Imputer:

Sequence Modeling via Imputation and Dynamic Programming

William Chan, Chitwan Saharia, Geoffrey Hinton, Mohammad Norouzi, Navdeep Jaitly
Google Brain

Presenter: Desh Raj

Overview

- Preliminary: end-to-end ASR model types
- The Imputer model
  - Training scheme
  - Decoding policies
- Experimental results

Preliminary

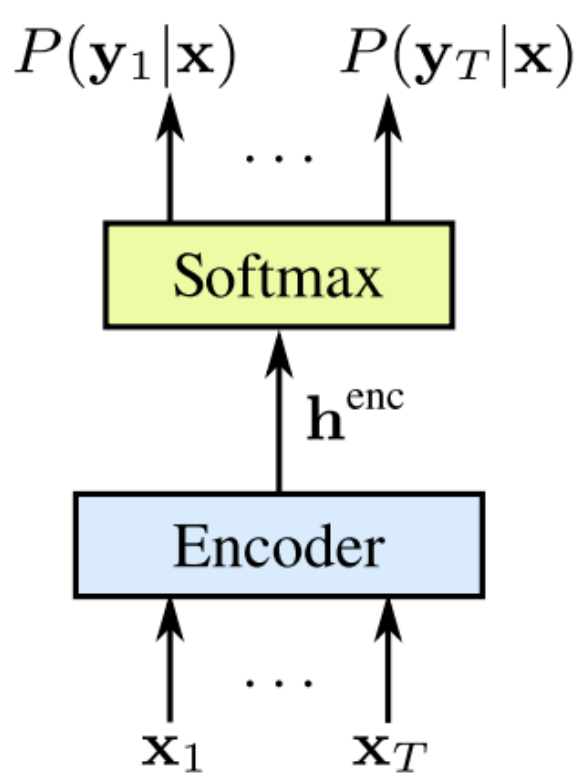
- 3 kinds of popular end-to-end ASR models:

    1. CTC-based

    2. RNN-Transducer

    3. Encoder-decoder (with attention)

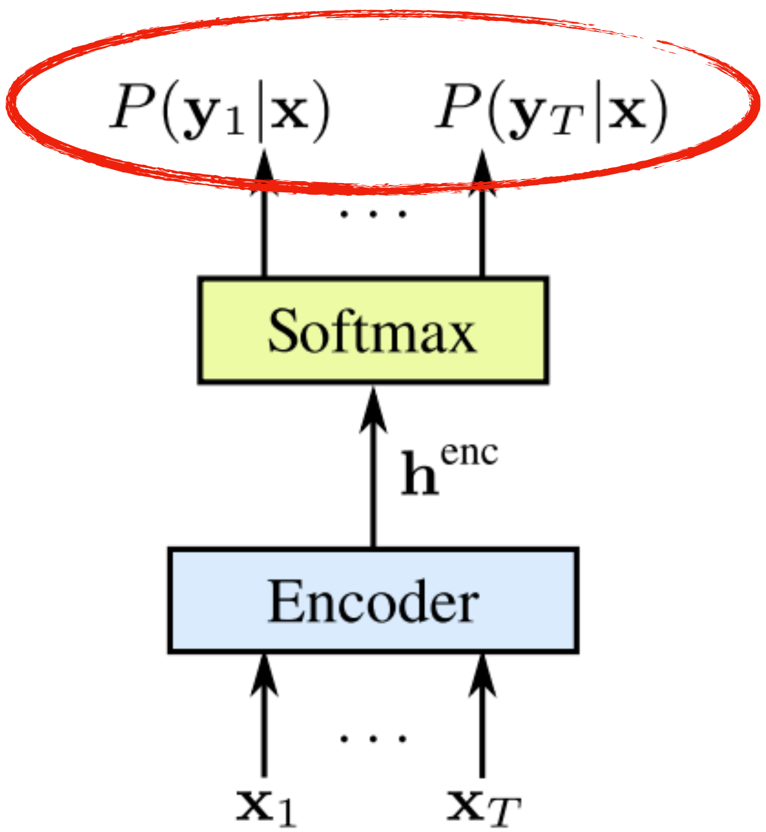
CTC



$$P(\mathbf{y}|\mathbf{x}) = \sum_{\hat{\mathbf{y}} \in \mathcal{B}(\mathbf{y},\mathbf{x})} \prod_{t=1}^{T} P(\hat{y}_t|\mathbf{x})$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

CTC



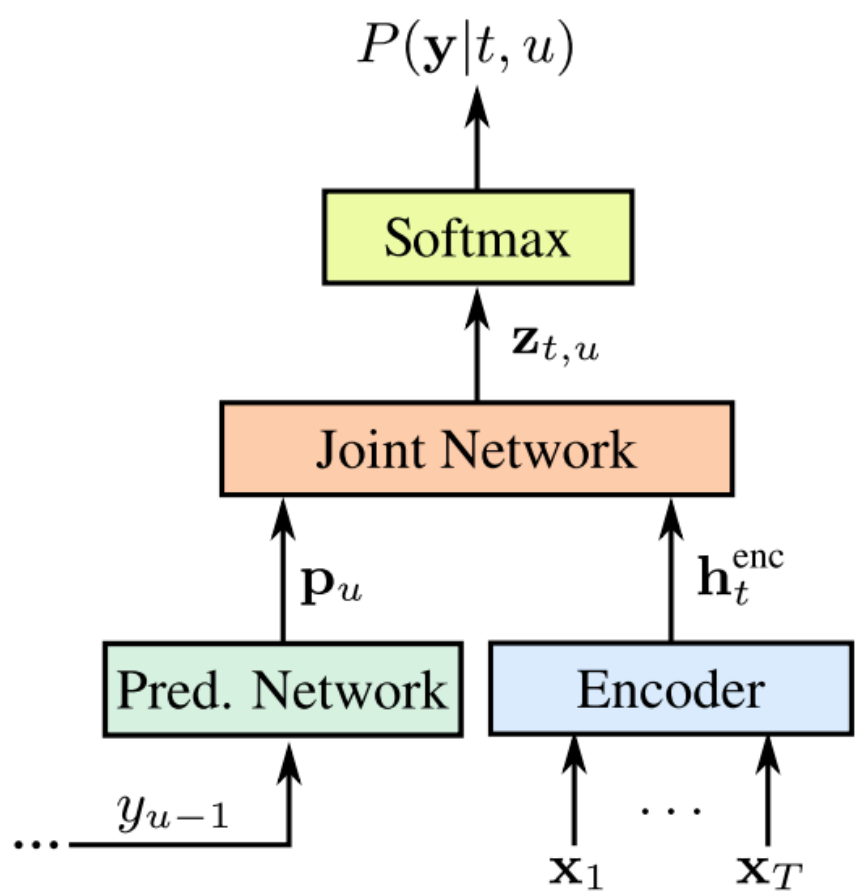
Conditional independence assumption

So all tokens can be generated in parallel

$$P(\mathbf{y}|\mathbf{x}) = \sum_{\hat{\mathbf{y}} \in \mathcal{B}(\mathbf{y},\mathbf{x})} \prod_{t=1}^{T} P(\hat{y}_t|\mathbf{x})$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

