A Character-Level Length-Control Algorithm for Non-Autoregressive Sentence Summarization

Puyuan Liu, **Xiang Zhang**, Lili Mou

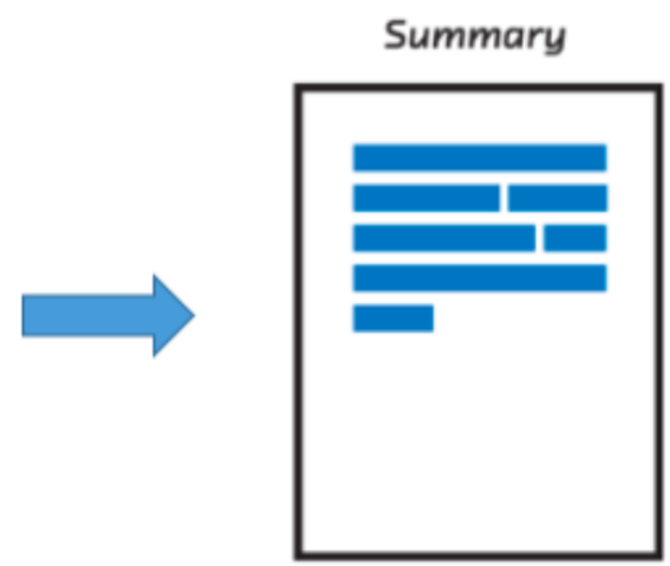
  

puyuan@ualberta.ca, xzhang23@ualberta.ca
doublepower.mou@gmail.com

Dept. Computing Science, University of Alberta
Alberta Machine Intelligence Institute (Amii)

NeurIPS-2022

Summarization Task



Source text

Summary

Summarization Task

- Applications: headline generation
- Granularities:
  - Single-document summarization
  - Multi-document summarization
  - Sentence-level summarization
    Generate summaries for an input sentence

Example: The amphibia, which is the animal class to which our frogs and toads belong, were the first animal to crawl from the sea and inhabit the earth -> The first animals to leave the sea and live on land were the amphibia.

Background

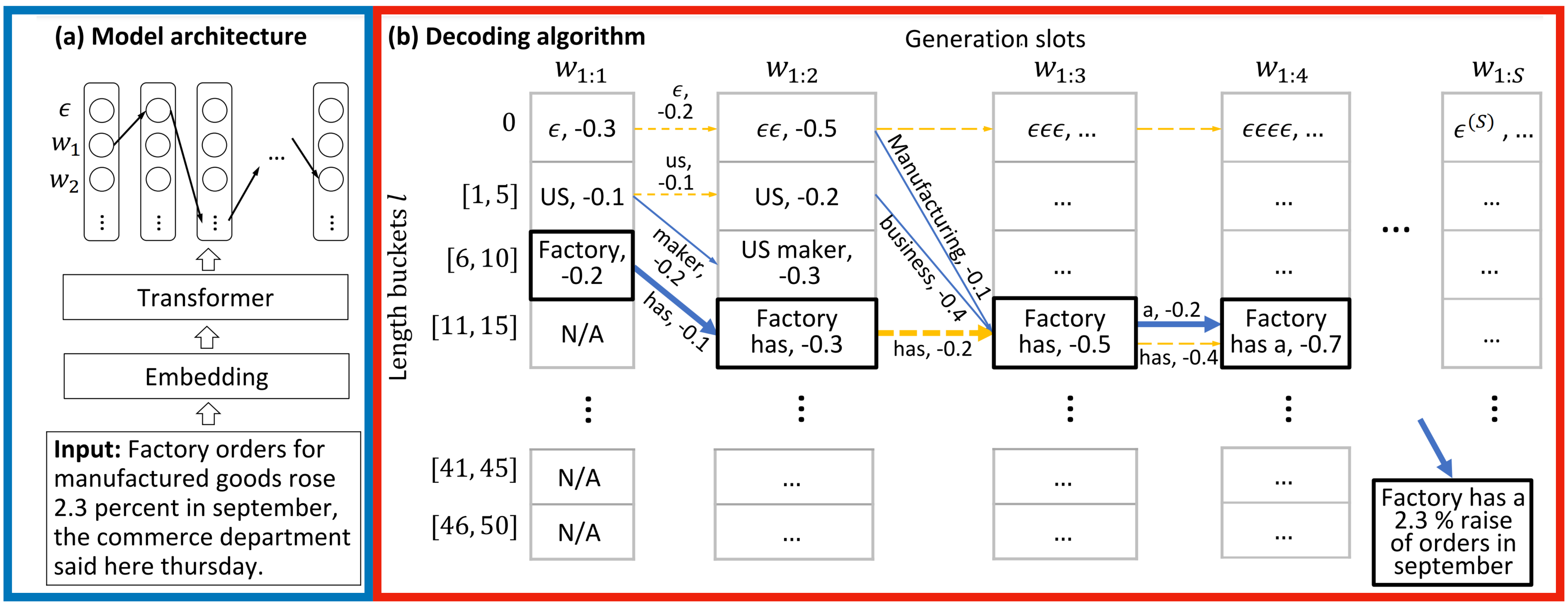
- Length control for text summarization
    - Has real-world applications
    - ROUGE scores being sensitive to the summary length (Itsumi et al., 2020)

