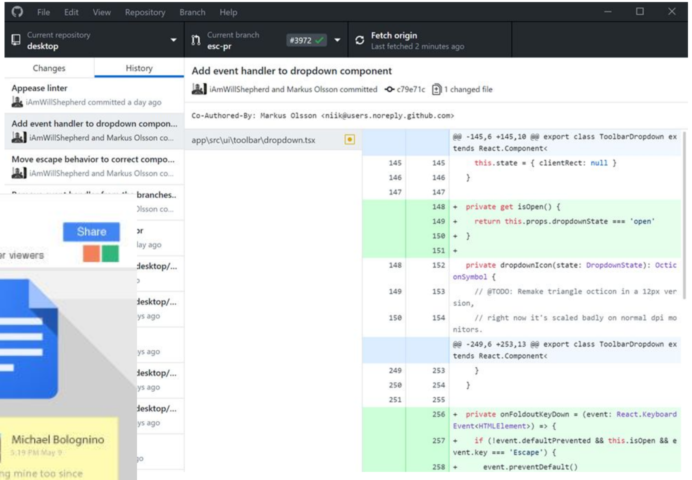


$$P(S) = P(\text{Where}) \times P(\text{are} \mid \text{Where}) \times P(\text{we} \mid \text{Where are}) \times P(\text{going} \mid \text{Where are we})$$

They have good performance and work for many things....

....but they’re very counter intuitive





With current autoregressive LMs

- We cannot revise text in an intuitive way.
- Iterative refinement is hard.
- This is an important task for that is simple for humans to perform but extremely difficult for models

Given this, I will go over some my work on editing

- Application-centric editing for text style transfer in “LEWIS: Levenshtein Editing for Unsupervised Text Style Transfer” ACL Findings 2021

- Learning to Model Editing Processes, Findings of EMNLP 2022

- **New preprint!** DiffusER: Discrete Diffusion via Edit-based Reconstruction

Applying editing for style transfer

LEWIS: Levenshtein Editing for Unsupervised Text Style Transfer

Joint work with Victor Zhong (UW)

Text Style Transfer: Motivation & Problem Definition

Negative to Positive:

I had a terrible time… → I had a great time…

Positive to Negative:

The worst ribs I’ve ever had! → Probably the best ribs ever!

Many current style transfer approaches require fully regenerating large portions of the original sentence

These sentences have a large text overlap, so editing could be a good idea

The text style transfer task

POSITIVE: I had a really ~~great~~ time at the theater, they ~~attended to all of my needs.~~

↓

NEGATIVE: I had a really terrible time at the theater, they ignored all my requests.

Benefits of editing

- Efficient

- Allows for content preservation + fluency preservation

- More precise control over the sequence transduction process

Levenshtein Editing

Predict Levenshtein operations {<ins>, <keep>, <repl>, <del>} and generate for <ins> and <repl> operations

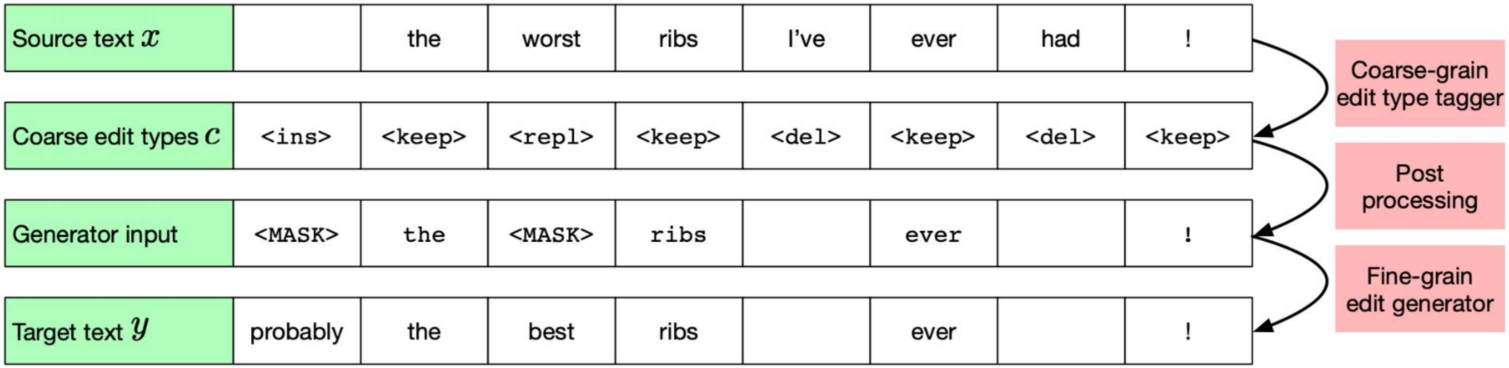


Figure 1: Coarse-to-fine Levenshtein editor. Given the source text, the two-step editor first generates coarse edit types via a tagger. A subsequent generator fills in insertions and replacements while taking into account the source text and the edit types.

Levenshtein Editing (cont.d)

Negative to Positive:

I had a terrible time… → I had a great time…

Positive to Negative:

The worst ribs ~~I've~~ ever ~~had~~! → Probably the best ribs ever!

LEWIS

2 Steps:

1. Synthesizing pseudo-parallel data

2. Learn a Levenshtein editing model with a coarse-grained editor and fine-grained generator to modify style of text

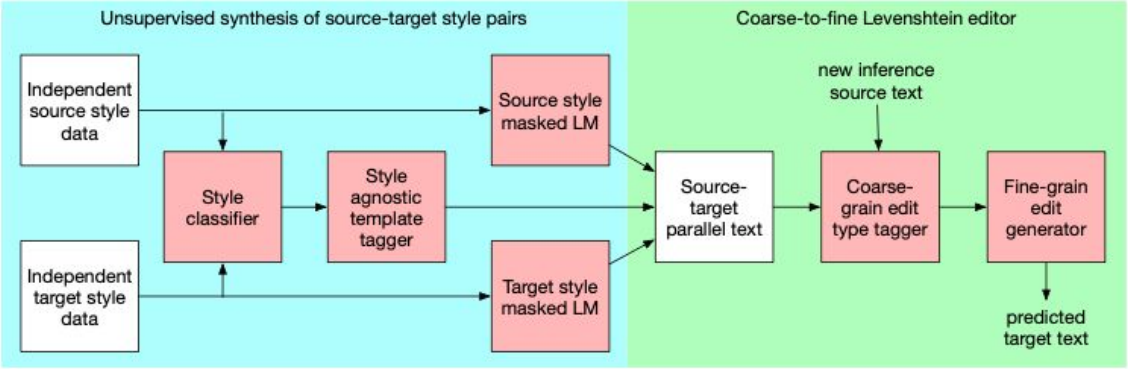


Figure 2: LEWIS consists of two components. Given source-target style text pairs, a coarse-to-fine Levenshtein editor (yellow) first identifies coarse-grain Levenshtein edit types to perform for each token in the source text (e.g. insert, replace, delete), then fills in the final edits with a fine-grain generator to produce the target text. In most applications, supervised source-target style text pairs rarely exist. To resolve this lack of annotated data, we perform unsupervised synthesis of source-target style pairs (blue) by first learning to produce style-agnostic templates given arbitrary style text. Next, we fill in slots in the template by sampling from style-specific masked language-models. In this figure, source and intermediate data are shown in white while model components are shown in red.