RNN-Transducer

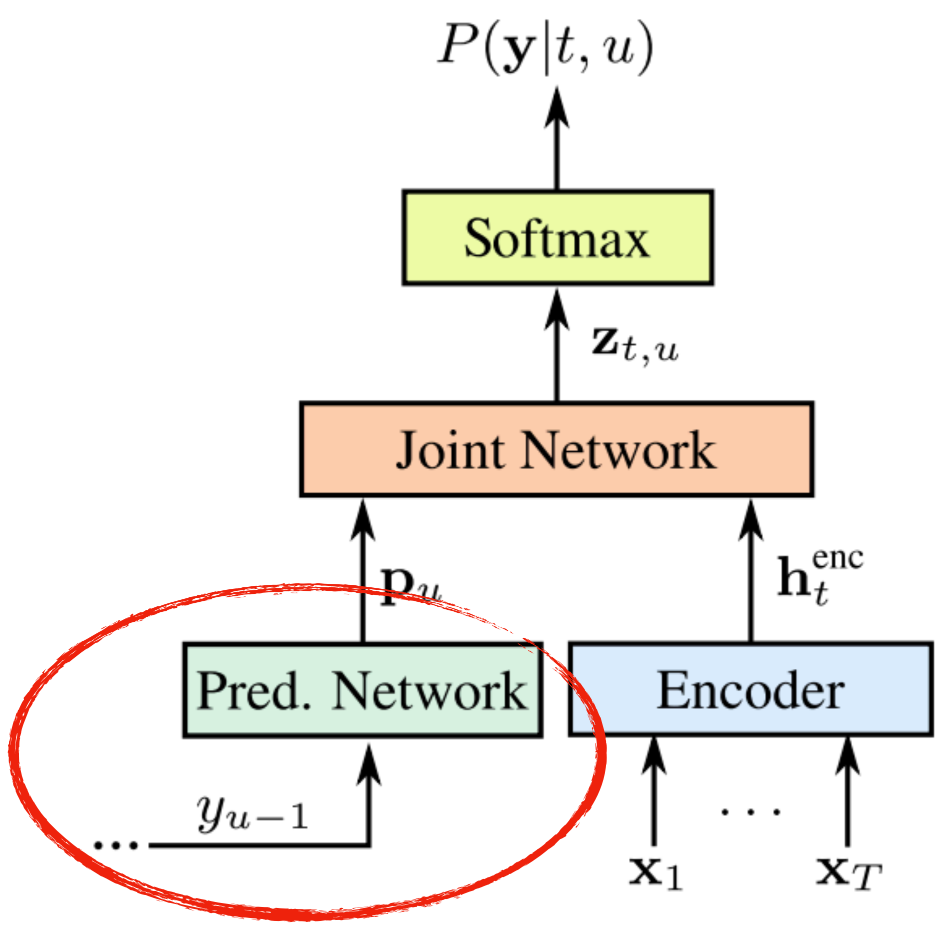


$$\mathbf{h}_{t,u}^{\text{joint}} = \tanh(A\mathbf{h}_t^{\text{enc}} + B\mathbf{p}_u + b)$$

$$\mathbf{z}_{t,u} = D\mathbf{h}_{t,u}^{\text{joint}} + d$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

RNN-Transducer



Generated token depends on previous output

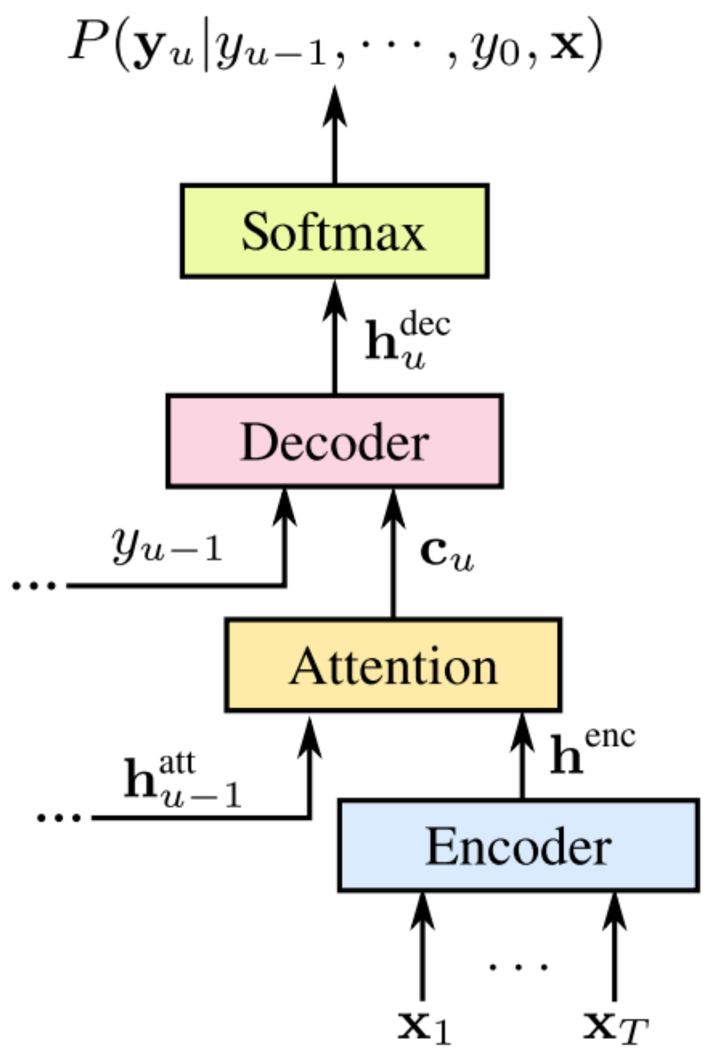
So have to generate sequentially

$$\mathbf{h}_{t,u}^{\text{joint}} = \tanh(A\mathbf{h}_t^{\text{enc}} + B\mathbf{p}_u + b)$$

$$\mathbf{z}_{t,u} = D\mathbf{h}_{t,u}^{\text{joint}} + d$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

Encoder-decoder (with attention)