Background

- Length control for text summarization
  - Has real-world applications
  - ROUGE scores being sensitive to the summary length (Itsumi et al., 2020)
- Previous length-control methods
  - Only controlling number of words in summaries
  - Cannot explicitly control summary length

Our Approach

- Overview
  1) Non-autoregressive model
  2) Character-level length-control algorithm

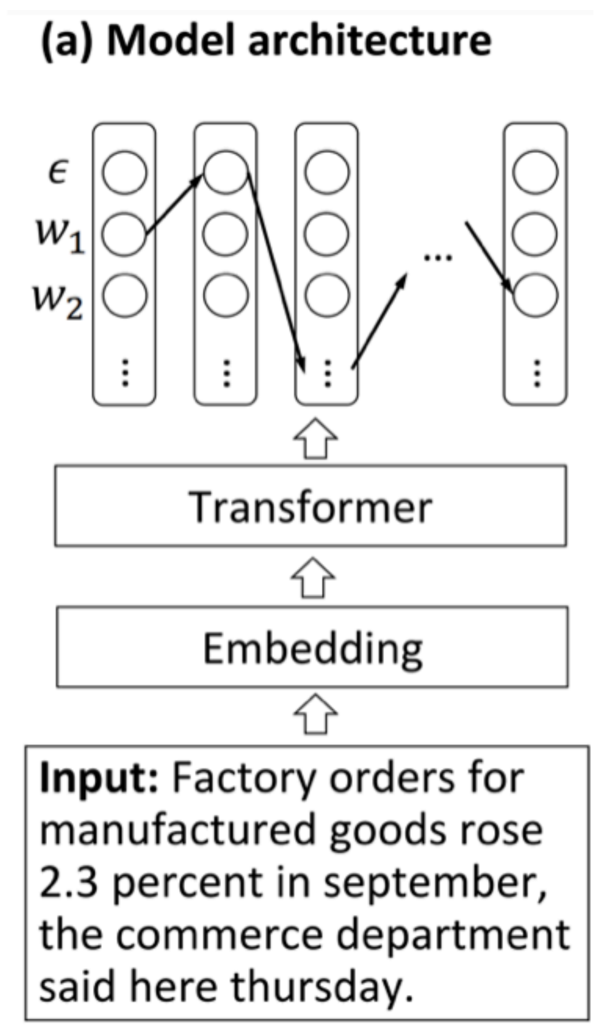


Non-autoregressive model

Character-level length-control algorithm

Our Approach

- Non-autoregressive model
  - Encoder-only architecture
    - Utilizing source—target correspondence
      ⇒ suitable for summarization



Our Approach

- Non-autoregressive model
  - Encoder-only architecture
    - Utilizing source—target correspondence
      ⇒ suitable for summarization
  - Generating at different output slots in parallel
    - High inference efficiency
    - Local predicted probabilities
      ⇒ dynamic programming for length control
    - Independent probability

