PRINCETON UNIVERSITY

Machine Translation

Elisabetta Cavallo, Ben Dodge

COS598C - Deep Learning for Natural Language Processing

February 25th, 2020

Sequence to Sequence Learning with Neural Networks

Sutskever I., Vinyals O., V. Le Q. (2014)

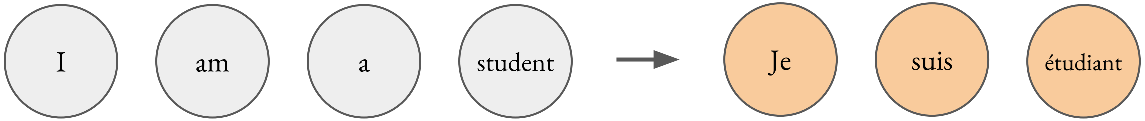
Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyasu@google.com

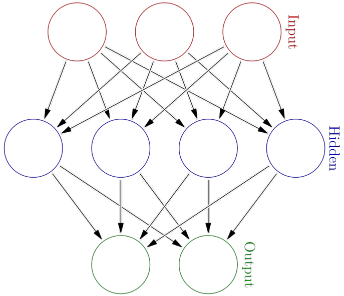
Oriol Vinyals
Google
vinyals@google.com

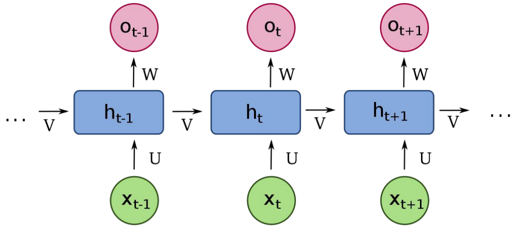
Quoc V. Le
Google
qvl@google.com

Goal:



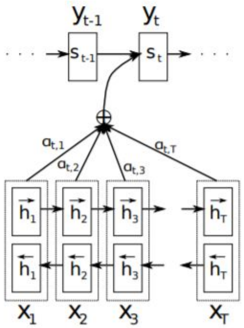
With:

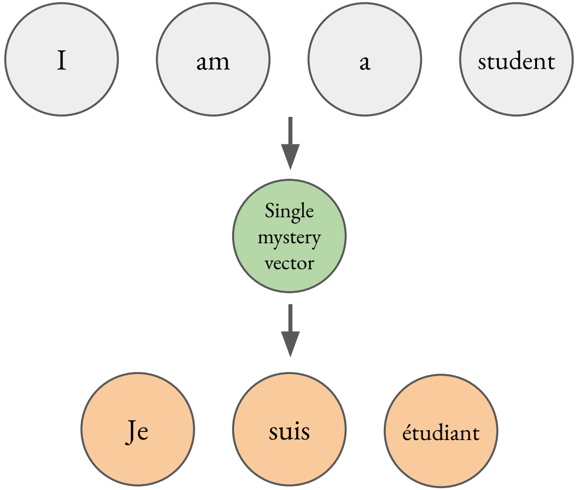




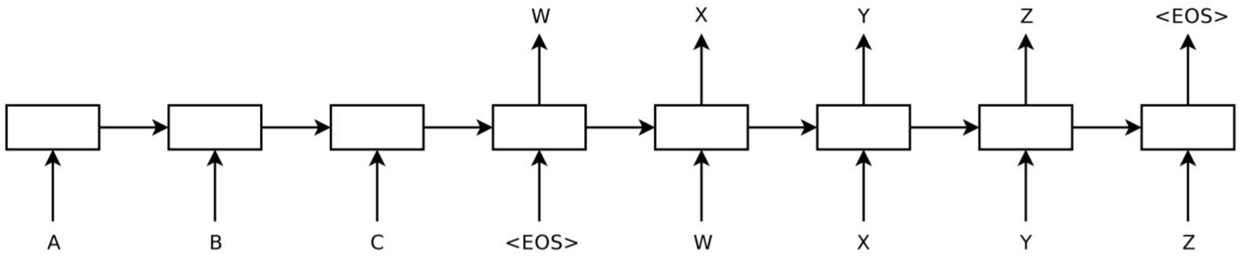
Background

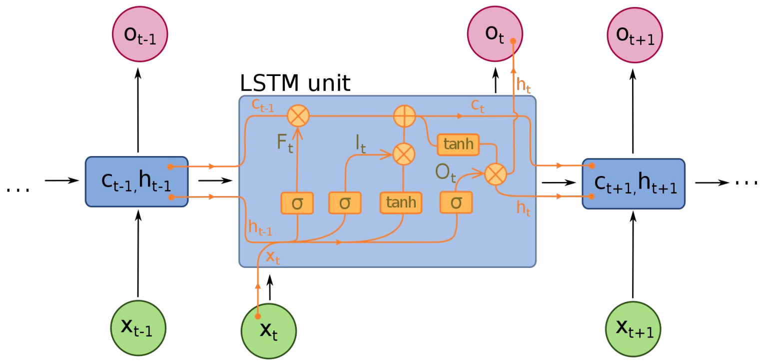
- Statistical Machine Translation (SMT)
- RNN to rescore baseline translations
- Encode and decode a fixed-size vector
  - Using **CNN**: Kalchbrenner and Blunsom (2013)
  - Integrating into **SMT**: Cho et al. (2014)
  - Using **Attention**: Bahdanau et al. (2014)