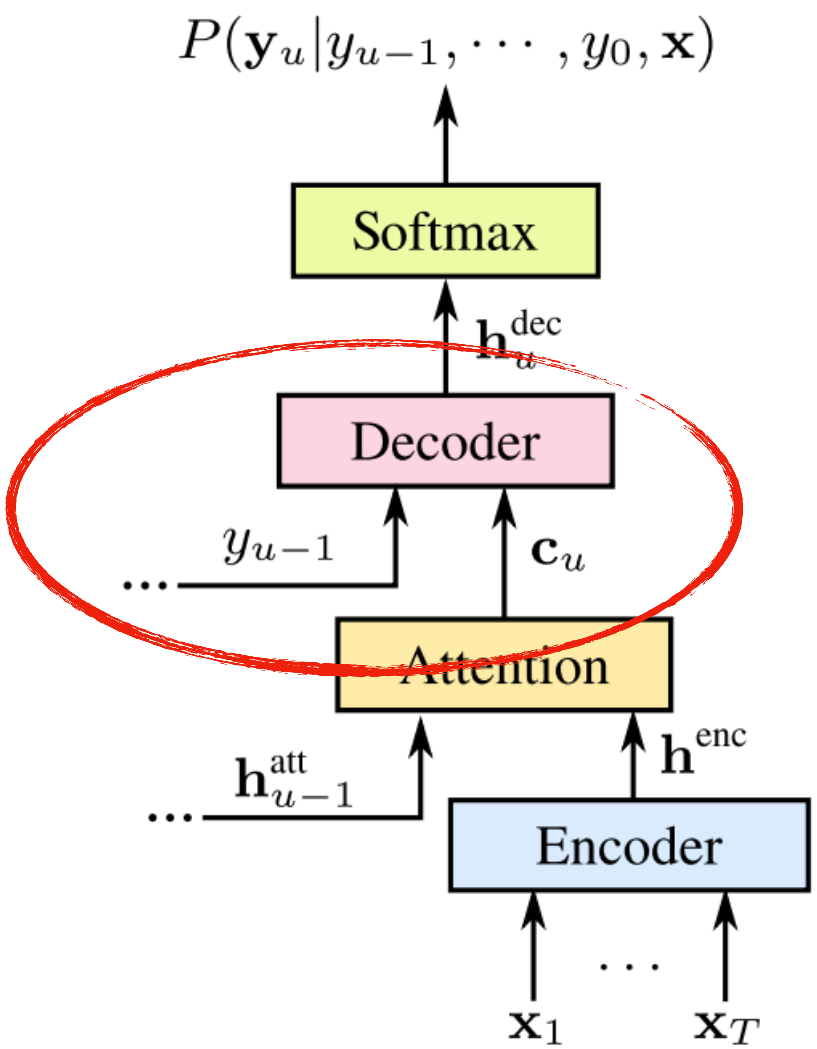
$$\beta_{t,u} = \langle \phi(\mathbf{h}_t^{\text{enc}}), \psi(\mathbf{h}_{u-1}^{\text{att}}) \rangle$$

$$\alpha_{t,u} = \frac{e^{\beta_{t,u}}}{\sum_{i=1}^{T} e^{\beta_{i,u}}}$$

$$\mathbf{c}_u = \sum_t \alpha_{t,u} \mathbf{h}_t^{\text{enc}}$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

Encoder-decoder (with attention)



Generated token depends on previous output

So have to generate sequentially

$$\beta_{t,u} = \langle \phi(\mathbf{h}_t^{\mathrm{enc}}), \psi(\mathbf{h}_{u-1}^{\mathrm{att}}) \rangle$$

$$\alpha_{t,u} = \frac{e^{\beta_{t,u}}}{\sum_{i=1}^{T} e^{\beta_{i,u}}}$$

$$\mathbf{c}_u = \sum_t \alpha_{t,u} \mathbf{h}_t^{\mathrm{enc}}$$

A comparison of sequence-to-sequence models for speech recognition. Prabhavalkar et al. Interspeech 2017.

Trade-off

- Speed vs. performance tradeoff
- For **faster inference**, sequence generation must be independent of previous tokens e.g. CTC
- But for **better performance**, conditionally dependent sequence generation is required e.g. RNN-T

Trade-off

- Speed vs. performance tradeoff
- For **faster inference**, sequence generation must be independent of previous tokens e.g. CTC
- But for **better performance**, conditionally dependent sequence generation is required e.g. RNN-T

Autoregressive

Trade-off

- Speed vs. performance tradeoff

Non-autoregressive

- For **faster inference**, sequence generation must be independent of previous tokens e.g. CTC
- But for **better performance**, conditionally dependent sequence generation is required e.g. RNN-T

Autoregressive

Motivating Question

- Can we have something in between?
- A model which is not fully autoregressive (i.e. does not take O(n) steps during inference)
- But also does not have conditional independence assumptions

“Yes, we can.”

– Authors of the Imputer paper

A bit more on CTC

$$p(Y \mid X) = \sum_{A \in \mathcal{A}_{X,Y}} \prod_{t=1}^{T} p_t(a_t \mid X)$$

The CTC conditional **probability**

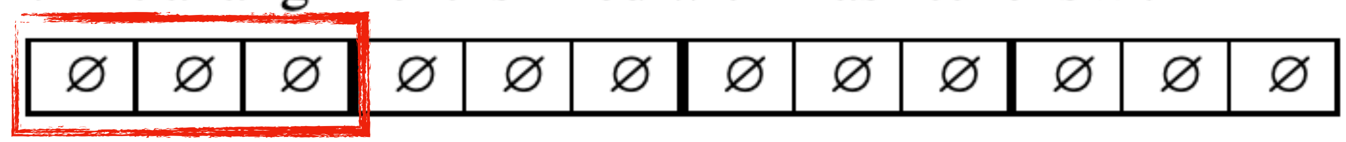
marginalizes over the set of valid alignments

computing the **probability** for a single alignment step-by-step.

Sequence modeling with CTC. Awni Hannun. https://distill.pub/2017/ctc/

The Imputer Decoding Model

1. Initial alignment is filled with mask tokens ∅.

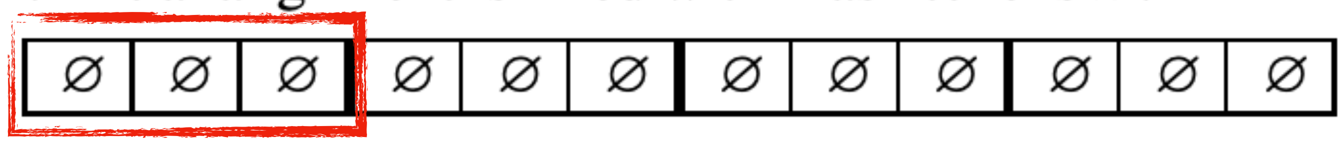


Output sequence length = |x| is divided into blocks of size B

A block is generated independent of other blocks

The Imputer Decoding Model

1. Initial alignment is filled with mask tokens ∅.



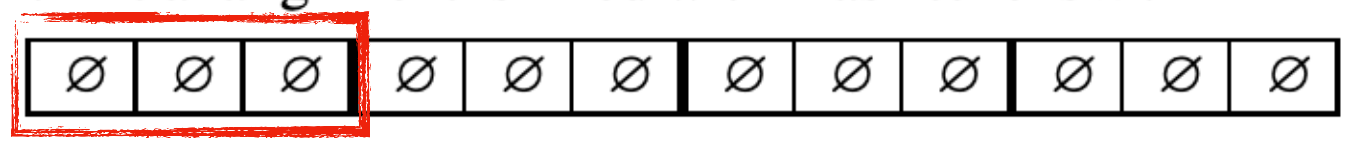
Output sequence length = |x| is divided into blocks of size B

A block is generated independent of other blocks

B = 1 ?