CTC Loss

- Non-autoregressive mode generates the output of the same length as the input, which can not be summary
    - Padding the target with empty ε
    - **Example:** I ϵ like reading ϵ ϵ ϵ ϵ books
      $\Rightarrow$ I like reading books
- CTC (Graves et al. 2006) Training objective: MLE $\sum_{\mathbf{w}:\Gamma(\mathbf{w})=\mathbf{y}} P(\mathbf{w}|\mathbf{x})$
  - Computed by dynamic programming

Our Approach

- Character-level length control
  - Based on dynamic programming
  - Formulating length control as a Knapsack problem

    ⇒ Number of characters in a word as the weight $v(\cdot)$

    ⇒ Predicted log-probability of a word as the value $u(\cdot)$

$$\underset{\boxed{\mathrm{w}_1,\cdots,\mathrm{w}_S}}{\text{maximize}} \sum_{s=1}^{S} v_s(\mathrm{w}_s), \qquad \text{subject to} \sum_{\substack{\mathrm{y}\in\mathbf{y} \\ \mathbf{y}=\boldsymbol{\Gamma}(\boxed{\mathrm{w}_1,\cdots,\mathrm{w}_S})}} u(\mathrm{y}) < \boxed{U}$$

word sequence

budget

Dynamic programming

- Divide the lengths into buckets for efficient inference
  - lth bucket cover the length ranging from $\alpha \cdot (l - 1) + 1$ to $\alpha \cdot l$ characters

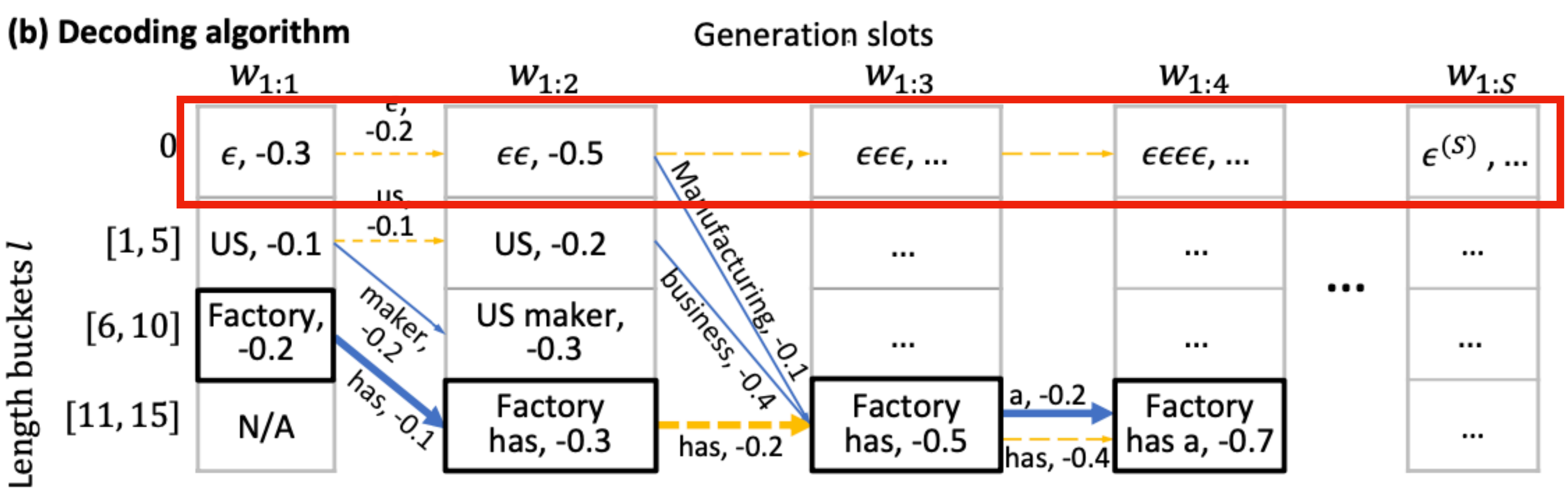
Dynamic programming

- Divide the lengths into buckets for efficient inference.
  - lth bucket cover the length ranging from $\alpha \cdot (l - 1) + 1$ to $\alpha \cdot l$ characters

