Sequence Generation Tasks

Machine Learning for Natural Language Processing, ENSAE 2022

Lecture 6

Benjamin Muller, INRIA Paris

Lectures Outline

1. The Basics of Natural Language Processing (February 1st)
2. Representing Text with Vectors (February 1st)
3. Deep Learning Methods for NLP (February 8th)
4. Language Modeling (February 8th)
5. Sequence Labelling (Sequence Classification) (February 15th)
6. **Sequence Generation Tasks (February 15th)**

Framework

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$

a target sequence of tokens $(y_1, .., y_{T'}) \in {V'}^{T'}$.

Framework

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$

a target sequence of tokens $(y_1, .., y_{T'}) \in V'^{T'}$.

Framework

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$

a target sequence of tokens $(y_1, .., y_{T'}) \in {V'}^{T'}$.

Our goal is to estimate:

$$p_\theta(y_1, .., y_{T'} | x_1, .., x_T)$$

Framework

We assume an input sequence of tokens $(x_1, .., x_T) \in V^T$

a target sequence of tokens $(y_1, .., y_{T'}) \in {V'}^{T'}$.

Our goal is to estimate:

$$p_\theta(y_1, .., y_{T'} | x_1, .., x_T)$$

We frame it as a classification task:

$$\hat{y}_t = argmax_{y \in V'} \, p_\theta(y | (x_1, .., x_T), (y_t, .., y_{t-1})) \;\; \forall \, t \in [|1, T'|]$$

Sequence Generation Tasks

- Machine Translation
- Summarization
- Question Answering

Machine Translation

Given a sequence in one language → translate it in another language

Cats eat mice* → *Les chats mangent les souris

A lot of parrallel data for this task *from* and *to* English

Harder to Translate ***Hawaiian*** to ***Swiss German*** than ***English*** **to** ***French***

Machine Translation

Evaluating Sequence Generation Tasks Automatically is Challenging

Two Translations can be as good as each other

BLEU SCORE:

- **N-Gram-Based Evaluation metric**
- It measures how much of the n-gram **in the prediction appear in the reference translation**
- **1** means the prediction is the same as the gold translation
- **0** means there is no n-gram in common

Summarization

Source Text
An officer , responding to reports of a suspicious person , shot and killed an unarmed man who was running around in a metro atlanta apartment complex naked . the officer fired two shots when the man charged at him , said cedric alexander , the public safety director of dekalb county . but given that the man was not carrying a weapon , the police department immediately turned over the case to the georgia bureau of investigations for an independent probe . `` what i have requested here. A result of what 's going on currently across this country as it relates to police shootings , " alexander told reporters . the officer was white ; the deceased man was african-american , alexander said . the incident took place monday afternoon at an apartment complex in chamblee , a suburb of atlanta . someone called 911 to report a man `` acting deranged , knocking on doors and crawling around naked , " alexander said . when the officer arrived , the man charged at him , alexander said . `` the officer called him to stop while stepping backward , drew his weapon and fired two shots , " he said . the man , struck twice in the upper body , died .
Reference Summary
Police : officer fired two shots when the man charged at him .The case was immediately turned over to the gbi .

Summarization Evaluation

Evaluating Summarization models is challenging

A Standard Metric is the ROUGE score

It measures how much the words (and/or n-grams) in the human reference summaries appeared **in the machine generated summaries.**

Design Questions

★ **What output activation function and loss?**

★ **What architecture?**

★ **How do you represent a token to feed the model?**

NB: Questions to ask for all NLP tasks modeled with Deep Learning

Design Questions

★ **What output activation function and loss?**

Softmax and Cross Entropy

★ **What architecture?**

★ **How do you represent a token to feed the model?**

What Architecture?

$$p_\theta(y_t|(x_1,..,x_T),(y_t,..,y_{t-1}))$$

We want to model an output sequence conditioned on an input *sequence*

With deep learning, we do that with: a encoder-decoder model

NB: also called “sequence to sequence” or “*seq2seq*”

The Encoder-Decoder Paradigm

$$p_\theta(y_t|(x_1,..,x_T),(y_t,..,y_{t-1}))$$

Intuition:

- **We know how to model a single sequence at a time with a DL model**
  E.g. with a LSTM or a Transformer
- **Here we want to model two sequences together**
  One conditioned on the other