Synthetic data generation

- Use classifier attention to replace style-specific text with the SLOT token
- Fill that text with both style-specific LMs to create pseudo-parallel data

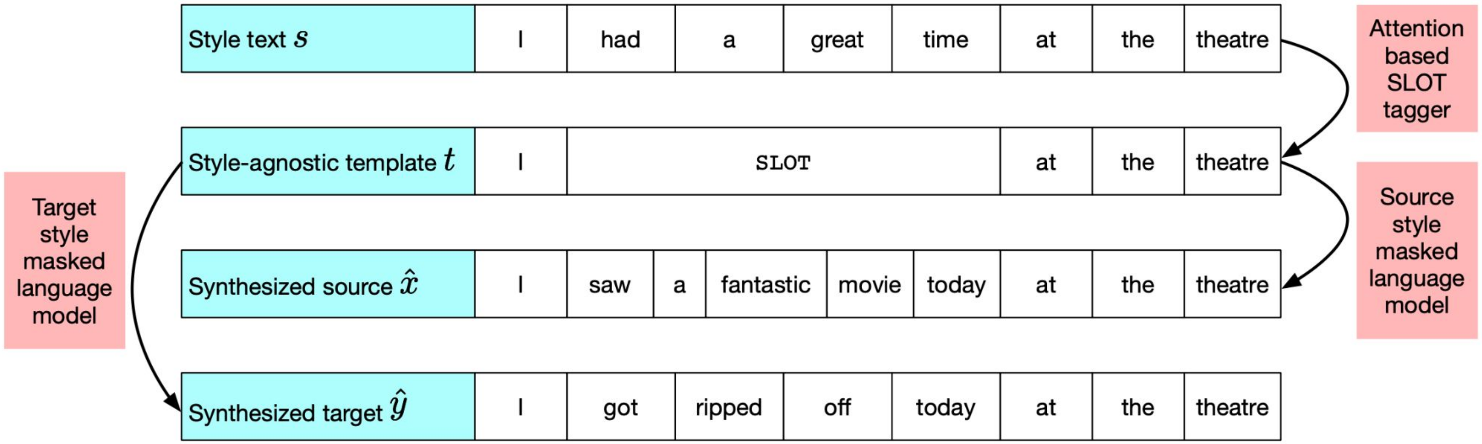


Figure 3: Unsupervised synthesis of source-target style pairs. We first train an attentive style classifier, whose attention weights we use to identify style-specific content. Next, we remove style-specific content with slots to form a style-agnostic template. This template is finally filled using style-specific masked language-models for each style to synthesize parallel style text pairs.

Results

- LEWIS improves over previous work on human and automatic evaluation!
- With our synthetic parallel data, editing based methods work much better...

Dataset	Model	Fluency	CP	Style
YELP	TG	3.84±1.01	3.63±0.93	3.67±1.02
	LEWIS	**3.94**±0.99	**3.76**±0.88	**3.72**±0.98
AMAZON	TG	3.60±1.01	3.48±0.93	3.37±1.02
	LEWIS	**3.65**±0.88	**3.50**±0.88	3.37±0.90
POLITE	TG	3.83±0.84	3.76±0.90	3.48±1.04
	LEWIS	**3.93**±0.78	**3.87**±0.83	**3.63**±0.98

Table 7: Human evaluation results comparing LEWIS and Tag and Generate (TG)

Model	Acc	SBLEU	BLEU	SBERT	BERT
Baselines					
Input Copy	1.5	100.0	24.8	100.0	53.74
Reference	81.6	25.3	100.0	53.7	100.0
Generation methods					
Delete and Retrieve (Li et al., 2018)	88.6	36.8	12.2	48.5	33.3
Tag and Generate (Madaan et al., 2020)	86.2	47.1	19.8	57.9	37.2
DeepLatentSeq (He et al., 2020b)	83.8	48.4	18.7	57.9	36.0
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.

However, what if we can make editing not application-based, but more general?

Learning to Model Editing Processes

Joint with with Graham Neubig (CMU)

Carnegie Mellon University
Language Technologies Institute

Wikipedia Edits

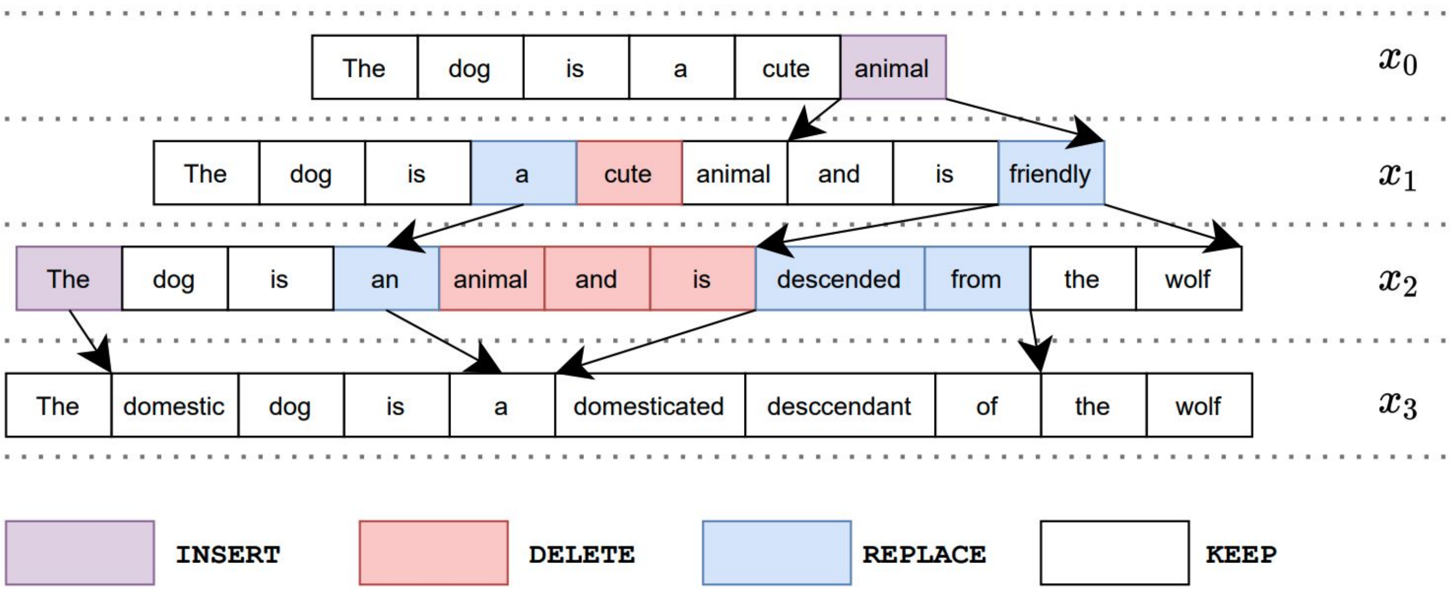


Figure 1: An example of a natural editing process based on the description of "Dog" on Wikipedia. The legend below denotes the edit operations for each step of this process.

Why do we want to model it?

- Humans generate content iteratively (not in one pass -> GPT-style)

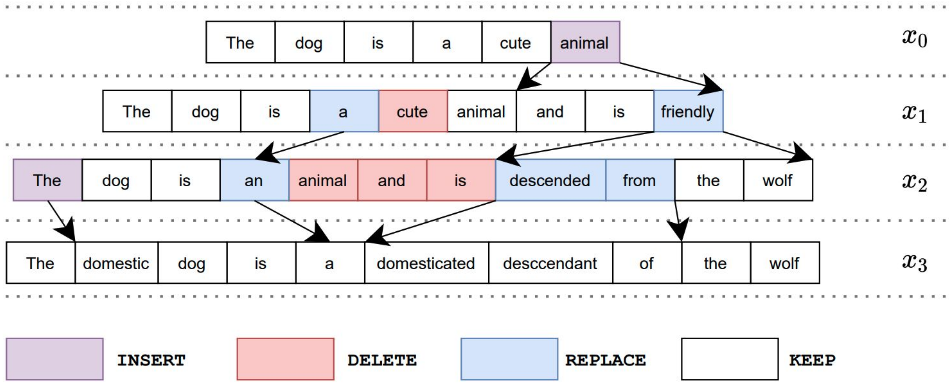


Figure 1: An example of a natural editing process based on the description of "Dog" on Wikipedia. The legend below denotes the edit operations for each step of this process.