The Basic Model

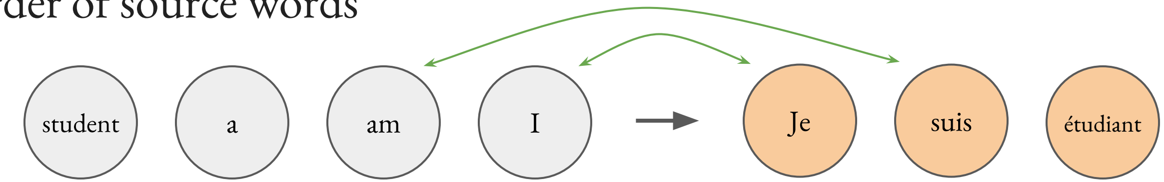




4 layers
1000 cells
1000d embeddings
380M parameters

Extra Bits

- Reverse order of source words



- Separate encoder/decoder parameters
- Beam search decoding
- Ensemble of models

Empirical Results

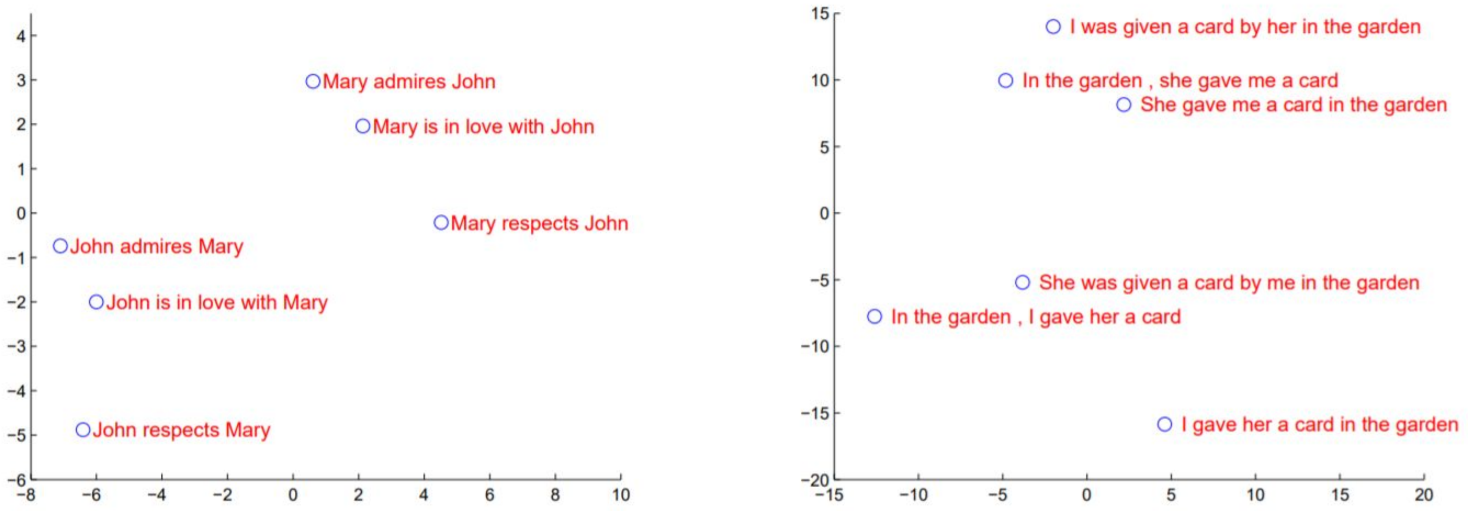
Neural Network Only (except SMT baseline)

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Neural Network Rescoring SMT

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
State of the art [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	~45

Qualitative Results



Model learns the **meaning** of sentences, even with complex reordering

Machine Translation Datasets!

- Workshop on Statistical Machine Translation
  - **WMT '14 English to French** (36M sentence pairs)
  - Mostly from the **Europarl** corpus
  - Also has En↔De, En↔Hi, En↔Cs, En↔Ru, …
- International Workshop on Spoken Language Translation (IWSLT)
- "Google-internal production datasets" (Wu et al., 2016)

Non-Autoregressive Neural Machine Translation

Gu J., Bradbury J., Xiong C., O.K. Li V., Socher R. (2018)

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Autoregressive models

Idea: sequential model which decodes the output depending on the input words the words previously generated (i.e. translated).

- In formulas, given a source sentence $X = \{x_1, ..., x_{T'}\}$ (e.g. *I am a student*) and an output sentence $Y = \{y_1, ..., y_T\}$ (e.g. *Je suis étudiant*):

$$p_{\mathcal{AR}}(Y|X;\theta) = \prod_{t=1}^{T+1} p(y_t|y_{0:t-1}, x_{1:T'};\theta)$$

✓ **Pros**: usually fluent as it corresponds to the word-by-word nature of human language production, easy to train, state-of-the-art performance on large-scale corpora, but…

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Autoregressive models

✘ **…Cons:** unidirectional conditioning on previously translated words, beam search suffers from diminishing returns with respect to beam size, NOT parallelizable at inference, etc.