- Recursion Variables:
  - $\mathbf{d}^{s,l}$ as the most probable s-token sequence that is reduced to a summary in the lth length bucket

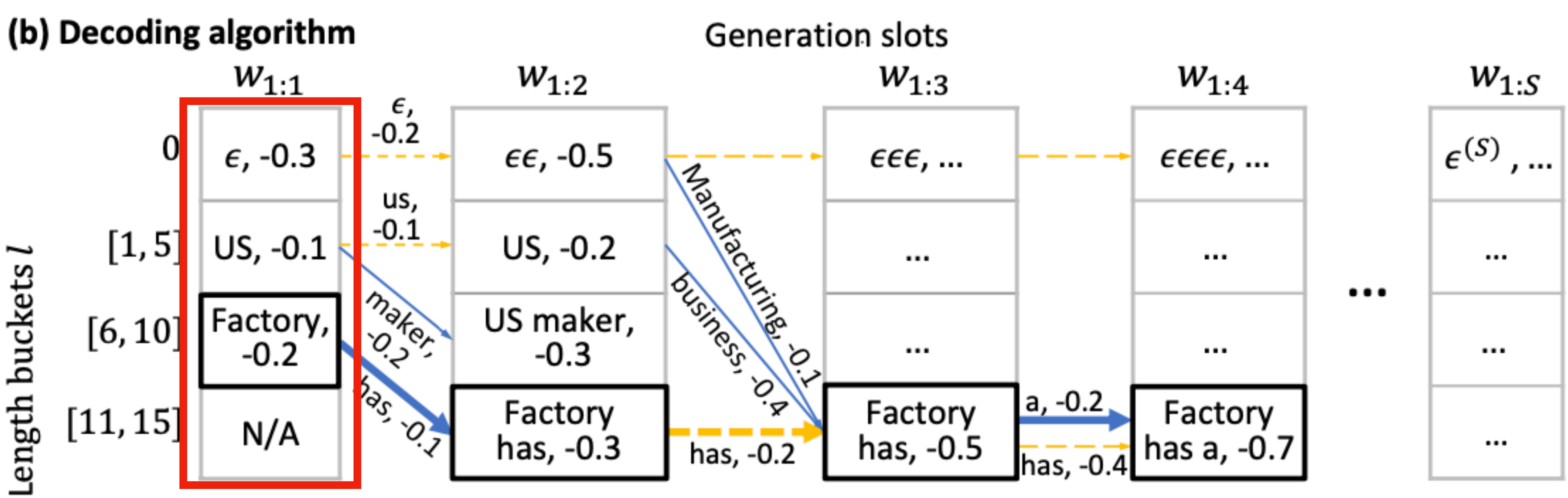
Dynamic programming

- Base Case:
    - $\mathbf{d}^{s,0} = \epsilon \cdots \epsilon$ (s-many)