Why do we want to model it?

- Humans generate content iteratively (not in one pass -> GPT-style)

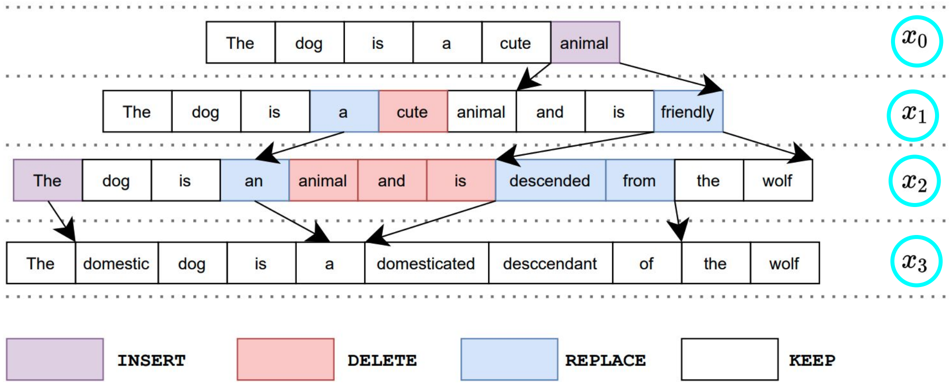


Figure 1: An example of a natural editing process based on the description of "Dog" on Wikipedia. The legend below denotes the edit operations for each step of this process.

Why do we want to model it?

- Humans generate content iteratively (not in one pass -> GPT-style)

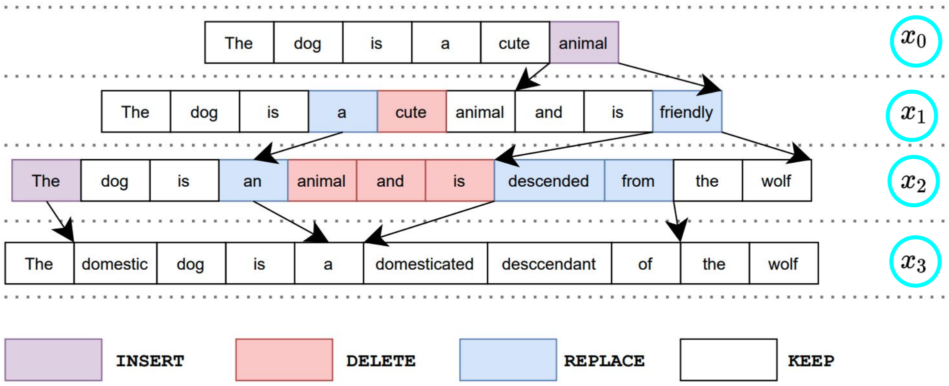


Figure 1: An example of a natural editing process based on the description of “Dog” on Wikipedia. The legend below denotes the edit operations for each step of this process.

- Are there patterns when editing processes?

Problem Definition

- We want to model the likelihood of the current document by way of an entire sequence of document edits

$$p(\boldsymbol{x}_N) = \sum_{\{\tilde{X}=\tilde{\boldsymbol{x}}_1^N \mid \tilde{\boldsymbol{x}}_N=\boldsymbol{x}_N\}} p(\tilde{X}).$$

Problem Definition: n-order editing

- We want to model the likelihood of the current document by way of an entire sequence of document edits, but we can simplify this to a Markov process (which is single step editing; Reid and Zhong., 2021)

$$p(\boldsymbol{x}_i|\boldsymbol{x}_0^{i-1})$$

- However, when modeling edit **processes** (the aim of this work), we look to include the context of previous revisions (controlled by *n*).

$$p(\boldsymbol{x}_i|\boldsymbol{x}_{i-n}^{i-1})$$

Problem Definition: Edit Operations

- Practically, we use edit operations (e_i) (INSERT, DELETE, KEEP, REPLACE) to edit this and make this process more efficient:

$$p(\mathbf{x}_i|\mathbf{x}_{i-n}^{i-1}) \approx p(\mathbf{x}_i, \mathbf{e}_i|\mathbf{x}_{i-n}^{i-1})$$
$$= p(\mathbf{x}_i|\mathbf{e}_i, \mathbf{x}_{i-n}^{i-1})p(\mathbf{e}_i|\mathbf{x}_{i-n}^{i-1}).$$

Problem Definition: Edit Log Likelihood

- We can then use this formulation to define edit log-likelihood (which we use to train our model)

$$\mathcal{L}_{\mathbf{xe}} \coloneqq \log P(\mathbf{x}_1^N) = \sum_{i=1}^{N} \log p(\mathbf{x}_i | \mathbf{e}_i, \mathbf{x}_{i-n}^{i-1}) + \log p(\mathbf{e}_i | \mathbf{x}_{i-n}^{i-1}).$$

Problem Definition: Decomposed Log Likelihood

- We can also decompose edit log likelihood into the operation prediction:

$$\mathcal{L}_{\mathbf{e}} := \sum_{i=1}^{N} \log p(\mathbf{e}_i | \mathbf{x}_{i-n}^{i-1})$$

- And operation-conditioned generation

$$\mathcal{L}_{\mathbf{x}|\mathbf{e}} := \sum_{i=1}^{N} \log p(\mathbf{x}_i | \mathbf{e}_i, \mathbf{x}_{i-n}^{i-1})$$

Model: EditPro

Model: EditPro

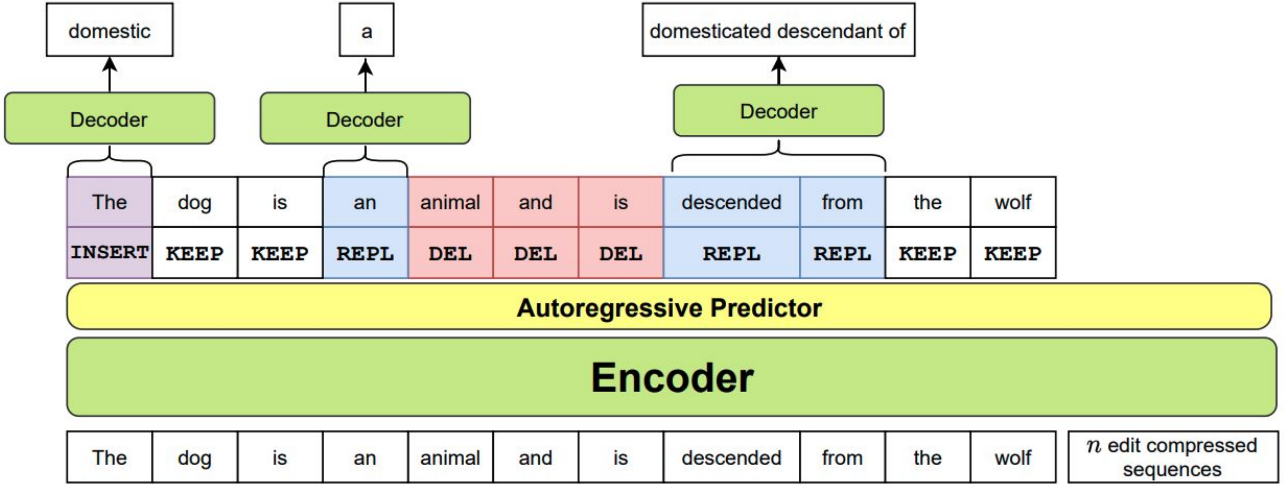


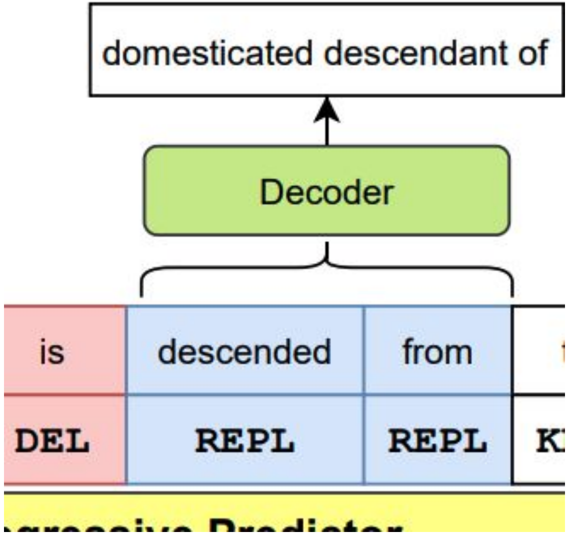
Figure 2: EDITPRO given the examples of modeling $p(\boldsymbol{x}_3|\boldsymbol{x}_2)$ from Figure 1. We feed the input tokens into an encoder with an autoregressive tag predictor, and then use the predicted edit operations to condition the generation of `REPLACE` and `INSERT` spans.