Naive solution: why not generating output words independent of previously translated words?

$$p_{\mathcal{NA}}(Y|X;\theta) = p_L(T|x_{1:T'};\theta) \cdot \prod_{t=1}^{T} p(y_t|x_{1:T'};\theta)$$

Assumption: target sequence length T can be modelled with a conditional distribution p_L

Probability of a token conditionally independent of previous tokens

Multimodality problem: complete conditional independence is a poor approximation to the true target distribution (i.e. some translation could be equally likely while not both correct).

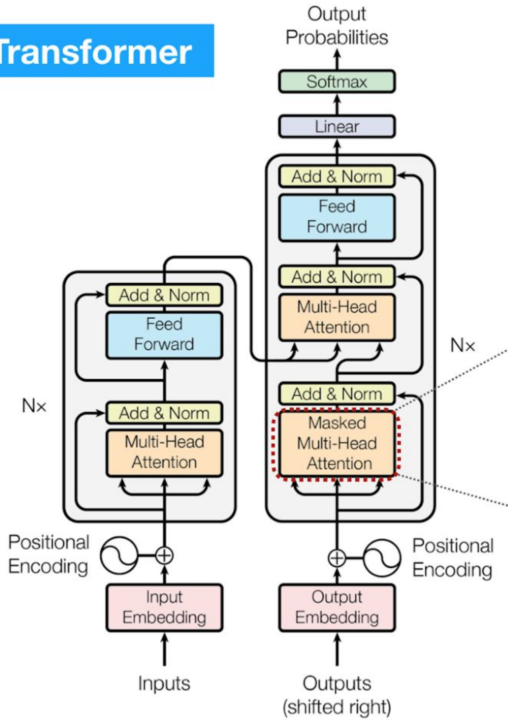
→ e.g. $P(Danke\ schon\,|\,Thank\ you) = P(Vielen\ Dank\,|\,Thank\ you) = \ldots P(Danke\ dank\,|\,Thank\ you)$ *which isn't German…*

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

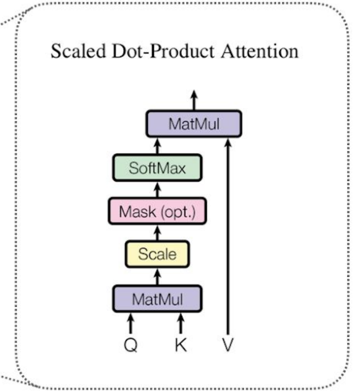
State of Art AR Neural Model before Gu et Al.

Neural Machine Translation with Transformers and Self-Attention by Vaswani et al., 2017

Transformer



Self-attention



$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where:

Q = **query** (vector representation of one word in the sequence)

K = **keys** (matrix representations of all the words in the sequence)

V = **values** (matrix representations of all the words in the sequence)

d_k = **dimension** of queries and keys

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

State of Art AR Neural Model before Gu et Al.

Animation from: https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html?m=1

Non-Autoregressive Transformer (NAT) by Gu et Al.

Idea (Gu et Al., 2018): **Non-autoregressive translation model based on a Transformer network** (Vaswani et al., 2017)**, with modified encoder to predict fertilities.**

→ can produce translations of an entire sentence at a time in a fully parallel way.