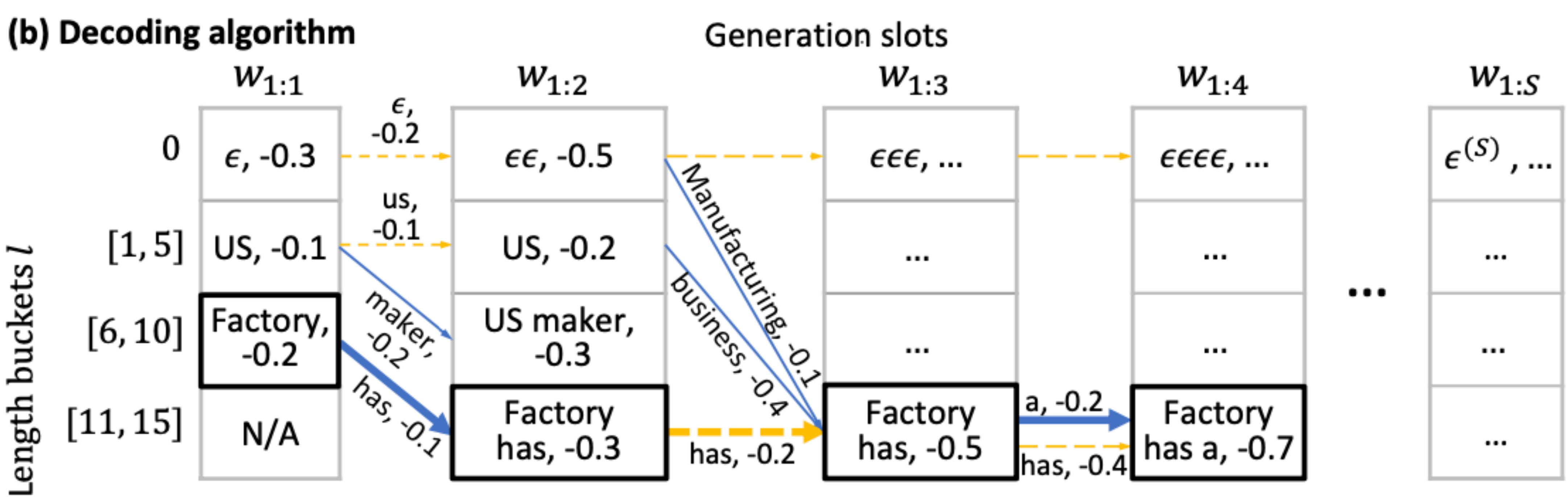
Dynamic programming

- Base Case:
  - $\mathbf{d}^{s,0} = \epsilon \cdots \epsilon$ (s-many)
  - $$\mathbf{d}^{1,l} = \begin{cases} \epsilon, & \text{if } l = 0 \\ \underset{\mathrm{w}:u(\mathrm{w}) \in [\alpha \cdot (l-1)+1, \alpha \cdot l]}{\operatorname{argmax}} v_1(\mathrm{w}), & \text{if } l > 0 \end{cases}$$



Dynamic programming

- Base Case:
  - $\mathbf{d}^{s,0} = \epsilon \cdots \epsilon \; (s\text{-many})$
  - $$\mathbf{d}^{1,l} = \begin{cases} \epsilon, & \text{if } \; l = 0 \\ \underset{\mathrm{w}:u(\mathrm{w}) \in [\alpha \cdot (l-1)+1, \alpha \cdot l]}{\operatorname{argmax}} v_1(\mathrm{w}), & \text{if } \; l > 0 \end{cases}$$



Dynamic programming

- Recursive steps
  - $$\mathscr{D}_1^{s,l} = \left\{\mathbf{d}^{s-1,l} \oplus \epsilon\right\}$$
  - $$\mathscr{D}_2^{s,l} = \left\{\mathbf{d}^{s-1,l} \oplus \mathrm{d}_{s-1}^{s-1,l}\right\}$$
  - $$\mathscr{D}_3^{s,l} = \Big\{\mathbf{d}^{s-1,l'} \oplus \mathrm{w}_s \,:\, \Big(u(\mathrm{w}_s) + \sum_{\mathrm{d}\in\mathbf{d}^{s-1,l'}} u(\mathrm{d})\Big) \in [\alpha\cdot(l-1)+1, \alpha\cdot l],$$ $$\mathrm{w}_s \neq \epsilon, \mathrm{w}_s \neq \mathrm{d}_{s-1}^{s-1,l'},\ \text{and } l' \leq l\Big\}$$