The Imputer Decoding Model

1. Initial alignment is filled with mask tokens ∅.



Output sequence length = |x| is divided into blocks of size B

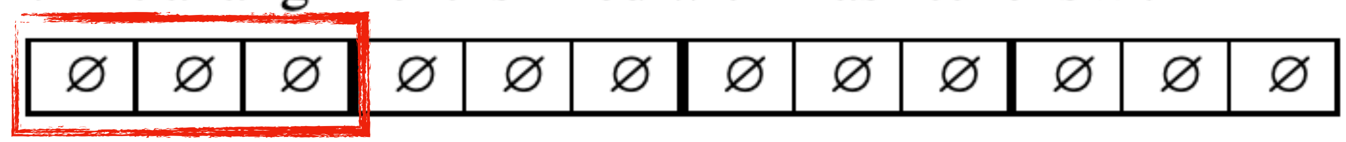
A block is generated independent of other blocks

B = 1 ? **CTC**

B = |x| ?

The Imputer Decoding Model

1. Initial alignment is filled with mask tokens ∅.



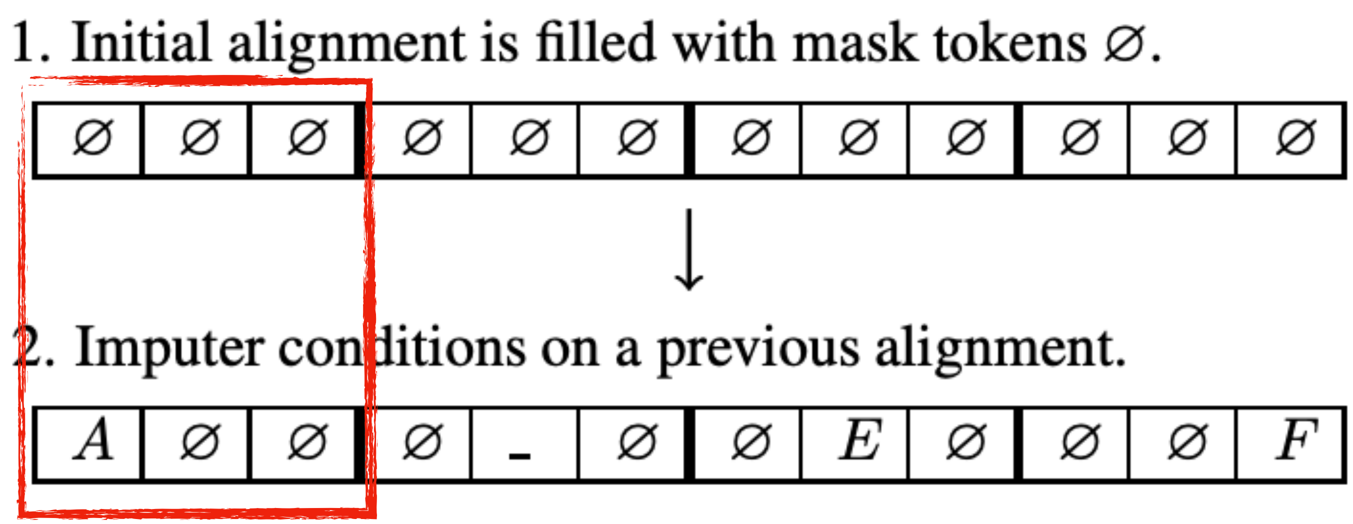
Output sequence length = |x| is divided into blocks of size B