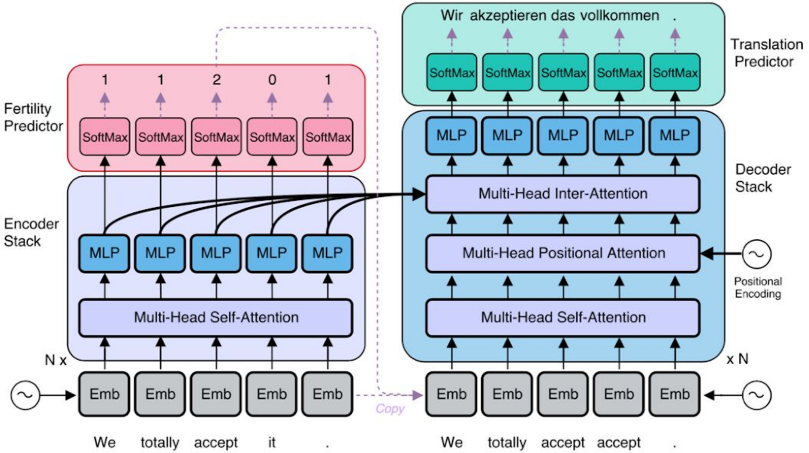


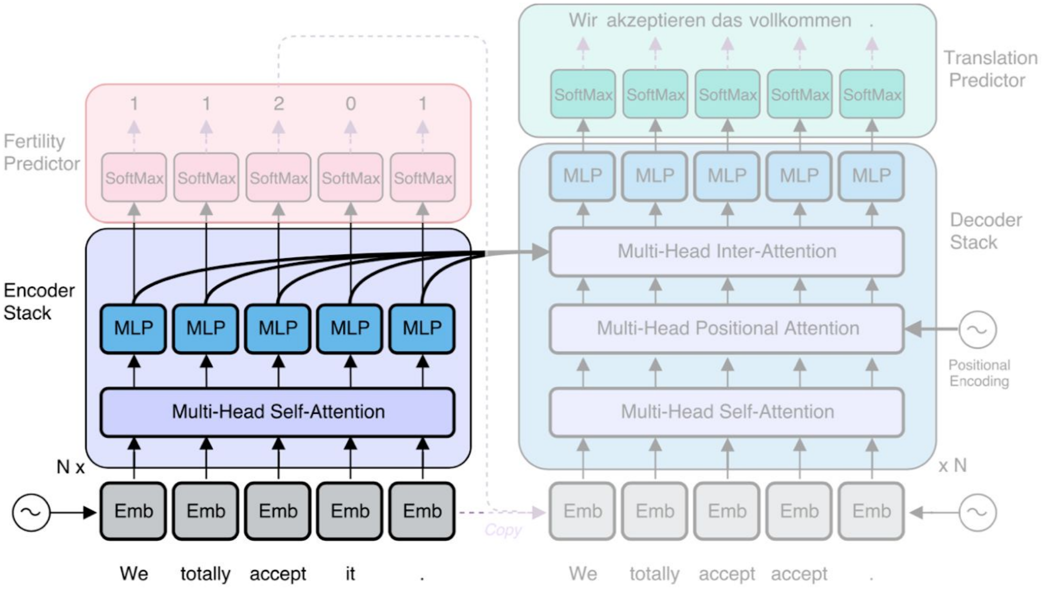
Figure 2: The architecture of the NAT, where the black solid arrows represent differentiable connections and the purple dashed arrows are non-differentiable operations. Each sublayer inside the encoder and decoder stacks also includes layer normalization and a residual connection.

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Encoder Stack

Structure: Multi-Head Self-Attention modules + Feedforward NN (MLP) *(same as in Vaswani et al., 2017)*

→ no RNN = no inherent requirement for sequential execution.



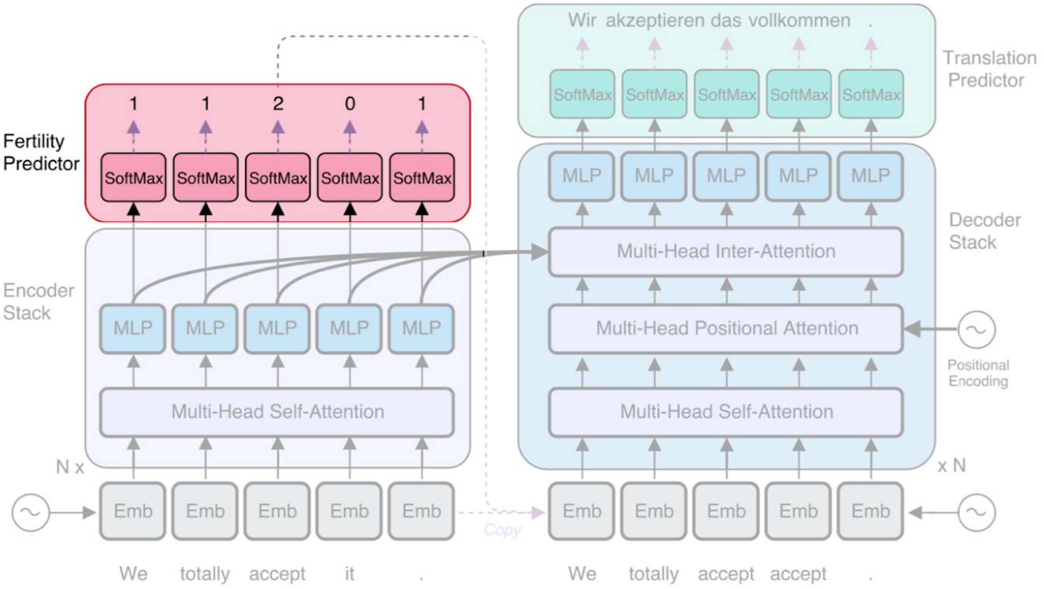
Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Encoder Stack → **Fertility predictor**

The encoder + fertility predictor have two jobs:

1) understanding and interpreting the input sentence,

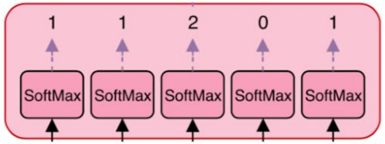
2) **predicting a sequence of numbers** (e.g. [1, 1, 2, 0, 1]) **called fertilities** that are used as input to the parallel decoder.



What are fertilities?

Fertilities are latent variables which represent how many output words each input word generates.

e.g. "Please" has fertility 4 for French translation ("S'il te plait")



Fertility $p_F\,(f_{t'}|x_{1:T'})$ is modelled at each position independently using a one-layer neural network with a softmax classifier on top of the output of the last encoder layer.

→ fertility values are a **property of each input word** while depending on **information and context from the entire sentence**.