Components

- Edit Encoder
- Edit Operation Prediction
- Generating Replacements and Insertions
- Encoding Edit History

Generating Replacements and Insertions

- E.g. we take a **mean pool** of replaced tokens and **sum them with a** REPLACE **embedding** and use that to **initialize** a decoder for that span



Edit-compressed history

- We use previous edit operations to to compress previous edit history into their separate spans of edits
- (Hard to explain here, so please refer to the paper!)

Data

WikiRevisions & CodeRevisions

- We propose the datasets with full document-level edit history for both natural language (WikiRevisions from Wikipedia) and code (CodeRevisions from Github)

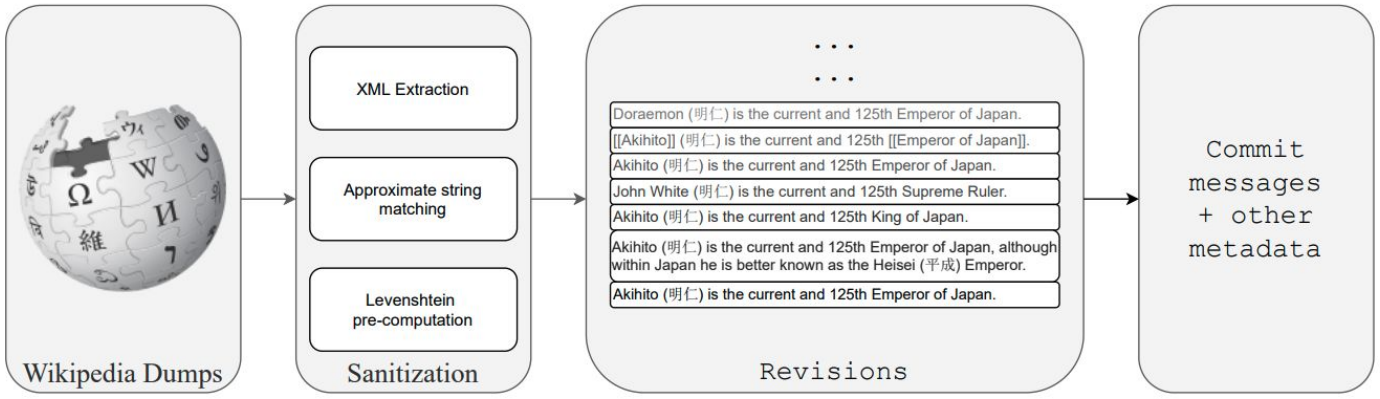


Figure 3: An overview of the WIKIREVISIONS data generation process for collecting clean multi-step revision data.

Evaluation Metrics

Evaluation Metrics

- **Edit Perplexity (ePPL)**, exponent of the NLL for both edits and generated outputs, normalized by length of both outputs
  $$\exp\left(\frac{\mathcal{L}_{\mathbf{xe}}}{|\mathbf{x}|+|\mathbf{e}|}\right)$$

Evaluation Metrics

- **Edit Perplexity (ePPL)**, exponent of the NLL for both edits and generated outputs, normalized by length of both outputs

  $$\exp\left(\frac{\mathcal{L}_{\mathbf{xe}}}{|\mathbf{x}|+|\mathbf{e}|}\right)$$

- **Operation Perplexity (oPPL),** exponentiated NLL of operation prediction

  $$\exp\left(\frac{\mathcal{L}_{\mathbf{e}}}{|\mathbf{e}|}\right)$$

Evaluation Metrics

- **Edit Perplexity (ePPL)**, exponent of the NLL for both edits and generated outputs, normalized by length of both outputs

  $$\exp\left(\frac{\mathcal{L}_{\mathbf{xe}}}{|\mathbf{x}|+|\mathbf{e}|}\right)$$

- **Operation Perplexity (oPPL),** exponentiated NLL of operation prediction

  $$\exp\left(\frac{\mathcal{L}_{\mathbf{e}}}{|\mathbf{e}|}\right)$$

- **Generation Perplexity (gPPL),** exponentiated NLL of generating replaced or inserted spans (when compared with ground truth edit seq)

  $$\exp\left(\frac{\mathcal{L}_{\mathbf{x}|\mathbf{e}}}{|\mathbf{x}|}\right)$$

Experiments & Results

Tasks

- Edit Modeling
- Edit Classification
- Conditional Editing
- Edit-conditioned Generation

Edit Modeling

Extra order edit modeling helps!

DATASET	Model	ePPL	gPPL	oPPL			
				DEL	KEEP	REPL	INS
WIKIREVISIONS	LEWIS	65.94	48.85	24.29	**1.09**	19.49	507.76
	EDITPRO (1-order)	57.32	42.43	25.53	**1.09**	18.36	1826.21
	EDITPRO (2-order)	53.91	39.87	20.70	1.13	15.49	376.15
	EDITPRO (3-order)	**50.84**	**37.66**	**19.30**	1.13	**14.88**	**252.14**
CODEREVISIONS	EDITPRO (1-order)	34.22	28.02	125.21	**1.05**	10.38	544.57
	EDITPRO (2-order)	30.85	26.26	84.77	**1.05**	9.30	**304.90**
	EDITPRO (3-order)	**29.47**	**25.37**	**75.19**	1.06	**8.16**	441.42

Table 2: Results on Edit Modeling

Edit Modeling

Extra order edit modeling helps! **Knowing where** text came from helps predict future iterations.

Dataset	Model	ePPL	gPPL	oPPL			
				DEL	KEEP	REPL	INS
WikiRevisions	LEWIS	65.94	48.85	24.29	**1.09**	19.49	507.76
	EditPro (1-order)	57.32	42.43	25.53	**1.09**	18.36	1826.21
	EditPro (2-order)	53.91	39.87	20.70	1.13	15.49	376.15
	EditPro (3-order)	**50.84**	**37.66**	**19.30**	1.13	**14.88**	**252.14**
CodeRevisions	EditPro (1-order)	34.22	28.02	125.21	**1.05**	10.38	544.57
	EditPro (2-order)	30.85	26.26	84.77	**1.05**	9.30	**304.90**
	EditPro (3-order)	**29.47**	**25.37**	**75.19**	1.06	**8.16**	441.42

Table 2: Results on Edit Modeling

Downstream Tasks

Same findings hold, even for discriminative edit-based tasks

Dataset	Model	BLEU	F1	ePPL (Δ)
WikiRevisions	EditPro (1-order)	10.7	57.8	54.72 (-2.60)
	EditPro (2-order)	11.3	**61.3**	51.83 (-2.08)
	EditPro (3-order)	**11.6**	61.2	**49.91** (-0.93)
CodeRevisions	EditPro (1-order)	13.8	—	33.65 (-0.57)
	EditPro (2-order)	14.3	—	30.13 (-0.72)
	EditPro (3-order)	**14.5**	—	**29.08** (-0.39)

Table 3: Results on Edit Generation (BLEU), Edit Classification (measured with micro-F1), and Conditional Edit Generation (measured Edit Perplexity = ePPL). Note that the Δ symbol refers to the change between the model's non-message conditioned version in Table 2.