A block is generated independent of other blocks

B = 1 ? **CTC**

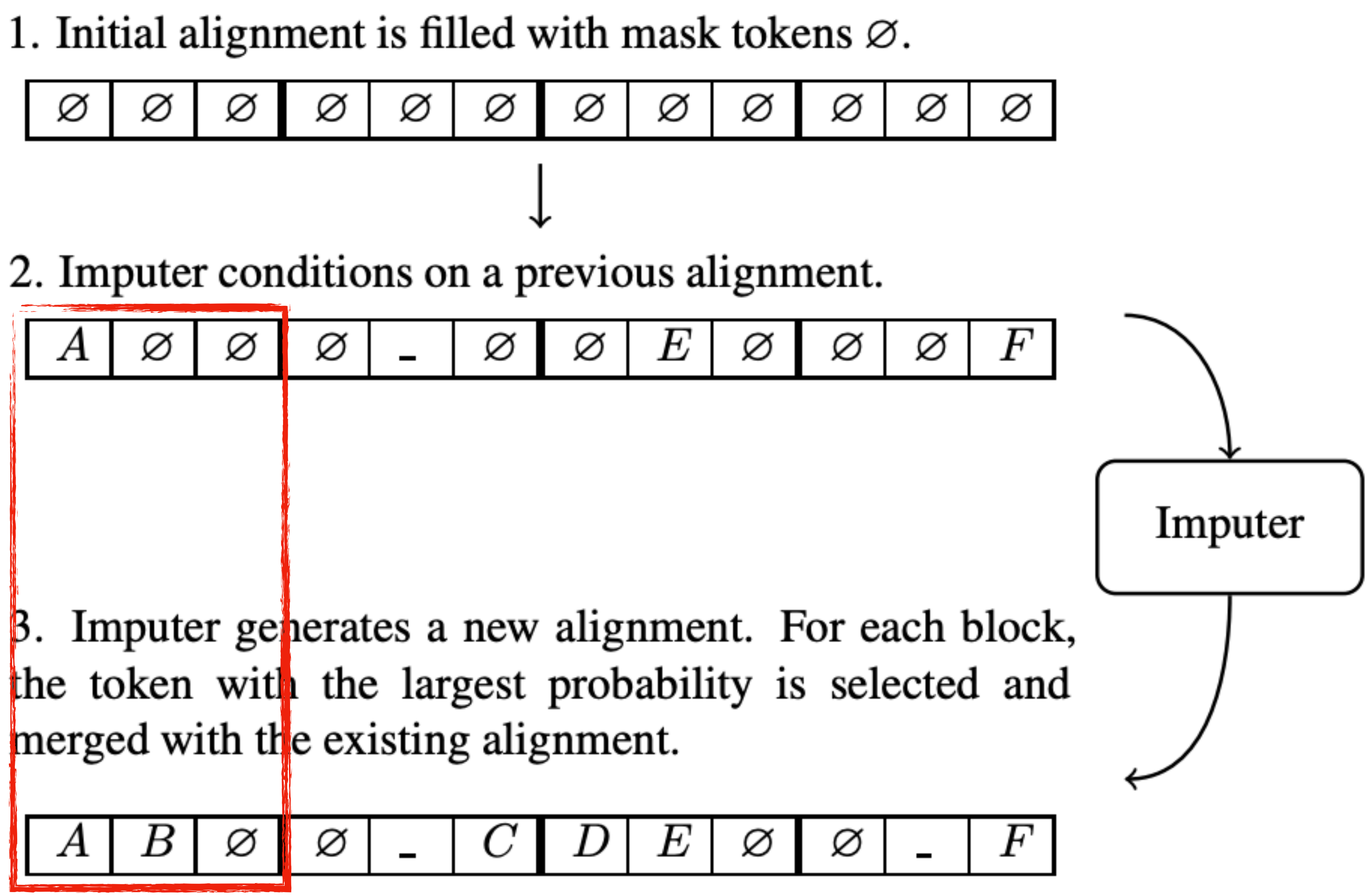
B = |x| ? **Fully autoregressive**

The Imputer Decoding Model



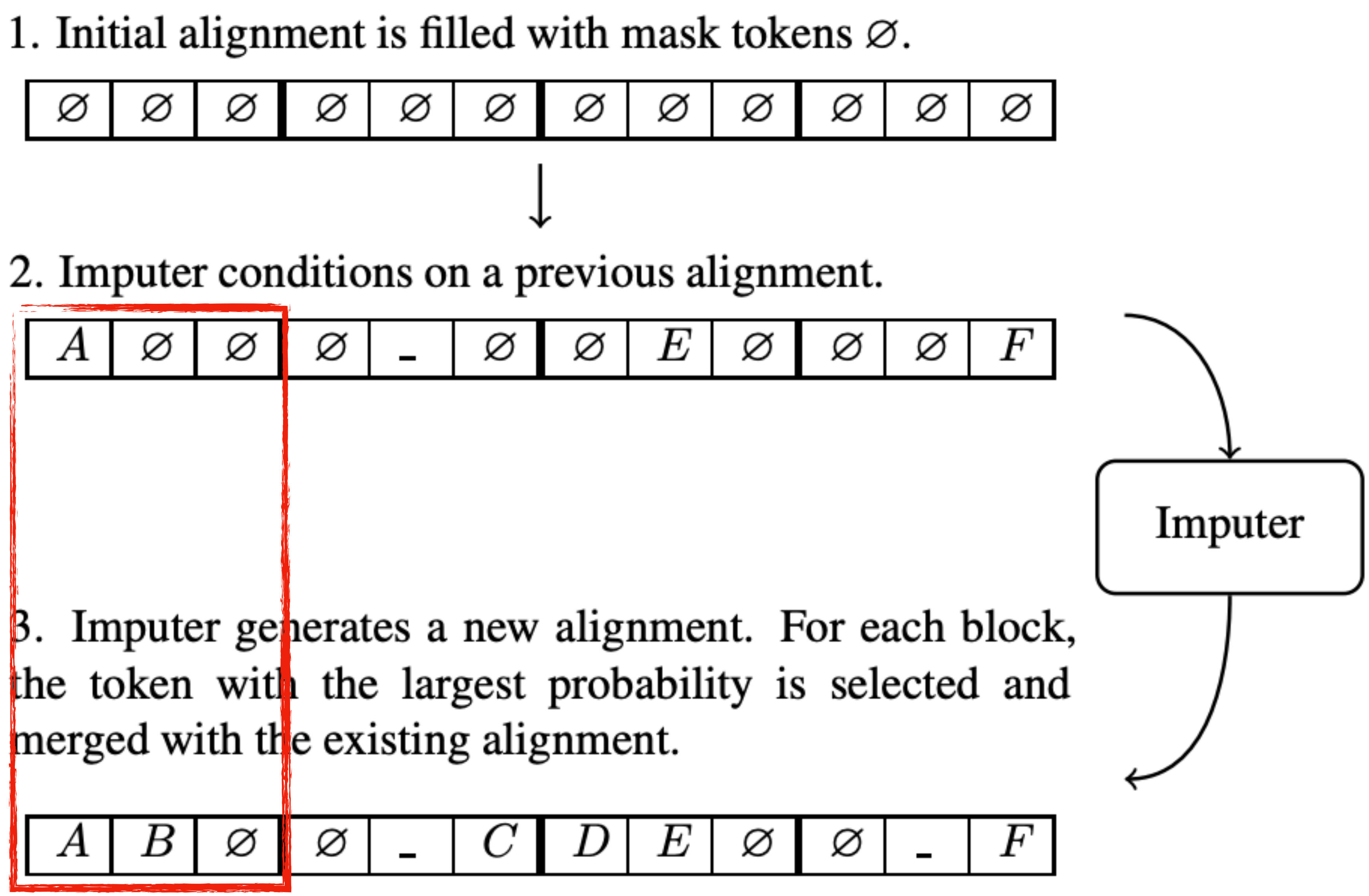
New alignment for the block is generated depending on input x and previous alignment