→ but… ***what about reordering of words?***

$$p_{\mathcal{NA}}(Y|X;\theta) = \sum_{f_1,\ldots,f_{T'}\in\mathcal{F}} \left(\prod_{t'=1}^{T'} p_F(f_{t'}|x_{1:T'};\theta) \cdot \prod_{t=1}^{T} p(y_t|x_1\{f_1\},..,x_{T'}\{f_{T'}\};\theta) \right)$$

token for x_i repeated f_i times

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

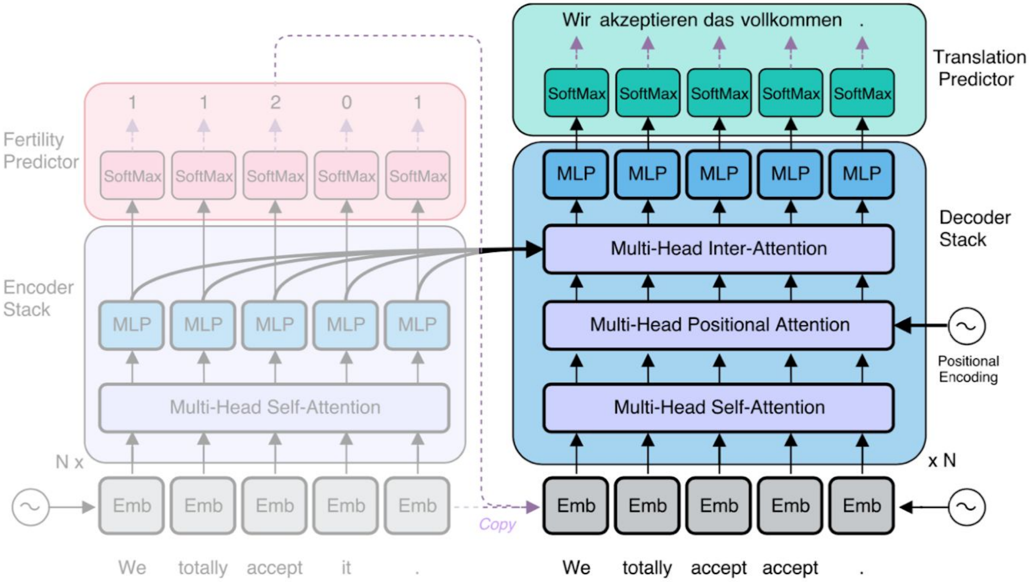
Encoder Stack → Fertility predictor → **Decoder Stack**

Structure: Copied source inputs from the encoder side using ***fertilities***

+

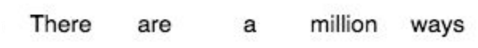
Self-attention layers and MLP

(Non-causal + Positional + Inter-Attention)



Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

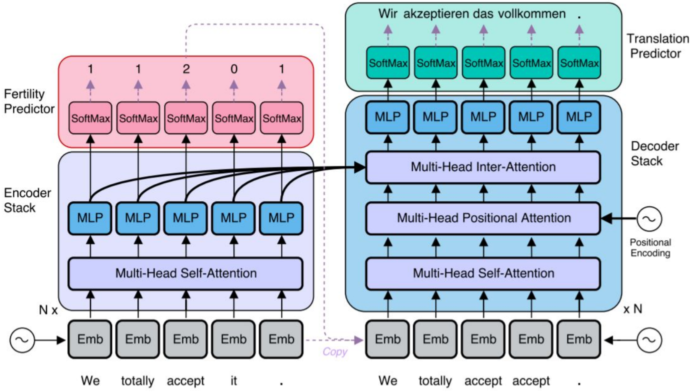
Encoder Stack → Fertility predictor → Decoder Stack: **in Action!**



Animation from:
https://blog.einstein.ai/fully-parallel-text-generation-for-neural-machine-translation/?fbclid=IwAR3VX1ZCn3ArjBnAeWmkxvELOknMvtlh9Hfd-rzEa0ovYmi_OFCrTl3R5AQ

Fertility Training

- Why can't we train the fertilities end-to-end?
  - Cannot flow gradients
  - Need separate supervision
- Loss function



$$\mathcal{L}_{\text{ML}} = \log p_{\mathcal{NA}}(Y|X;\theta) = \log \sum_{f_{1:T'}\in\mathcal{F}} p_F(f_{1:T'}|x_{1:T'};\theta)\cdot p(y_{1:T}|x_{1:T'},f_{1:T'};\theta)$$

$$\geq \mathop{\mathbb{E}}_{f_{1:T'}\sim q}\left(\underbrace{\sum_{t=1}^{T}\log p(y_t|x_1\{f_1\},..,x_{T'}\{f_{T'}\};\theta)}_{\text{Translation Loss}} + \underbrace{\sum_{t'=1}^{T'}\log p_F(f_{t'}|x_{1:T'};\theta)}_{\text{Fertility Loss}}\right) + \mathcal{H}(q)$$

IBM Model 2

- Idea is to use **alignments** from SMT model
  - Easy to translate into fertilities