Dynamic programming

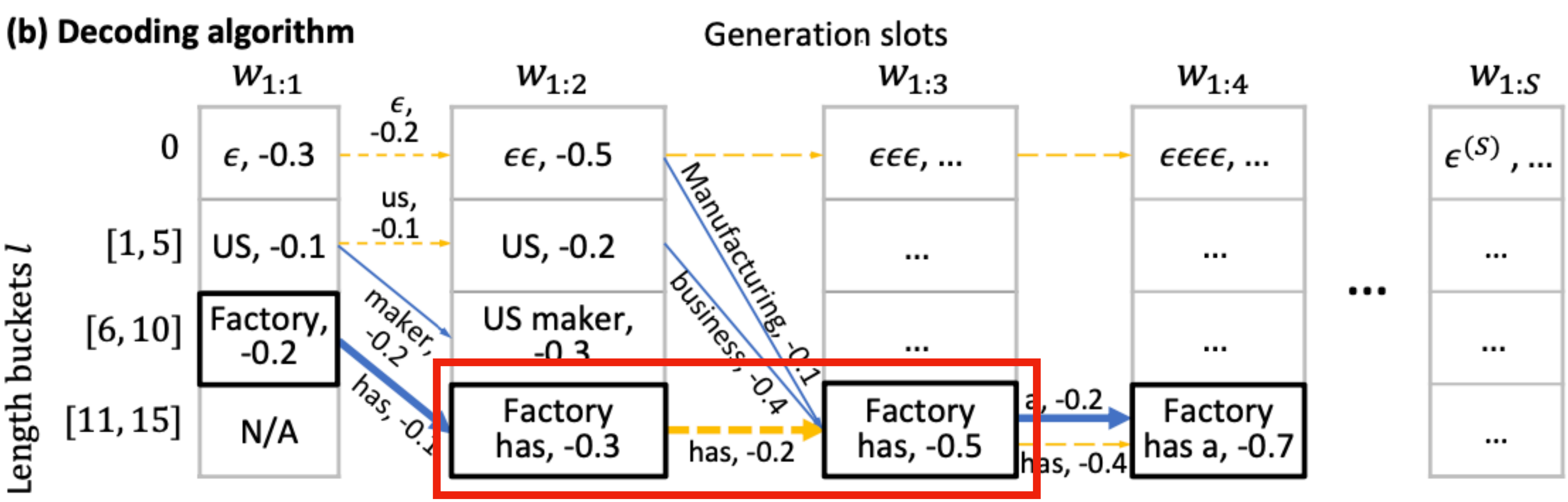
- Recursive steps

- $$\mathscr{D}_2^{s,l} = \left\{ \mathbf{d}^{s-1,l} \oplus \mathrm{d}_{s-1}^{s-1,l} \right\}$$

Dynamic programming

- Recursive steps

- $$\mathscr{D}_2^{s,l} = \{\mathbf{d}^{s-1,l} \oplus \mathrm{d}_{s-1}^{s-1,l}\}$$



Dynamic programming

- Recursive steps
  - $\mathscr{D}_1^{s,l} = \left\{ \mathbf{d}^{s-1,l} \oplus \epsilon \right\}$
  - $\mathscr{D}_2^{s,l} = \left\{ \mathbf{d}^{s-1,l} \oplus \mathrm{d}_{s-1}^{s-1,l} \right\}$
  - $$\mathscr{D}_3^{s,l} = \Bigg\{ \mathbf{d}^{s-1,l'} \oplus \mathrm{w}_s \; : \; \Big(u(\mathrm{w}_s) + \sum_{\mathrm{d} \in \mathbf{d}^{s-1,l'}} u(\mathrm{d}) \Big) \in [\alpha \cdot (l-1) + 1, \alpha \cdot l], \\ \mathrm{w}_s \neq \epsilon, \mathrm{w}_s \neq \mathrm{d}_{s-1}^{s-1,l'}, \text{ and } l' \leq l \Bigg\}$$
- Update recursion variables

$$\mathbf{d}^{s,l} = \underset{\mathbf{d} \in \mathscr{D}_1^{s,l} \cup \mathscr{D}_2^{s,l} \cup \mathscr{D}_3^{s,l}}{\operatorname{argmax}} \sum_{\mathrm{s}=1}^{S} v_s(\mathrm{d}_s)$$