The Imputer Decoding Model



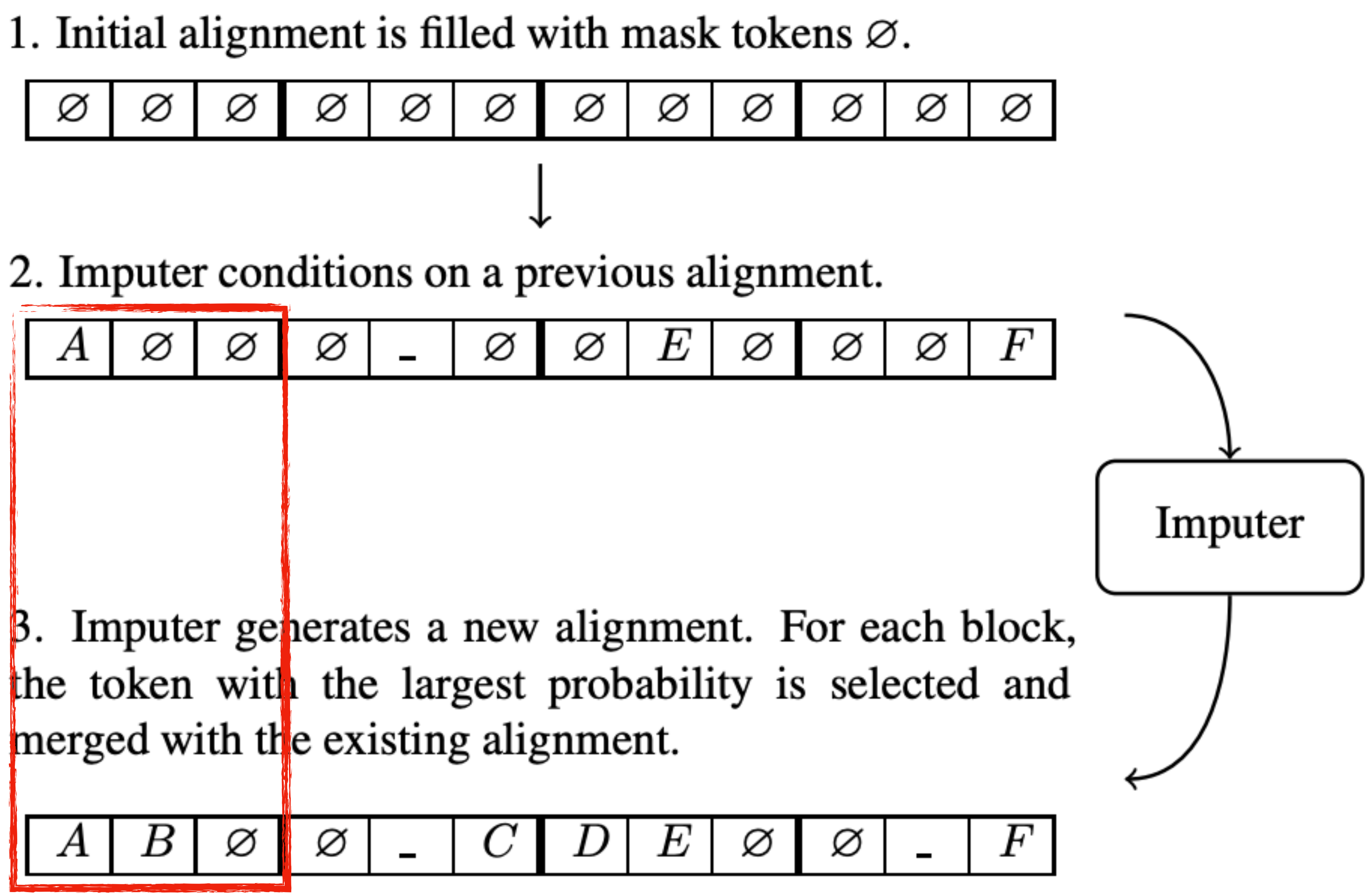
Keep the token with largest probability and merge with existing alignment