The Encoder-Decoder Paradigm

$$p_\theta(y_t|(x_1,..,x_T),(y_t,..,y_{t-1}))$$

Intuition:

- **We know how to model a single sequence at a time with a DL model**
  E.g. with a LSTM or a Transformer
- **Here we want to model two sequences together**
  One conditioned on the other

➔ **Combine two Deep Learning Architectures together**

The Encoder-Decoder Paradigm

$$p_\theta(y_t|(x_1,..,x_T),(y_t,..,y_{t-1}))$$

Encode **input sequence**

$$enc_{\theta_e}: \quad V^T \rightarrow \mathbb{R}^T$$

$$(x_1,..,x_T) \mapsto (h_1,...,h_T)$$

The Encoder-Decoder Paradigm

$$p_\theta(y_t|(x_1,..,x_T),(y_t,..,y_{t-1}))$$

Encode **input sequence**

$$enc_{\theta_e}: \quad V^T \rightarrow \mathbb{R}^T$$

$$(x_1,..,x_T) \mapsto (h_1,...,h_T)$$

Decode **target sequence** given hidden states of the encoder

$$dec_{\theta_d}: \quad \mathbb{R}^T \times V'^t \rightarrow [0,1]^V$$

$$((h_1,..,h_T),(y_1,..,y_t)) \mapsto \hat{p}$$

The Encoder-Decoder Paradigm

How to integrate (h1,..hT) in the *decoder ?*

➔ **It depends what architecture is chosen for the *encoder* and the *decoder***

The Encoder-Decoder Paradigm

How to integrate (h1,..hT) in the *decoder ?*

➔ **It depends what architecture is chosen for the *encoder* and the *decoder***

- **RNN encoder-decoder** (possibly with an Attention Mechanism)
- **Transformer Model**

The Encoder-Decoder with RNNs

Recall: Vanilla RNN with *L'* and time step *sequence of length T'*

$$h_{i+1,t+1} = \varphi_i(W_i h_{i,t} + U_i h_{i+1,t} + b_i), \forall\, i \in [|1, L' - 1|]$$

$$\text{with } h_{1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{L',t+1} \quad \forall\, t \in [|1, T' - 1|]$$

$$\text{with } \varphi_{L'} = softmax$$

The Encoder-Decoder with RNNs

Given $(x_1, .., x_T)^T$ and $(y_1, .., y_t) \in V'^{T'}$, we predict $\hat{p}_{t+1}$, distribution over V' with:

Decoder

$$h_{dec,i+1,t+1} = \varphi_i(W'_i h_{dec,i,t} + U'_i h_{dec,i+1,t} + b'_i + V_i h_{enc,L+1,T+1}), \forall\, i \in [|1, L'|]$$

$$\text{with } h_{dec,1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{dec,L'+1,t+1} \quad \forall\, t \in [|1, T'|]$$

$$\text{with } \varphi_{L'} = softmax$$

The Encoder-Decoder with RNNs

Given $(x_1, .., x_T)^T$ and $(y_1, .., y_t) \in V'^{T'}$, we predict $\hat{p}_{t+1}$, distribution over V' with:

Decoder

$$h_{dec,i+1,t+1} = \varphi_i(W'_i h_{dec,i,t} + U'_i h_{dec,i+1,t} + b'_i + V_i h_{enc,L+1,T+1}), \forall\, i \in [|1, L'|]$$

$$\text{with } h_{dec,1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{dec,L'+1,t+1} \quad \forall\, t \in [|1, T'|]$$

$$\text{with } \varphi_{L'} = softmax$$

- Decoder of *with **L' layer**, with parameters **W'i, U'i, b', Vi for all i***
- It decodes sequentially the target sequence

The Encoder-Decoder with RNNs

Given $(x_1, .., x_T)^T$ and $(y_1, .., y_t) \in V'^{T'}$, we predict $\hat{p}_{t+1}$, distribution over V' with:

Decoder

$$h_{dec,i+1,t+1} = \varphi_i(W'_i h_{dec,i,t} + U'_i h_{dec,i+1,t} + b'_i + V_i h_{enc,L+1,T+1}), \forall\, i \in [|1, L'|]$$

$$\text{with } h_{dec,1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{dec,L'+1,t+1} \quad \forall\, t \in [|1, T'|]$$

$$\text{with } \varphi_{L'} = softmax$$

- Decoder of *with **L' layer**, with parameters **W'i, U'i, b', Vi for all i***
- It decodes sequentially the target sequence
- It is conditioned on **the input sequence through the encoding output**