Experiments

- Gigaword

Setting	#		Approach	Len	ROUGE F1 R-1	R-2	R-L	ΔR	Time
Supervised	1	NAR	Su et al. [34] (truncate)	38.43	32.28	**14.21**	30.56	0	0.016
	2		Qi et al. [29] (truncate)	27.98	31.69	12.52	30.05	-2.79	0.019
	3		Yang et al. [41] (truncate)	35.37	28.85	6.45	27.00	-14.75	–
	4		NACC (truncate)	34.15	33.12	13.93	31.34	1.34	**0.011**
	5		NACC (length control)	34.40	**33.66**	13.73	**31.79**	**4.74**	0.017
Unsupervised	6	Baseline	Lead-50 chars	49.03	20.66	7.08	19.30	-9.23	–
	7	Search	Schumann et al. [33] (truncate)	45.45	24.98	9.08	23.18	0.97	9.573
	8		Char constrained search	44.05	25.30	**9.25**	23.43	1.71	17.324
	9	NAR	Su et al. [34] (truncate)	45.24	24.65	8.64	22.98	0	0.017
	10		Qi et al. [29] (truncate)	44.54	24.31	7.66	22.48	-1.82	0.019
	11		Yang et al. [41] (truncate)	49.37	21.70	4.60	20.13	-9.84	–
	12		NAUS [18] (truncate)	47.15	25.71	8.55	23.85	1.84	0.032
	12		NACC (truncate)	47.77	25.79	8.94	23.75	2,21	**0.012**
	13		NACC (length control)	47.03	**27.45**	8.87	**25.14**	**5.19**	0.025

Table 1: Performance on the Gigaword headline generation test set, where NAR stands for non-autoregressive. **Len:** Average number of characters in the predicted summaries. **R-1, R-2, R-L:** ROUGE-1, ROUGE-2, ROUGE-L. **ΔR:** The difference of total ROUGE (sum of R-1, R-2, and R-L) in comparison with the (previous) state-of-the-art NAR summarization system [34]. **Time:** Average inference time in seconds for one sample on an i9-9940X CPU and an RTX6000 GPU.

Experiments

- DUC2004

#	Approach		ROUGE Recall				Time
			R-1	R-2	R-L	ΔR	
1	Baseline	Lead-75 chars	22.52	6.50	19.74	-4.97	–
2	Search	Schumann et al. [33] (truncate)	26.09	**8.03**	22.86	3.25	30.362
3		Char-constrained search	26.30	7.95	22.78	3.30	31.540
4	NAR	Su et al. [34] (truncate)	24.67	7.25	21.81	0	0.017
5		Qi et al. [29] (truncate)	22.79	5.91	20.05	-4.98	0.018
6		NACC (truncate)	26.43	7.86	22.66	3.22	0.012
7		NACC (length control)	**28.37**	7.74	**24.30**	**6.68**	0.030

Table 2: Results on DUC 2004 dataset.

Experiments

- Human evaluation

	Decoding	Wins	Ties	Loses	p-value
Overall quality	Truncate	18%	44%	38%	0.0001
	Length control	**38%**	44%	**18%**	
Completeness & fluency	Truncate	22%	36%	42%	0.0002
	Length control	**42%**	36%	**22%**	

Table 3: Human evaluation comparing truncating and length-control decoding of our NACC approach on 150 samples selected from the Gigaword headline generation dataset in the unsupervised setting. The p-value is given by a two-sided binomial test.

Experiments

- Length-transfer generation
  - Generating summaries of different numbers of characters than the training target