The Imputer Decoding Model



1 token "committed" in each step

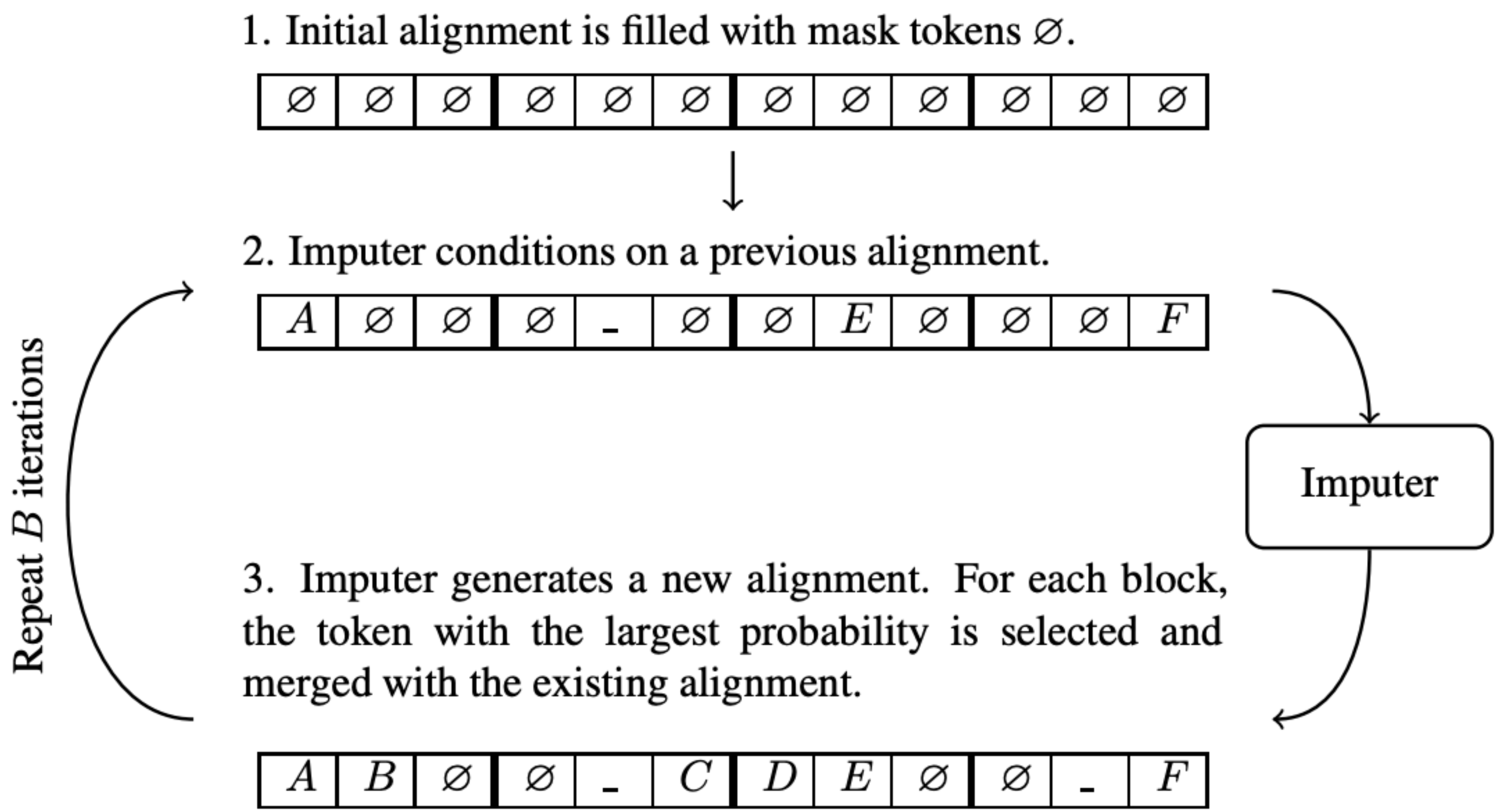
The Imputer Decoding Model



1 token "committed" in each step

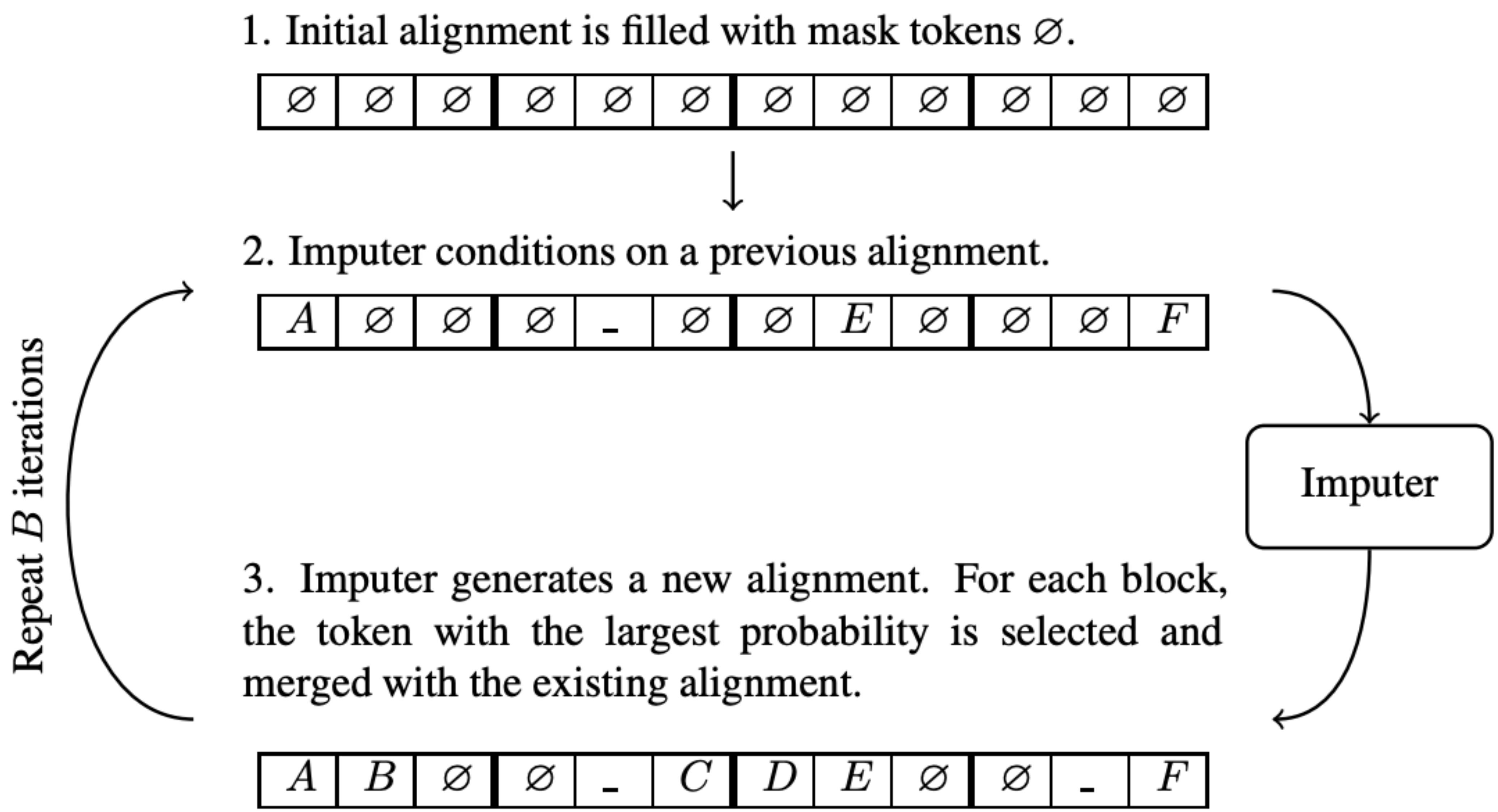
So how many steps would we need in total to get the sequence?

The Imputer Decoding Model



Sequence is generated in B steps

The Imputer Decoding Model



In a block, token generation is dependent on other tokens within the block

The model

$$p_\theta(a|\tilde{a}, x) = \prod_i p(a_i|\tilde{a}, x; \theta)$$

The model

$$p_\theta(a|\tilde{a}, x) = \prod_i p(a_i|\tilde{a}, x; \theta)$$

New alignment

Previous alignment

By conditioning on previous alignment, the tokens become conditioned on each other

Training objective

$$\log p_\theta(y|x) = \log \sum_{a \in \beta(y)} \sum_{\tilde{a} \in \gamma(a)} p_\theta(a|\tilde{a}, x)$$

Training objective

$$\log p_\theta(y|x) = \log \sum_{a \in \beta(y)} \sum_{\tilde{a} \in \gamma(a)} p_\theta(a|\tilde{a}, x)$$

All alignments of y

All masking permutations

Training objective

$$J(\theta) = \mathbb{E}_{a\sim q}[\mathbb{E}_{\tilde{a}\sim r}[\log p_\theta(a|\tilde{a}, x)]]$$

All alignments of y

All masking permutations

Training objective

$$J(\theta) = \mathbb{E}_{a \sim q}[\mathbb{E}_{\tilde{a} \sim r}[\log p_\theta(a|\tilde{a}, x)]]$$

1. **How to sample alignments from q?**
2. **How to sample masks from r?**

Alignment policy

- Suppose we have an “expert” model pretrained e.g. CTC
- **Method 1**: Get all alignments, store them offline, and sample from this.
- **Method 2**: Get best alignment and add noise

$$J(\theta) = \mathbb{E}_{a \sim q}[\mathbb{E}_{\tilde{a} \sim r}[\log p_\theta(a|\tilde{a}, x)]]$$

Masking policy

- **Method 1**: Uniform or Bernoulli distribution -> every block may have different number of masked tokens

- **Method 2**: Choose b in [0,B) and mask out b tokens in each block randomly

$$J(\theta) = \mathbb{E}_{a\sim q}[\mathbb{E}_{\tilde{a}\sim r}[\log p_\theta(a|\tilde{a}, x)]]$$

How to train your Imputer?

1. **Imitation learning**

- Simply learn to copy the expert CTC model

$$J_{\mathrm{IM}}(\theta) = \mathbb{E}_{a\sim q_{\phi'}}\left[\mathbb{E}_{\tilde{a}\sim r}\left[\log p_\theta(a|\tilde{a}, x)\right]\right]$$