$l = 6,\ m = 7$

e = And the program has been implemented

f = Le programme a ete mis en application

- One alignment is

$$\{2, 3, 4, 5, 6, 6, 6\}$$

$$\boxed{p(a|e, m) = \prod_{j=1}^{m} q(a_j|j, l, m)} \longrightarrow \boxed{p(f, a|e, m) = \prod_{j=1}^{m} q(a_j|j, l, m)t(f_j|e_{a_j})}$$

- Trained with expectation maximization (EM) on data
  - Allows model to learn alignments that are not observed in the data

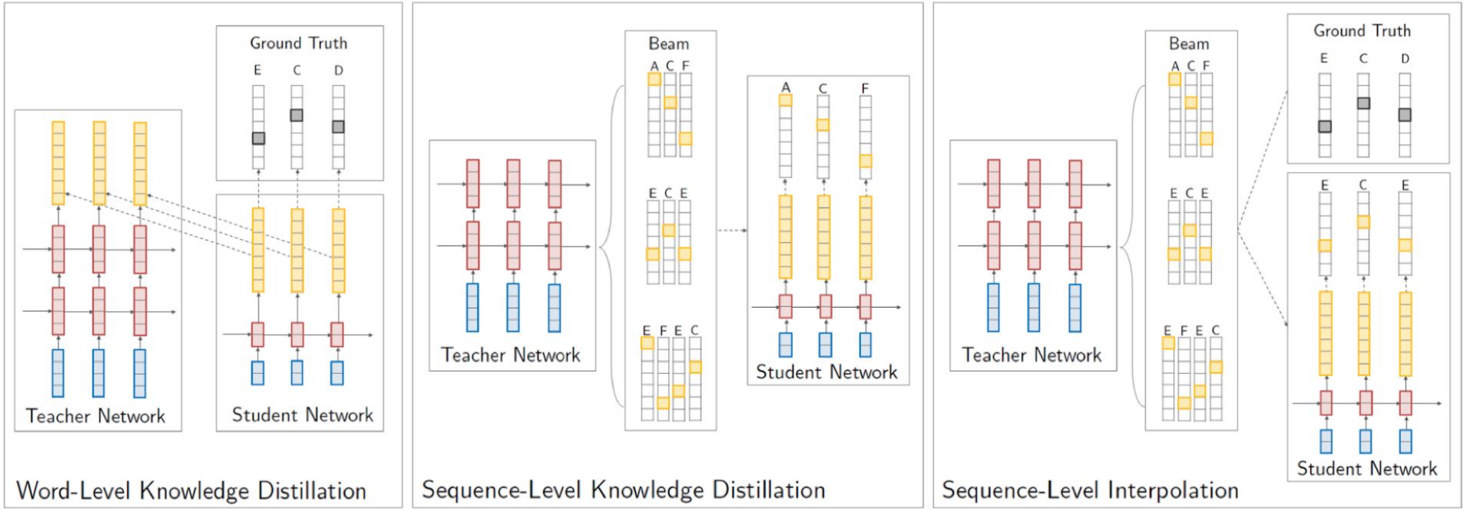
Is this a good latent variable?

- Their criteria
  - Easy to infer from training data
  - Should account for correlations across time (so each output is almost conditionally independent)
  - Should not convey too much information about target translation so that decoder still has something to learn

> "Including both fertilities and reordering in the latent variable would provide complete alignment statistics. This would make the decoding function trivially easy to approximate given the latent variable and force all of the modeling complexity into the encoder. Using fertilities alone allows the decoder to take some of this burden off of the encoder."

Knowledge Distillation

Idea (Hinton et Al., 2015, Kim and Rush, 2016): **distilling the knowledge of a teacher model** (e.g. larger model or ensemble of models) **into a student model**.



Upside: can train on less data while still learning how to generalise well, much faster during inference, outputs are less noisy… **Downside**: output of the teacher model are lower in quality than original data.

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Fine Tuning

Problem: our training is **not end-to-end** (translation model and fertilities predictor are trained separately).

→ after training the Non-AR Transformer to convergence: add a fine-tuning step

losses added with fine-tuning

original loss (student model from Knowledge distillation)

$$\mathcal{L}_{\text{FT}} = \lambda \left(\underbrace{\mathop{\mathbb{E}}_{f_{1:T'} \sim p_F} \left(\mathcal{L}_{\text{RKL}}\left(f_{1:T'}\right) - \mathcal{L}_{\text{RKL}}\left(\bar{f}_{1:T'}\right) \right)}_{\mathcal{L}_{\text{RL}}} + \underbrace{\mathop{\mathbb{E}}_{f_{1:T'} \sim q} \left(\mathcal{L}_{\text{RKL}}\left(f_{1:T'}\right) \right)}_{\mathcal{L}_{\text{BP}}} \right) + (1-\lambda)\mathcal{L}_{\text{KD}}$$

expectation over **fertility distribution** from ***translation model*** (p_F), normalised

→ trained with **Reinforcement Learning**

expectation over **fertility distribution** from ***external fertility inference model*** (q)