The Encoder-Decoder with RNNs

Given $(x_1, .., x_T)^T$ and $(y_1, .., y_t) \in V'^{T'}$, we predict $\hat{p}_{t+1}$, distribution over V' with:

Decoder

$$h_{dec,i+1,t+1} = \varphi_i(W'_i h_{dec,i,t} + U'_i h_{dec,i+1,t} + b'_i + V_i h_{enc,L+1,T+1}), \forall\, i \in [|1, L'|]$$

$$\text{with } h_{dec,1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{dec,L'+1,t+1} \quad \forall\, t \in [|1, T'|]$$

$$\text{with } \varphi_{L'} = softmax$$

Encoder: also a RNN that encodes the input sequence in a fixed vector

$$h_{enc,i+1,t+1} = \varphi_i(W_i h_{enc,i,t} + U_{enc,i} h_{i+1,t} + b_{enc,i})$$

$$\forall\, i \in [|1, L|] \ \forall\, t \in [|1, T|] \quad \text{with } h_{enc,1,t} = Emb(x_t)$$

The Encoder-Decoder with RNNs

Given $(x_1, .., x_T)^T$ and $(y_1, .., y_t) \in V'^{T'}$, we predict $\hat{p}_{t+1}$, distribution over V' with:

Decoder

$$h_{dec,i+1,t+1} = \varphi_i(W'_i h_{dec,i,t} + U'_i h_{dec,i+1,t} + b'_i + V_i h_{enc,L+1,T+1}), \forall\, i \in [|1, L'|]$$

$$\text{with } h_{dec,1,t} = Emb(y_t) \text{ and } \hat{p_{t+1}} = h_{dec,L'+1,t+1} \quad \forall\, t \in [|1, T'|]$$

$$\text{with } \varphi_{L'} = softmax$$

Encoder: also a RNN that encodes the input sequence in a fixed vector

$$h_{enc,i+1,t+1} = \varphi_i(W_i h_{enc,i,t} + U_{enc,i} h_{i+1,t} + b_{enc,i})$$

$$\forall\, i \in [|1, L|] \ \forall\, t \in [|1, T|] \quad \text{with } h_{enc,1,t} = Emb(x_t)$$

The Encoder-Decoder with RNNs

Simple RNN-based Encoder-Decoder Model:

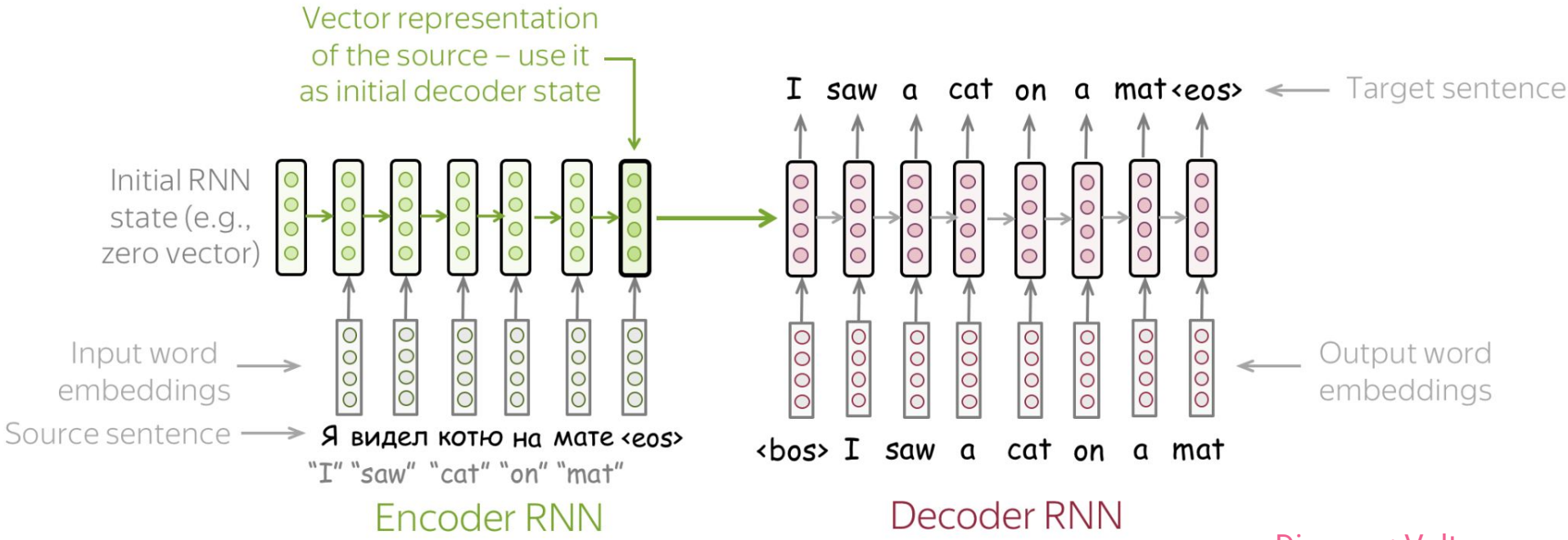


Diagram: Volta

The Encoder-Decoder Training

With Backpropagation

1. We feed the model with both the input and output sequence
2. We compute the loss based on the “gold“ output sequence
3. We update all the parameters of the model with backpropagation

Diagram: Volta

The Encoder-Decoder with RNNs: Limits

Limits: ***At test time in the encoder-decoder that we introduced***
The input sequence has a fixed representation ($h_{enc,L+1,T+1}$ is fixed)

The Encoder-Decoder with RNNs: Limits

Limits: ***At test time in the encoder-decoder that we introduced***
The input sequence has a fixed representation ($h_{enc,L+1,T+1}$ is fixed)

Example:

Je vois un chat sur un matelas ⇒ ***I see a cat on a mat***

The Encoder-Decoder with RNNs: Limits

Limits: ***At test time in the encoder-decoder that we introduced***
The input sequence has a fixed representation ($h_{enc,L+1,T+1}$ is fixed)

Example:

Je vois un chat sur un matelas ⇒ ***I see a cat on a mat***

Step 4:
Given: ***Je vois un chat sur un matelas*** ⇒ ***cat***
Step 7:
Given: ***Je vois un chat sur un matelas*** ⇒ ***mat***

The Encoder-Decoder with RNNs: Limits

Limits: ***At test time in the encoder-decoder that we introduced***
The input sequence has a fixed representation ($h_{enc,L+1,T+1}$ is fixed)

Example:

Je vois un chat sur un matelas ⇒ ***I see a cat on a mat***

Step 4:
Given: ***Je vois un chat sur un matelas*** ⇒ ***cat***
Step 7:
Given: ***Je vois un chat sur un matelas*** ⇒ ***mat***

➔ **We need "decoding-dependant" representation of the input sequence**

How to build more flexible encoder-decoder?

Solution 1:

- **Integrate an Attention Mechanism in a RNN-based encoder-decoder**

Solution 2:

- **Use an Encoder-Decoder Transformer**

NB: We detail only the latter solution

The Transformer Encoder-Decoder

Recall: A Transformer encoder is a stack of

- **Self-Attention Layer**
  that builds a contextual representation of the input sequence
- **Feed-Forward Layer**

Recall: Self-Attention Layers

Given a sequence of input vectors $X = (x_1, .., x_T) \in \mathbb{R}^\delta$ (noted $H = (h_{0,1}, .., h_{0,T})$).

We build 3 new vectorial representation of our sequence $H = (h_1, .., h_T)$).

The *query* $Q = (q_1, .., q_T)$, the *key* $K = (k_1, .., k_T)$ and the *value* $V = (v_1, .., v_T)$ vectors.

$$\tilde{H} = softmax(\frac{Q\,K^T}{\sqrt{\delta_K}})V$$

i.e. $\tilde{h_t} = softmax(\frac{q_t\,K^T}{\sqrt{\delta_K}}).V = \sum_{t'} s_{t'}\,v_{t'}$ with $s_{t'} = \frac{e^{q_{t'}k_t}}{\sum_t e^{q_{t'}k_t}}$

The Transformer Encoder-Decoder

A Transformer Encoder-Decoder is:

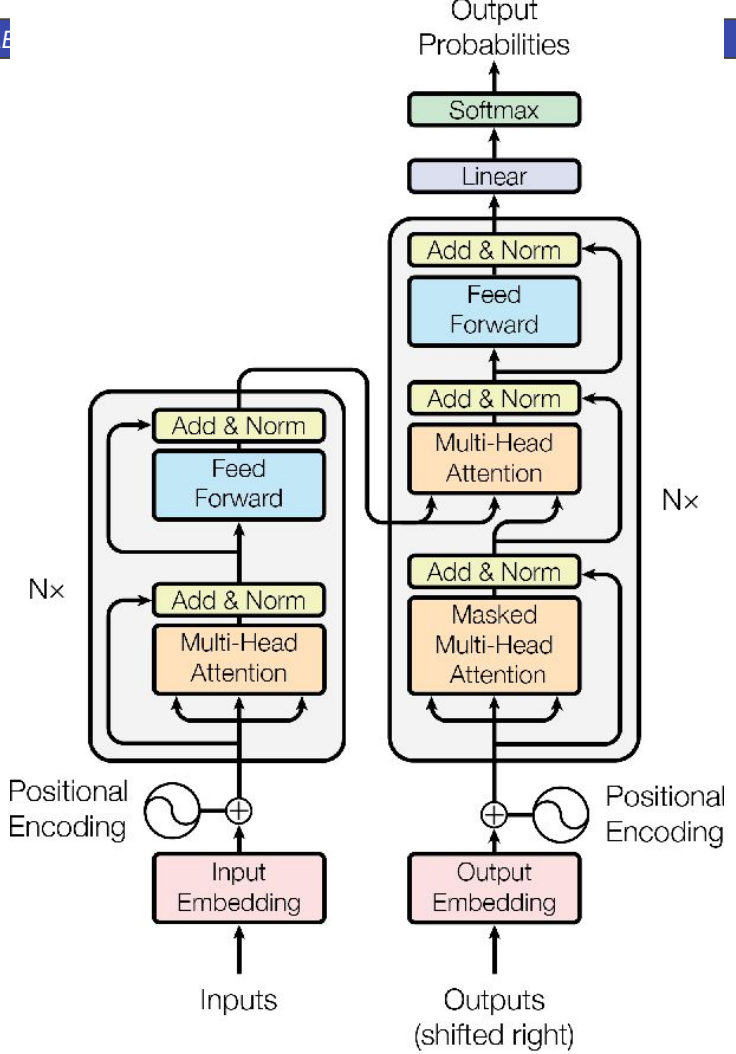
- A Transformer Encoder
- Plugged to a Transformer Decoder

The Transformer Encoder-Decoder

A Transformer Encoder-Decoder is:

- A Transformer Encoder
- Plugged to a Transformer Decoder
- The connection between both being made *with the self-attention layers*