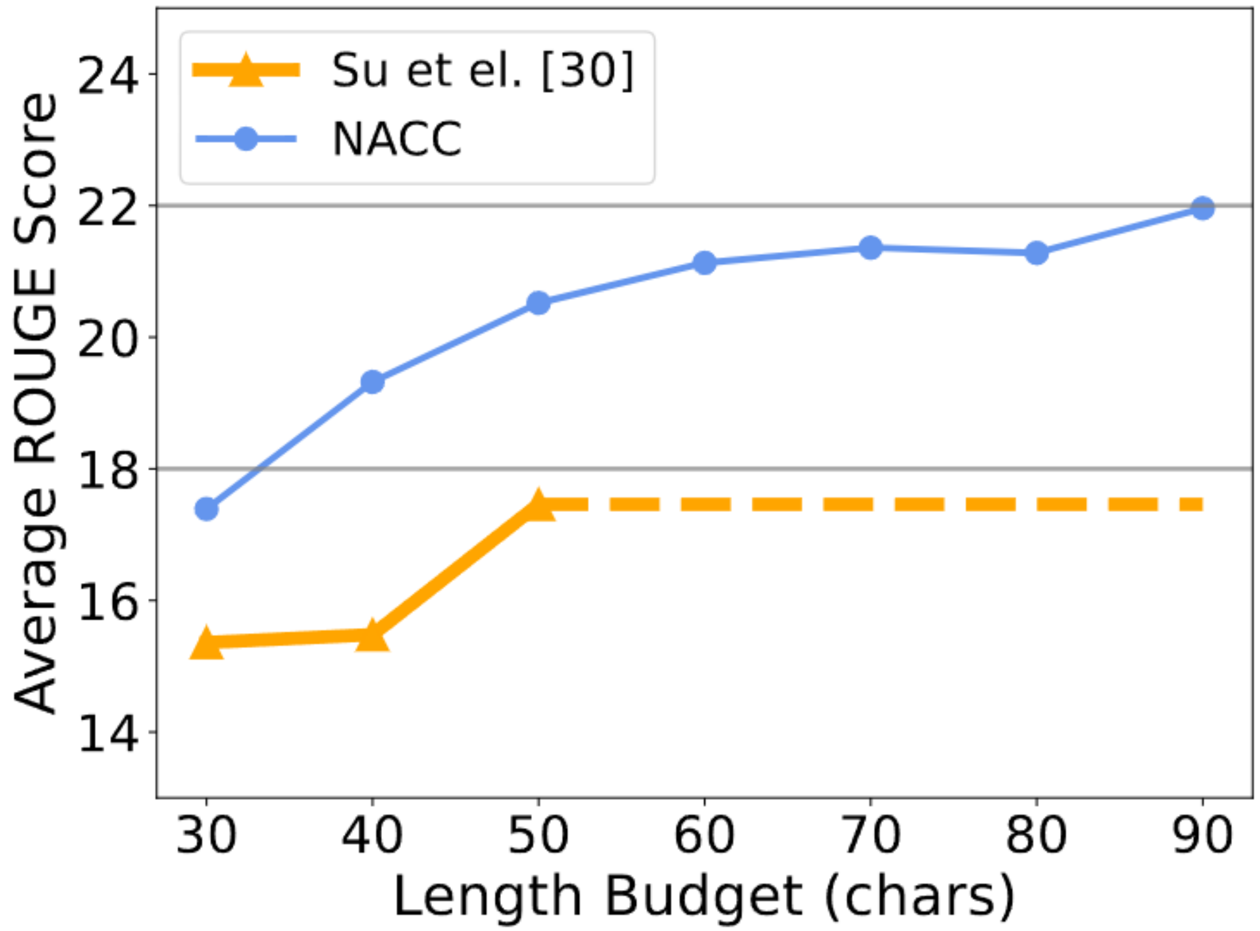


Figure 3: Length-transfer performance of NACC and Su et al. [34].

Thank you!

Acknowledgments

The research is supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant No. RGPIN2020-04465, the Amii Fellow Program, the Canada CIFAR AI Chair Program, a UAHJIC project, a donation from DeepMind, and Compute Canada (www.computecanada.ca).

References

Itsumi Saito, Kyosuke Nishida, Kosuke Nishida, Atsushi Otsuka, Hisako Asano, Junji Tomita, Hiroyuki Shindo, and Yuji Matsumoto. Length-controllable abstractive summarization by guiding with summary prototype. In: arXiv preprint arXiv:2001.07331, 2020.

Yixuan Su, Deng Cai, Yan Wang, David Vandyke, Simon Baker, Piji Li, and Nigel Collier. Non-autoregressive text generation with pre-trained language models. In EACL, pages 234–243, 2021.