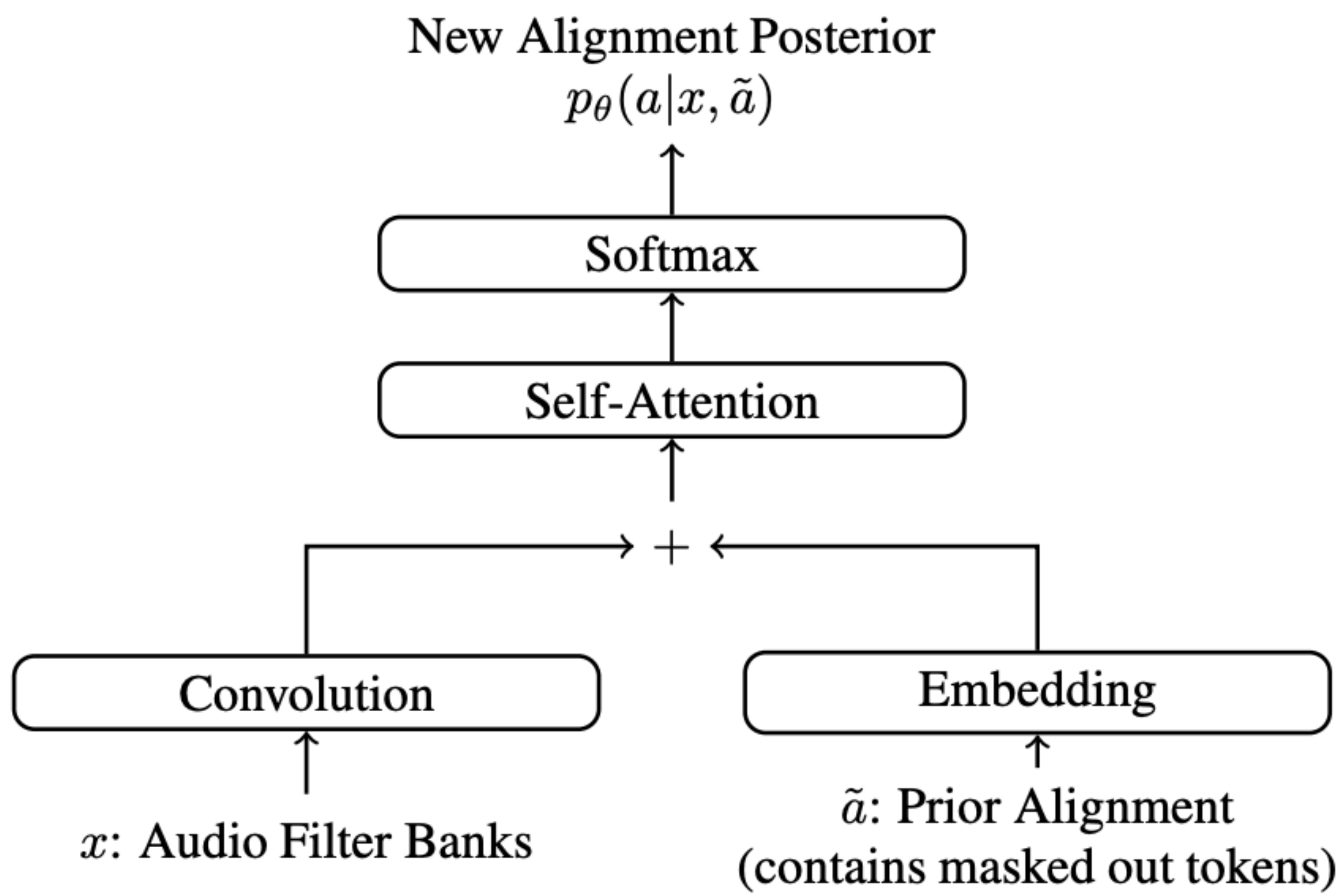
How to train your Imputer?

2. **Dynamic programming**

- Marginalize over all possible compatible alignments

$$J_{\mathrm{DP}}(\theta) = \mathbb{E}_{a \sim q_{\phi'}} \left[\mathbb{E}_{\tilde{a} \sim r} \left[\log \sum_{a' \in \beta'(\tilde{a}, a)} p_\theta(a' | \tilde{a}, x) \right] \right]$$

Model architecture


New Alignment Posterior
$p_\theta(a|x,\tilde{a})$
Softmax
Self-Attention
+
Convolution
Embedding
x: Audio Filter Banks
$\tilde{a}$: Prior Alignment
(contains masked out tokens)

Experimental results

WSJ - 82h

Table 1. Wall Street Journal Character Error Rate (CER) and Word Error Rate (WER).

Model	CER	WER	Iterations
seq2seq			
Bahdanau et al. (2016a)	6.4	18.6	n
Bahdanau et al. (2016b)	5.9	18.0	n
Chorowski & Jaitly (2017)	-	10.6	n
Zhang et al. (2017)	-	10.5	n
Chan et al. (2017)	-	9.6	n
Kim et al. (2017)	7.4	-	n
Serdyuk et al. (2018)	6.2	-	n
Tjandra et al. (2018)	6.1	-	n
Sabour et al. (2019)	3.1	9.3	n
CTC			
Graves & Jaitly (2014)	8.4	27.3	1
Liu et al. (2017)	-	16.7	1
CTC (Our Work)	5.6	15.2	1
Imputer (IM)	6.2	16.5	8
Imputer (DP)	4.9	12.7	8

Librispeech - 960h

Table 2. LibriSpeech test-clean and test-other Word Error Rate (WER).

Method	clean	other	Iterations
seq2seq			
Zeyer et al. (2018a)	4.9	15.4	n
Zeyer et al. (2018b)	4.7	15.2	n
Irie et al. (2019)	4.7	13.4	n
Sabour et al. (2019)	4.5	13.3	n
Luscher et al. (2019)	4.4	13.5	n
Park et al. (2019)	4.1	12.5	n
ASG/CTC			
Collobert et al. (2016)	7.2	-	1
Liptchinsky et al. (2017)	6.7	-	1
CTC (Our Work)	4.6	13.0	1
Imputer (IM)	5.5	14.6	8
Imputer (DP)	4.0	11.1	8

Some other stuff

- Block size 8 found to be best for training and inference
- Too large blocks can cause training issues
- All masking policies (Bernoulli, Uniform, Block) perform similar

Stuff which did not work :)

- Training with stale model samples instead of CTC expert
- Greedy decoding, simulated annealing decoding

Key takeaways

- Want something between CTC and fully autoregressive models
- Inference in constant time (B steps) + works better than both
- How to make things non-autoregressive? Use MASKING. (see BERT, Mask-Predict etc.)


thank you
gracias
merci
danke
obrigado
謝謝
ngiyabonga
teşekkür ederim
рахмат
Баярлалаа
спасибо
bedankt
dziękuję
go raibh maith agat
mochchakkeram
tapadh leat
sukriya
kop khun krap
terima kasih
arigatô
dakujem
grazie
takk
taiku
sagolun
maururu
hvala
dank je
mersi
barka
welalin
tack
spas
kia ora
faafetai lava
vinaka
спасибі
blagodaram
misaotra
matondo
paldies
grazzi
mahalo
хвала
asante
manana
obrigada
murakoze
tenki
kiitos
dankie
nanni
nandri
dhanyavad
köszönöm
gràçie
bayarlalaa
enkosi
akun
dankon
aciū
djiere dieuf
tau
дякую
mamnun
chokrane
chnorakaloutioun
gratias ago
gràcies
sulpáy
sobodi
děkuji
mèsi
didi madloba
najis tuke
kam sah hamnida
rahmat
ありがとう
tanemirt
rahmet
ευχαριστώ
diolch
dhanyavadagalu
shukriya
mercé
мерси
trugarez
তোমাকে ধন্যবাদ
감사합니다
xièxie