Example from sampling from a edit model

Initial Sentence (1-order)	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere.
x_2	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere. Spain is a member of the European Union.
x_3	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere. France is a member of the European Union.
x_4	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere. France is is a lieing country in the world. It is a bunch of crap.
x_5	Europe is a continent located entirely in the ~~Northern~~ Hemisphere and mostly in the Eastern Hemisphere. ~~France is a lieing country in the world. It is a bunch of crap.~~ There is a type of debate of a group of people who are not considered to be a part of the United Nations.
Initial Sentence (2-order)	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere.
x_2	Europe is a continent located entirely in the Northern Hemisphere and mostly in the Eastern Hemisphere. The Western South Eastman Islands are also located in Europe.
x_3	Europe is .k.ka.j.jf.go.skxklse
x_4	Europe is ~~.k.ka.j.jf.go.skxklse~~ a continent in the Northern Hemisphere. The Islands are also in Europe and they are great.

Table 4: Example generation when sampling with an edit model. We notice that the 2nd order model is able perform a revert operation given the context fed through the edit-compressed sequence about the previous revision, whereas the 1-order model although deleting its generated spam, generates something relatively unrelated. However we note that this reversion is not exact (likely due to the information loss during edit compression). This corresponds with our observations in our qualitative study (where likelihood of reverted edits is increased in the 2+ order models).

Not super fluent, but is likely to be an artefact of:

- Scale (small, undertrained model)

- Data: Wikipedia is a comparative cacophony to other forms of creation

But can we make this notion more general?

Introducing text + edit-based diffusion models!

go/diffuser

DiffusER: Diffusion via Edit-based Reconstruction

Joint work with Vincent Hellendoorn, Graham Neubig @CMU

S3D
Software and Societal
Systems Department

Language
Technologies
Institute

Setup

Most text generation models

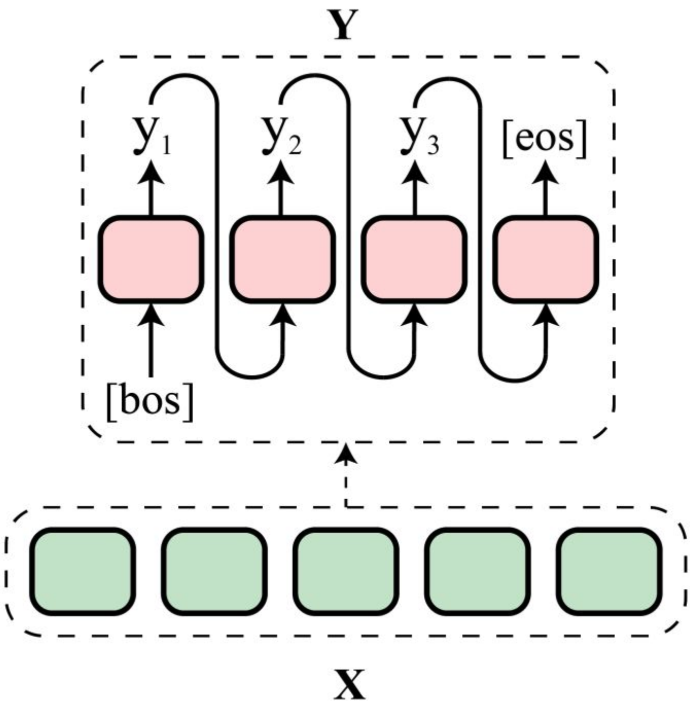
Left-to-right

Pros:

- Simple and effective setup

Cons:

- Hard to refine
- Not much flexibility when generating



(a)

Non-autoregressive models

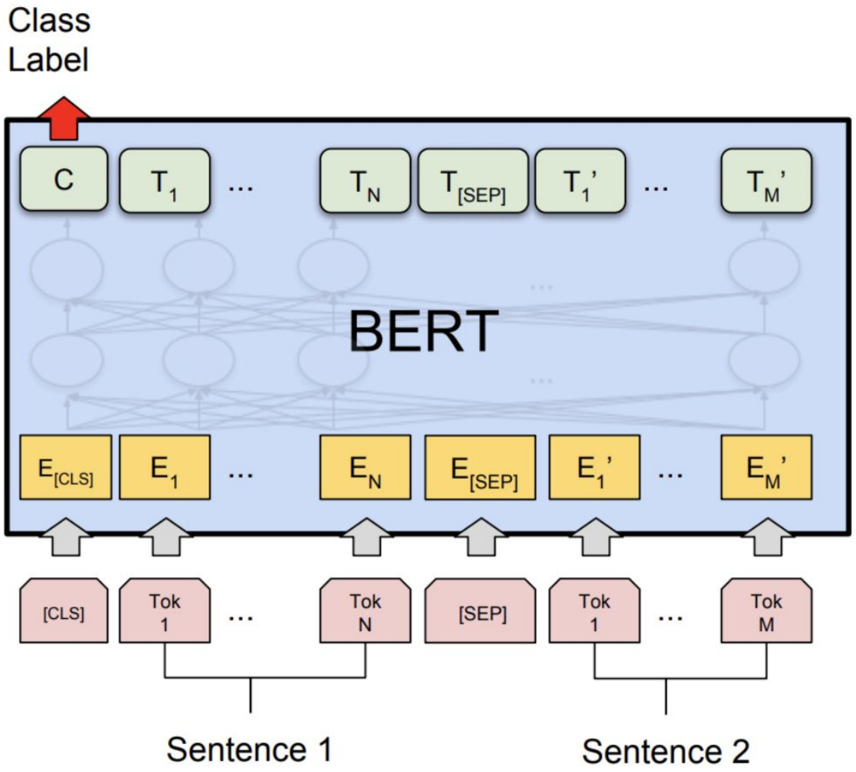
E.g. CMLM/MLMs

Pros:

- Simple
- Effective
- Fast

Cons:

- Arguably even less flexibility than AR models



Text diffusion models

Diffusion models have two components

1) Corruption (or forward process)

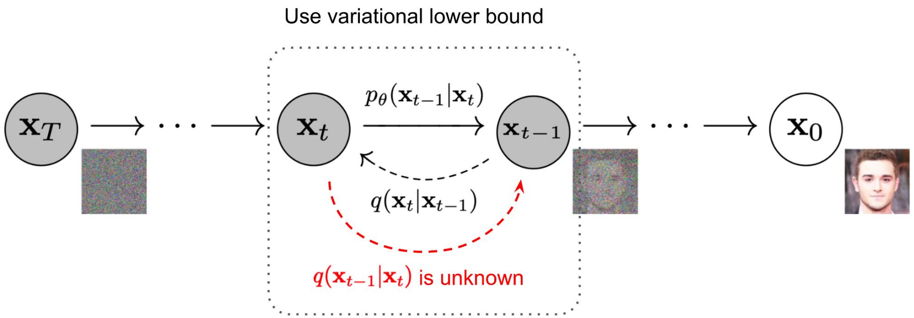
   E.g. in images

   Full image ⟶ noise

2) Denoising (or backward process)

   Generative Modeling

   Noise ⟶ Full image



[2006.11239] Denoising Diffusion Probabilistic Models

Issues with diffusion models for text

- Unlike images, score-based generative modeling is not straightforward as there is no clear method on how to formulate diffusion for categorical distributions

- The corruption process for text-based models and hence the denoising process is also not straightforward