→ trained with **Backprop**

where $\mathcal{L}_{\text{RKL}}$ is obtained from word-knowledge distillation based on KL divergence with the teacher model:

$$\mathcal{L}_{\text{RKL}}\left(f_{1:T'};\theta\right) = \sum_{t=1}^{T} \sum_{y_t} \left[\log p_{\mathcal{AR}}\left(y_t | \hat{y}_{1:t-1}, x_{1:T'}\right) \cdot p_{\mathcal{NA}}\left(y_t | x_{1:T'}, f_{1:T'};\theta\right)\right]$$

Decoding Process

- Argmax decoding

$$\hat{Y}_{\text{argmax}} = G(x_{1:T'}, \hat{f}_{1:T'}; \theta), \text{ where } \hat{f}_{t'} = \underset{f}{\operatorname{argmax}}\, p_F(f_{t'} | x_{1:T'}; \theta)$$

- Average decoding

$$\hat{Y}_{\text{average}} = G(x_{1:T'}, \hat{f}_{1:T'}; \theta), \text{ where } \hat{f}_{t'} = \text{Round}\left(\sum_{f_{t'}=1}^{L} p_F(f_{t'} | x_{1:T'}; \theta) f_{t'} \right)$$

- Noisy parallel decoding (NPD)

$$\hat{Y}_{\text{NPD}} = G(x_{1:T'}, \underset{f_{t'} \sim p_F}{\operatorname{argmax}}\, p_{\mathcal{AR}}(G(x_{1:T'}, f_{1:T'}; \theta) | X; \theta); \theta)$$

Noisy Parallel Decoding

- Non-autoregressive model can leverage autoregressive teacher during inference as well

- "Autoregressive" teacher can run very fast while evaluating candidate translation
  - Does not have to consume previous input
  - Operates off of candidate translation tokens (like teacher forcing during training)

- Sample size trades off speed and accuracy
  - We will see this in a moment

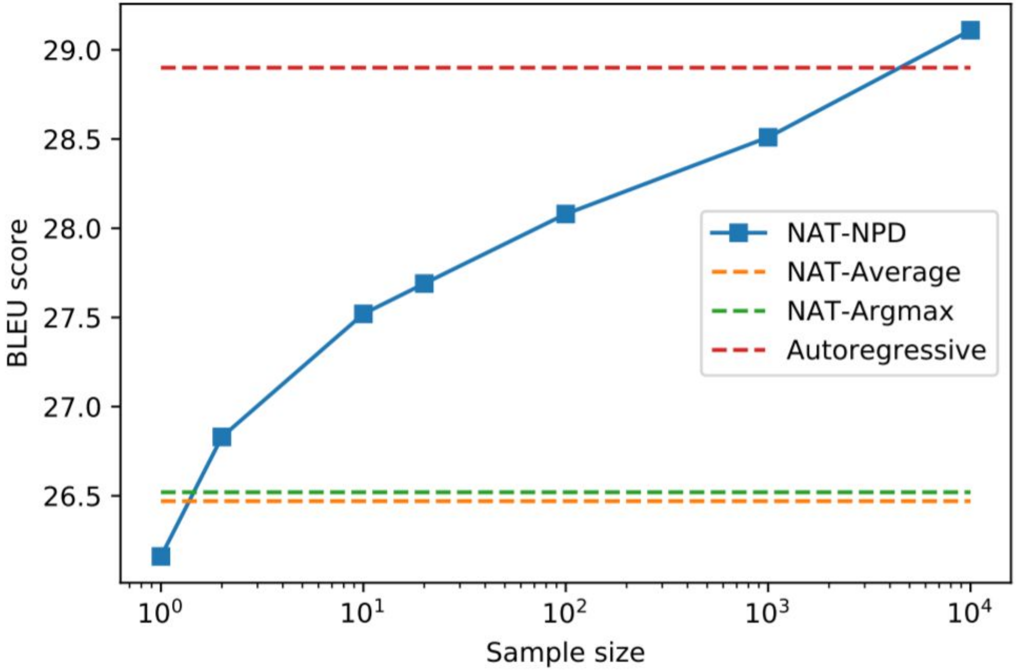
Example of NPD

se lucreaza la solutii de genul acesta .

Decoder input (copied by fertilities)	Decoder output
se la solutii de genul acesta .	solutions on this kind are done .
se lucreaza la solutii de acesta .	work done on solutions like this .
se lucreaza solutii de genul acesta .	solutions on this kind is done .
se se lucreaza la solutii de acesta .	work is done on solutions like this .
se lucreaza lucreaza la solutii de acesta .	work is done on solutions like this .
se se lucreaza lucreaza la solutii de acesta .	**work is being done on solutions like this .** AR favorite
se se lucreaza lucreaza la solutii de de acesta .	work is being done on solutions such as this .
se se lucreaza lucreaza la solutii de genul acesta .	work is being done on solutions such this kind .

Comparison of Decoding Methods

- So is NAT-NPD better than autoregressive?
- Could this model ever even theoretically outperform autoregressive models?