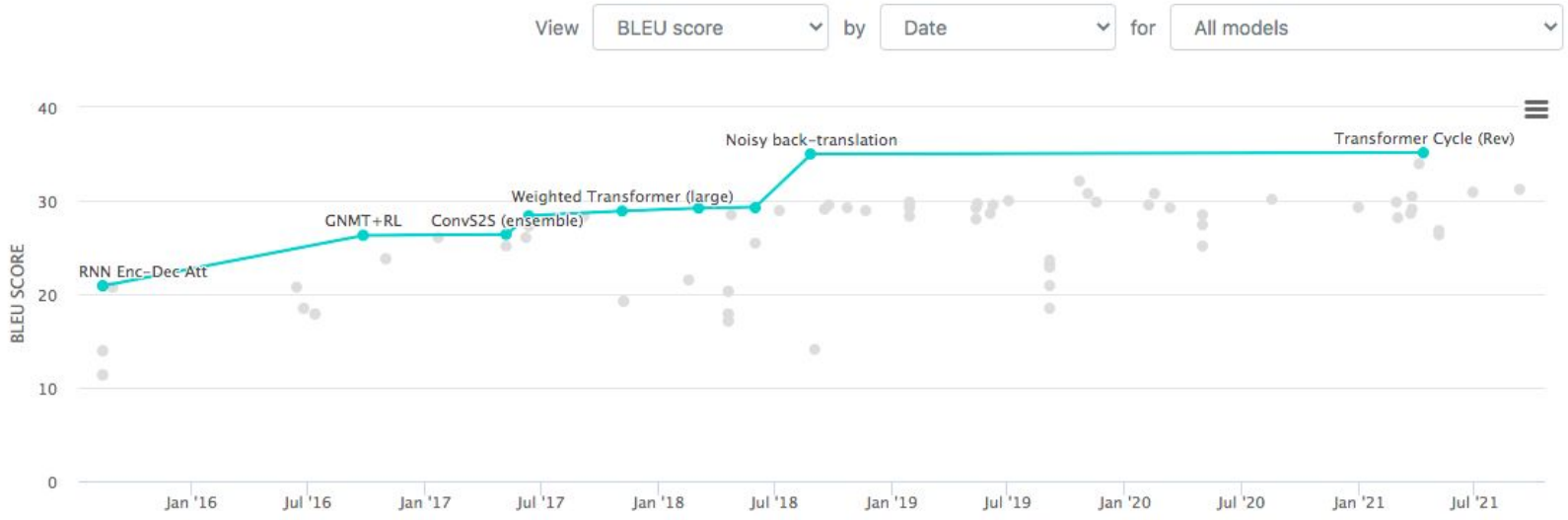
Transformer Encoder-Decoder



Vaswani et. al

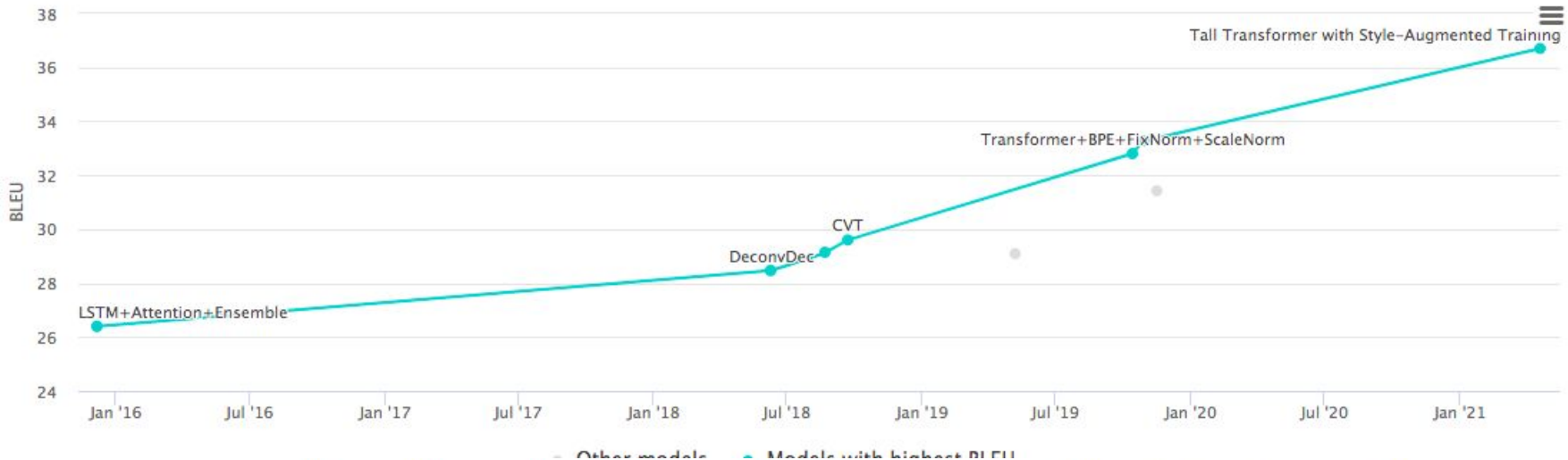
Performance in Machine Translation

English To German



Performance in Machine Translation

English To Vietnamese



Extension of the Encoder-Decoder Paradigm

Transfer Learning

- Similarly to **BERT**, it is possible to **pretrain an encoder-decoder** using some language-model-based objective **(cf. BART, T5)**

Tasks:

- We have seen how to use an encoder-decoder for sequence generation tasks: We can also use **encoder-decoder for sequence labeling/classification tasks (simply predict labels as tokens) (cf. T5)**

Lecture Summary

Encoder-Decoder Approach

- RNN-based
- Transformers

Perspectives

Scaling Pretraining (model size, amount of data, modalities) leads to impressive empirical progress

- Better Task-Specific Performance for most NLP tasks
- Better Task-Specific Few-Shot Performance
- Better Cross-Lingual Transfer Performance

Many Challenges Remain

- Integrate "**Knowledge Base**" with Large scale Language Models
- Potential **Harmful Uses** of those models (hateful content, biases…)
- "**Update**" Language Models efficiently
- **Speed** at test time & **Cost** of pretraining
- **Evaluating** Generative Models