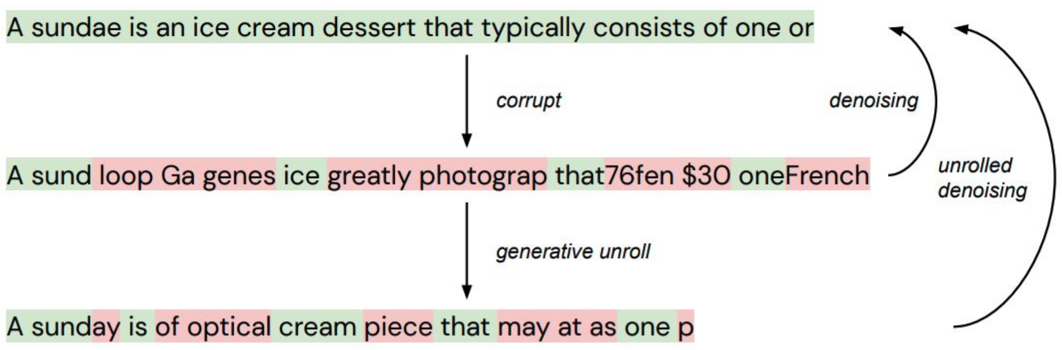
Previous work

Structured Denoising Diffusion Models in Discrete State-Spaces (i.e. using the BERT/CMLM objective in multiple steps)

```
t = 128  [MASK] [MASK] [MASK] [MASK] [MASK] [MASK]...
t = 25  In response [MASK] the demands , [MASK] [MASK]y Workers
union said [MASK] backflow fund [MASK]s would face further
investigation and a fine.
t = 0  In response to the demands , the Community Workers union
said the backflow fund managers would face further investigation
and a fine .
```

Previous work cont.d

Step-unrolled Denoising Autoencoders for Text Generation; SUNDAE (instead of using masks, replace iteratively using random text)



SUNDAE cont.d

- Pretty good MT performance
- However, cannot make use of flexible edit-operators; paradigm relatively inflexible compared to the ideal for edits

Model	Steps (T)	Raw BLEU EN→DE	Raw BLEU DE→EN
AR Models			
Transformer Base (65M) (Vaswani et al., 2017) (n=4)	-	27.3	31.78*
Non-AR Models			
NAT (Gu et al., 2017) (n=100)	1	-	-
LVM-DAE (Lee et al., 2018)	-	-	-
NAT-REG (Wang et al., 2019) (n=9)	1	-	-
LV-NAR (Shu et al., 2020) (n=50)	1	11.8	-
NART w/ hints (Li et al., 2019)(n=9)	1	-	-
FlowSeq (Ma et al., 2019) (n=30)	1	23.64	28.29
ReorderNAT (Ran et al., 2019)	1	-	-
NART (Sun et al., 2019) (n=19)	1	-	-
CMLM (Ghazvininejad et al., 2019) + Mask-Predict (n=5)	4	22.25	-
CMLM (Ghazvininejad et al., 2019) + Mask-Predict (n=5)	10	24.61	-
DisCo (Kasai et al., 2020) + Mask-Predict (n=5)	4	-	-
DisCo (Kasai et al., 2020) + Mask-Predict (n=5)	10	-	-
DisCo (Kasai et al., 2020) + Easy-First (n=5)	4-5†	24.8	-
NARLVM (Lee et al., 2020) (n=25)	4	-	-
JM-NAT (Guo et al., 2020) (n=3)	4	-	-
JM-NAT (Guo et al., 2020) (n=3)	10	-	-
SMART (Ghazvininejad et al., 2020) (n=5)	4	-	-
SMART (Ghazvininejad et al., 2020) (n=5)	10	-	-
Imputer (Saharia et al., 2020) (n=1)	4	24.7	-
Imputer (Saharia et al., 2020) (n=1)	8	25.2	-
SUNDAE (ours 63M)			
Deterministic (n=16)	4	25.01	29.53
Deterministic (n=16)	8	25.53	30.01
Deterministic (n=16)	10	25.54	30.11
Stochastic (n=16)	4	23.05	28.13
Stochastic (n=16)	8	26.08	30.48
Stochastic (n=16)	10	**26.25**	**30.80**
Stochastic (n=16)	16	26.24	30.76

Ours

Issues with previous work

The main one there are too many restrictions placed on what is diffusion/what consists of diffusion etc…

1) E.g. for the MLM diffusion models, the model doesn’t actually learn to correct incorrect text -> just learns to fill masks

2) For the SUNDAE model, it overcomes the first limitation, however it is quite restrictive in terms of the types of edits it can perform (essentially only replacement).

We aim to fix this by:

- We use the SUNDAE style of using randomly sampled text rather than <MASK>s (tackling problem **1**)

- We also include autoregressive generation in this process (though arguably there could be a purely non autoregressive formulation of this) **(this allows us to have compatibility with AR models)**

- We use Levenshtein edit operations (i.e. KEEP, DELETE, REPLACE, INSERT) to make the editing both more controllable and flexible.

Our corruption process is flexible + DELETE/INSERT are new

1) Extremely flexible, uses edit operations

Instead of simply replacing tokens we can perform the following 4 operations:

- KEEP (i.e. do nothing)
- REPLACE (i.e. replace a span of words with another random span of words — not length constraint)
- **DELETE** (i.e. delete a set of tokens, this means that they would have to be inserted in the next timestep)
- **INSERT** (i.e. randomly insert a set of tokens, this means that they would have to be deleted)

Edit-based Generation || Our corruption process

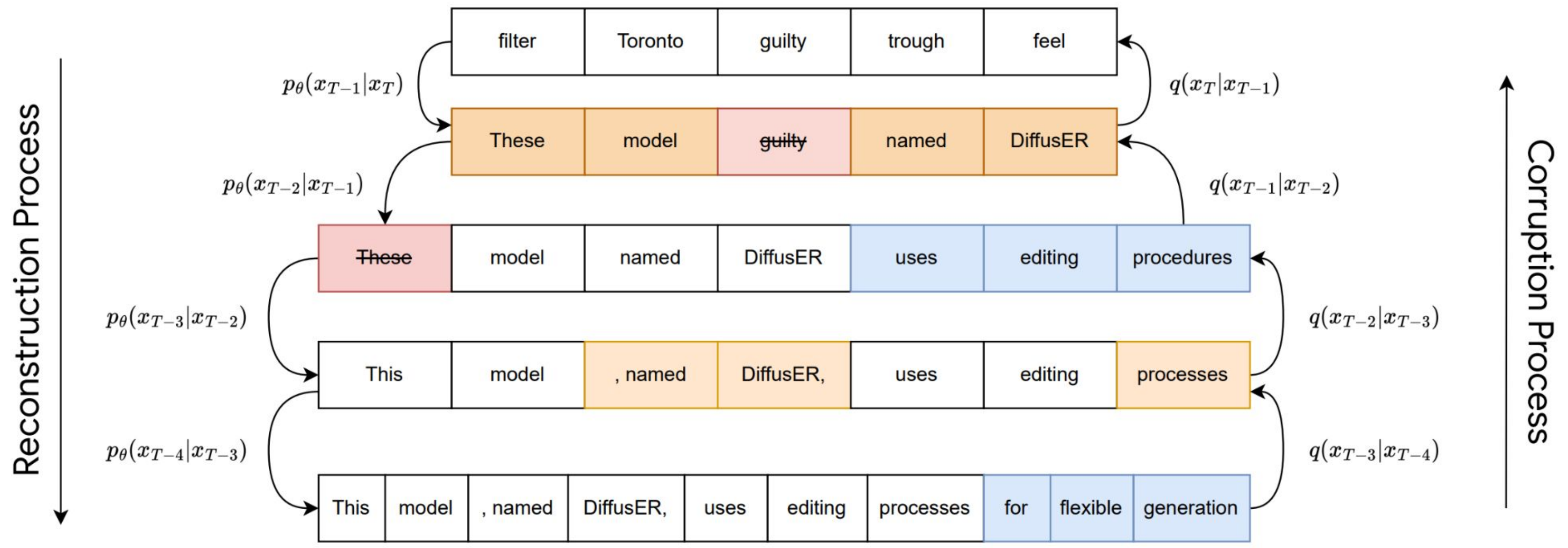


Figure 1: DIFFUSER's text generation process. Orange represents replacements, blue represents insertions, red represents deletions, and white represents keep operations. This process largely imitates a natural editing process (Reid & Neubig, 2022).