Results

Bottom line: sometimes very competitive BLEU with significant speedup

Models	WMT14		WMT16		IWSLT16		
	En→De	De→En	En→Ro	Ro→En	En→De	Latency / Speedup	
NAT	17.35	20.62	26.22	27.83	25.20	39 ms	15.6×
NAT (+FT)	17.69	21.47	27.29	29.06	26.52	39 ms	15.6×
NAT (+FT + NPD $s = 10$)	18.66	22.41	29.02	30.76	27.44	79 ms	7.68×
NAT (+FT + NPD $s = 100$)	19.17	23.20	29.79	**31.44**	28.16	257 ms	2.36×
Autoregressive ($b = 1$)	22.71	26.39	31.35	31.03	28.89	408 ms	1.49×
Autoregressive ($b = 4$)	23.45	27.02	31.91	31.76	29.70	607 ms	1.00×

Ablation Study

guess what this means?

Distillation b=1	Distillation b=4	Decoder Inputs +uniform	Decoder Inputs +fertility	+PosAtt	Fine-tuning $+\mathcal{L}_{KD}$	Fine-tuning $+\mathcal{L}_{BP}$	Fine-tuning $+\mathcal{L}_{RL}$	BLEU	BLEU (T)
				✓				≈ 2	
		✓		✓				16.51	
			✓	✓				18.87	
✓		✓		✓				20.72	
	✓	✓		✓				21.12	
✓			✓					24.02	43.91
✓			✓	✓				25.20	45.41
✓		✓		✓	✓	✓		22.44	
✓			✓	✓			✓	×	×
✓			✓	✓		✓		×	×
✓			✓	✓	✓	✓		25.76	46.11
✓			✓	✓	✓	✓	✓	**26.52**	**47.38**

Really need to copy source

Teacher distillation is very helpful

External fertility actually contributes a lot

Fine-tuning process gives another percent

Non-Autoregressive Neural Machine Translation (Gu et Al., 2018)

Quiz time

Can you think of any limitations of the proposed approach in (Gu et al, 2018)?

Quiz time

Can you think of any limitations of the proposed approach in (Gu et al, 2018)?

- **Non-differentiable component when copying fertilities into the decoder** → model cannot be trained-end-to-end (and fine-tuning leads to only negligible improvements).
- **Still relies on autoregressive to train the teacher model** → does not beat AR "on its own".
- **Very slow at training** → it has to train both the teacher (larger model = computationally expensive) and the student
- **Fertilities do no cope with the problem of re-ordering of words in the translated sentence** → they are just an alignment between number of words in the input and number of words in the output.

Bonus Paper!

Ghazvininejad M., Levy O., Liu Y., Zettlemoyer L. (2019)

Bonus Paper

Mask-Predict: Parallel Decoding of Conditional Masked Language Models

Marjan Ghazvininejad* **Omer Levy*** **Yinhan Liu*** **Luke Zettlemoyer**
Facebook AI Research
Seattle, WA

Pros: *Newer* *Better* *MUCH MUCH MUCH MUCH Prettier*

Cutting to the chase

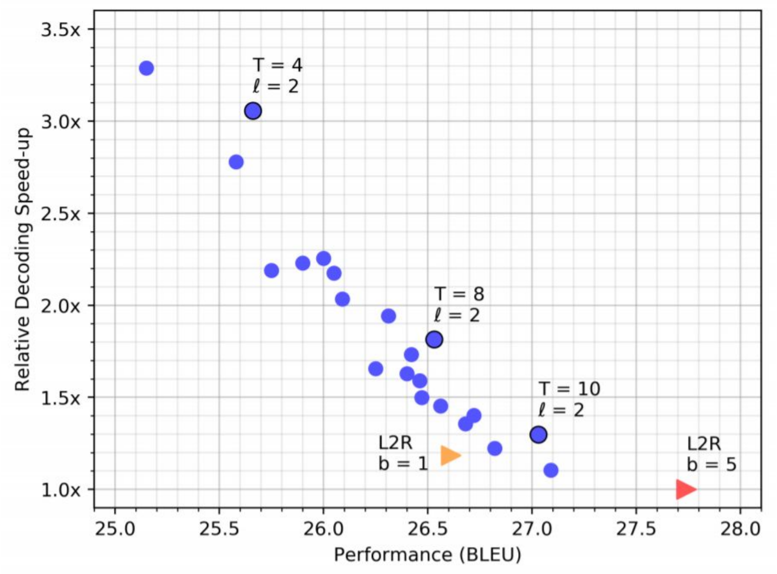
- Train to predict **masked** target tokens given source sequence and unmasked target tokens.
- Encoder also predicts **length** of target sequence based on source.

src	**Je suis étudiant**	
	[M] [M] [M] [M]	predicted L=4
$t = 0$	**I I am studying**	generated all masked tokens
$t = 1$	**I am am student**	replaced least certain tokens
$t = 2$	**I am a student**	arrived at final translation

src	Der Abzug der franzsischen Kampftruppen wurde am 20. November abgeschlossen .
$t = 0$	The departure of the French combat completed completed on 20 November .
$t = 1$	The departure of French combat troops was completed on 20 November .
$t = 2$	The withdrawal of French combat troops was completed on November 20th .

Conclusions

- Outperforms other parallel decoding schemes
- Linear-ish trade-off between speed and performance
- Still heavily reliant on knowledge distillation



Thank you

Any questions?