Our denoising process

Two step process:

1) **Tagger** -> we tag a corrupted sentence with the appropriate tags (i.e. replace, insert, delete, etc)

Our denoising process

Two step process:

1) **Tagger** -> we tag a corrupted sentence with the appropriate tags (i.e. replace, insert, delete, etc) -> similar to LEWIS

2) **Generator**: after tagging and summing tag embeddings and word embeddings, we generate using an autoregressive generator similar to CM3

Our denoising process

Two step process:

1) **Tagger** -> we tag a corrupted sentence with the appropriate tags (i.e. replace, insert, delete, etc) -> similar to LEWIS

2) **Generator**: after tagging and summing tag embeddings and word embeddings, we generate using an autoregressive generator similar to CM3

I have a <repl:0> dog </repl:0> his name is <insert:0> </s> <repl:0> great </s> <insert:0> Jonathan </s>

Our denoising process

Two step process:

1) **Tagger** -> we tag a corrupted sentence with the appropriate tags (i.e. replace, insert, delete, etc) -> similar to LEWIS

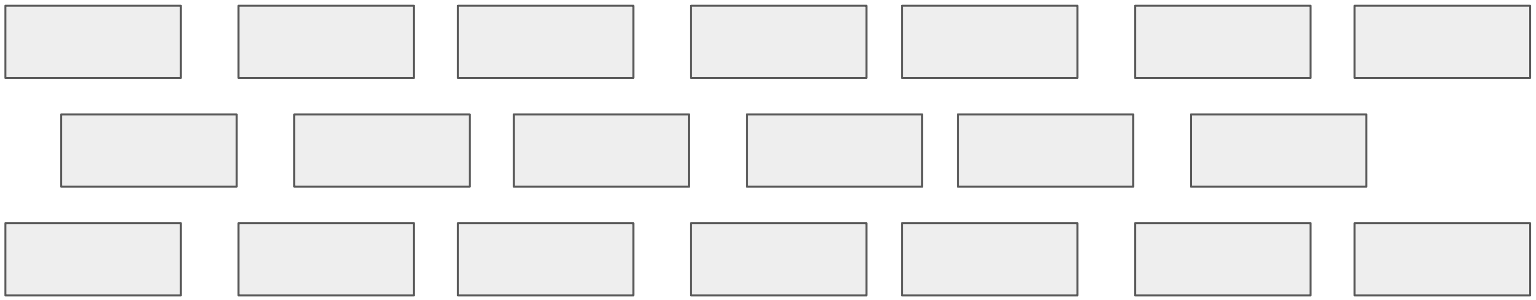
2) **Generator**: after tagging and summing tag embeddings and word embeddings, we generate using an autoregressive generator similar to CM3

I have a <repl:0> dog </repl:0> his name is <insert:0> </s> <repl:0> great dog </s> <insert:0> Jonathan </s>

I have a great dog his name is Jonathan

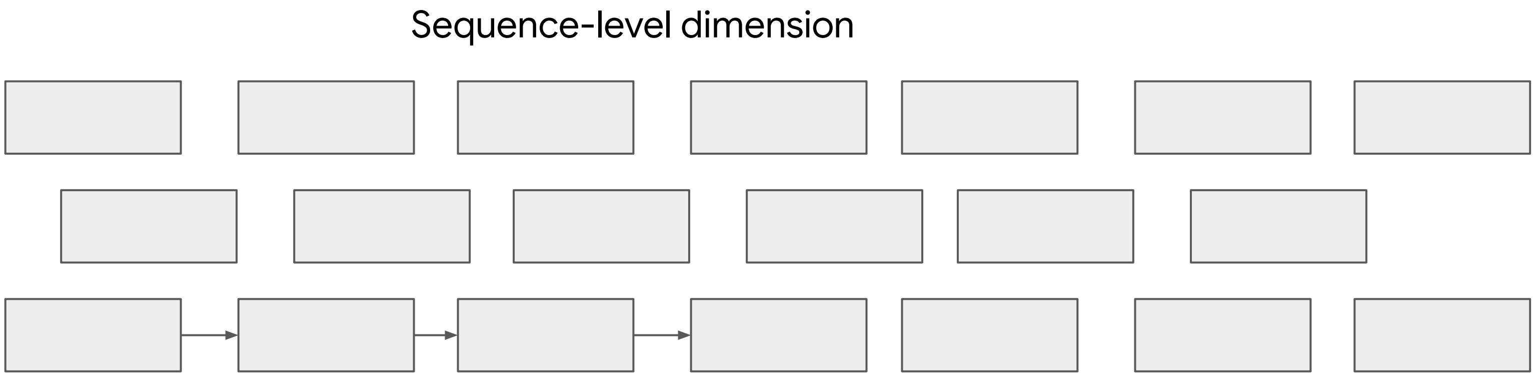
Generation process

- We perform 2d beam search, searching over 2 dimensions
  - Sequence level dimension (as standard)
  - **Revision level dimension**
- We can keep refining indefinitely, however we find 8-12 refinements work well.



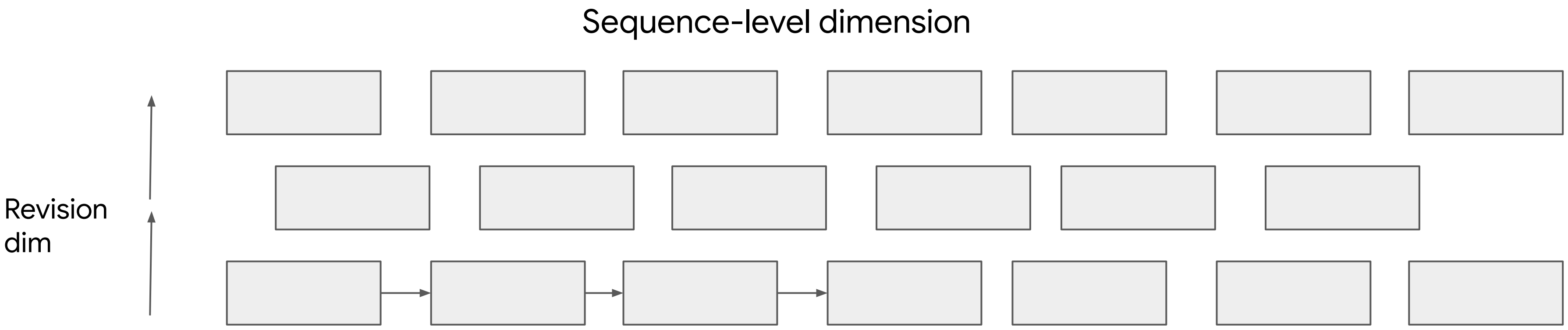
Generation process

- We perform 2d beam search, searching over 2 dimensions
  - Sequence level dimension (as standard)
  - **Revision level dimension**
- We can keep refining indefinitely, however we find 8-12 refinements work well.



Generation process

- We perform 2d beam search, searching over 2 dimensions
  - Sequence level dimension (as standard)
  - **Revision level dimension**
- We can keep refining indefinitely, however we find 8-12 refinements work well.



Decoder Initialization Techniques

Instead of initializing with continuous representations, we can actually initialize our decoder with discrete sequences

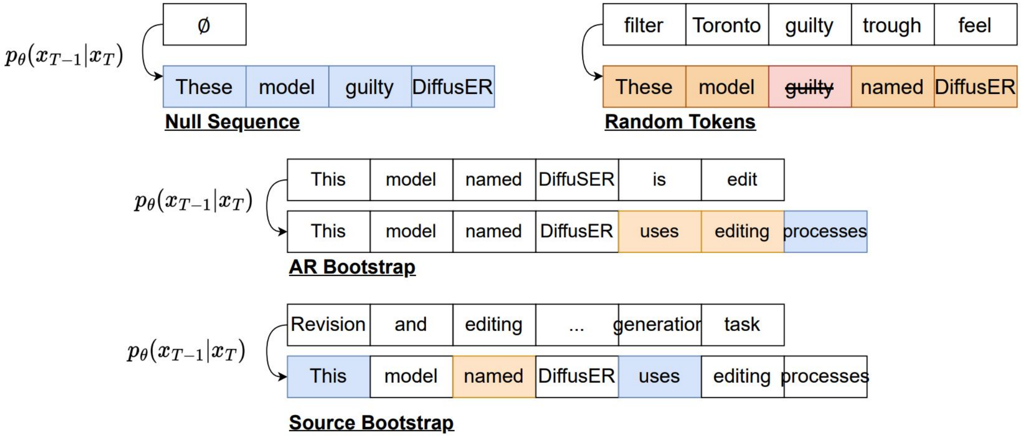


Figure 2: Figure illustrating bootstrapping methods for decoding.

Quantitative Results

Model	En-De (MT)	CNN-DM (Summ)
AR Transformer (Vaswani et al., 2017)	27.3	36.8
SUNDAE (Savinov et al., 2022)	26.3	37.0
CMLM (Ghazvininejad et al., 2019)	24.6	—
Levenshtein Transformer[2] (Gu et al., 2019)	23.7	—
DisCo (Kasai et al., 2020a)	24.7	—
Imputer	25.2	—
DIFFUSER	27.2	37.8
DIFFUSER + AR bootstrap	**28.8**	38.4
DIFFUSER + source bootstrap	24.5	**38.9**

Table 2: Machine Translation (MT) and Summarization (Summ) results on WMT'14 En-De (gold) and CNN-DailyMail. Experiments on MT use BLEU while summarization uses ROUGE. DIFFUSER is compatible with a standard autoregressive model, while outperforming previous methods.

With no distilled data, performs almost as well as standard AR for the first time!

Model	Accuracy	BLEU
Masker (Malmi et al., 2020)	40.9	14.5
Tag and Generate (Madaan et al., 2020)	86.2	19.8
LEWIS (Reid & Zhong, 2021)	**93.1**	24.0
DIFFUSER	87.6	**25.2**

Table 3: Results on Yelp dataset for text style transfer. Without task-specific training techniques, DIFFUSER performs comparably to previous task-specific methods.

Without task-specific techniques, works well with style transfer

Example Generation

Source Document	(CNN)They're not gonna take it anymore. Really. Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." "It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob. Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 1	~~(CNN)They're not gonna take it anymore. Really.~~ Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." "It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob. Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 2	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." ~~"It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob.~~ Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 3	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." Portnoy replaces A.J. Pero, who died March 20. The band will ~~also~~ perform two shows in Pero's honor ~~: one at Las Vegas ' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video~~in Las Vegas and New Jersey.
Step 4	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." ~~Portnoy replaces~~ A.J. Pero, ~~who~~ died March 20. The band will perform two shows in Pero's honor in Las Vegas and New Jersey.
Generated Summary	Twisted Sister says that its 2016 tour will be its last. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." A.J. Pero, died March 20. The band will perform two shows in Pero's honor in Las Vegas and New Jersey.

Table 5: Example of our summarization DIFFUSER process on a test set example. Here we show that the majority of the summarization process is deletion coupled with minor edits. Despite this simplicity, we are able to improve over existing purely abstractive models.

Ablations + Insights

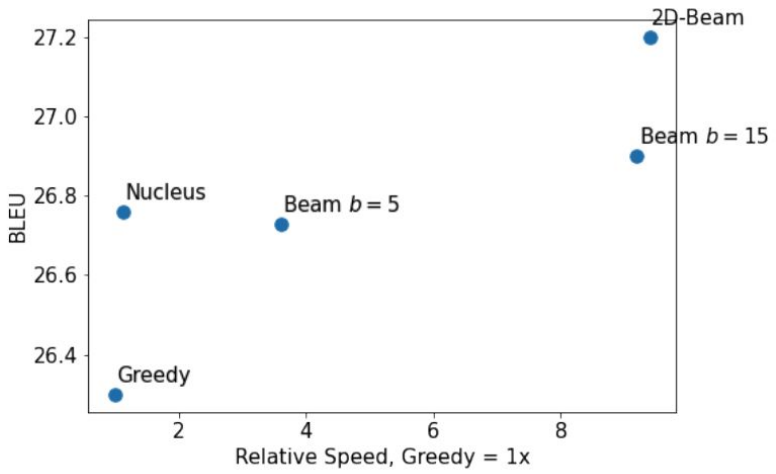


Figure 3: Relative time (seconds) comparison between decoding methods, measured on a single V100 GPU. There is a trade-off between inference cost and performance. Faster well-performing decoding algorithms for diffusion models are an area for further work.

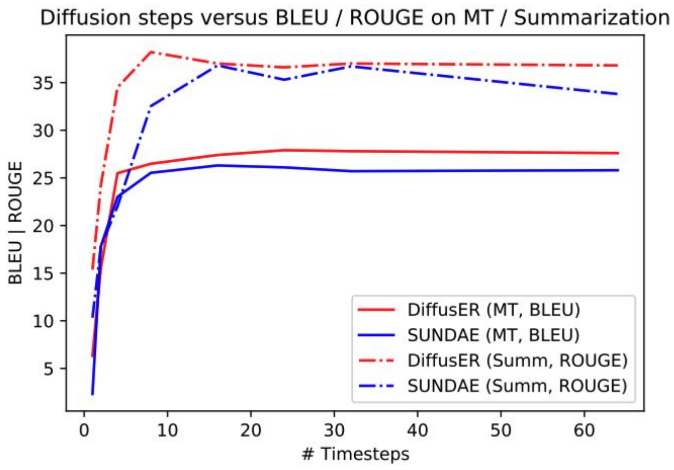


Figure 4: Number of steps versus BLEU/ROUGE on WMT'14 En-De and Summarization for both SUNDAE and DIFFUSER. We observe fast initial progression with performance, leveling off as steps increase.

Takeaways

Takeaways

- Text editing has a lot of promise and has been shown to be performant in certain situations but there is still a ways to go (LEWIS)

- A large drawback has been lack of data for style edits but with diffusion-inspired models we may be able to get there…

- But we have shown that we can incorporate this editing ability without compromising performance significantly and sometimes improving it!

Future Ideas

Improving Data Quality of Edits

- Issues with Wikipedia include:
  - Conflicting views
  - Spam
  - Bots
- Could we get golden Overleaf/Google Docs data?

Classifier-guided DiffusER

- One large issue: the corruptions in DiffusER are random and are largely conditioned on the task objective (i.e. machine translation, summarization)
- But can we use classifiers to induce different types of edits/paths?

Using DiffusER style models for data augmentation

- Given a seed sequence you can sample iteratively to form different perturbations of the same sequence.

Future ideas

- Have a large scale pre-trained self-editing model
- Ideally everyone should be using edit models!
- We need better data (e.g. Google Docs), where contributors are working towards a somewhat agreed goal to train better models
- Are there task-specific diffusion formulations that we could learn to combine?
- Ensembling large LMs/humans in the discrete space via iterative refinement (nice for API users; PEER paper does a great job in this direction!)

Thank you!

Q & A

Twitter: @machelreid
Email machelreid@google.com