Machine Translation

Module: Natural Language Processing
Date: 01.07.2021

Fernandes Elanton, Opitz Dominik

Content

Introduction

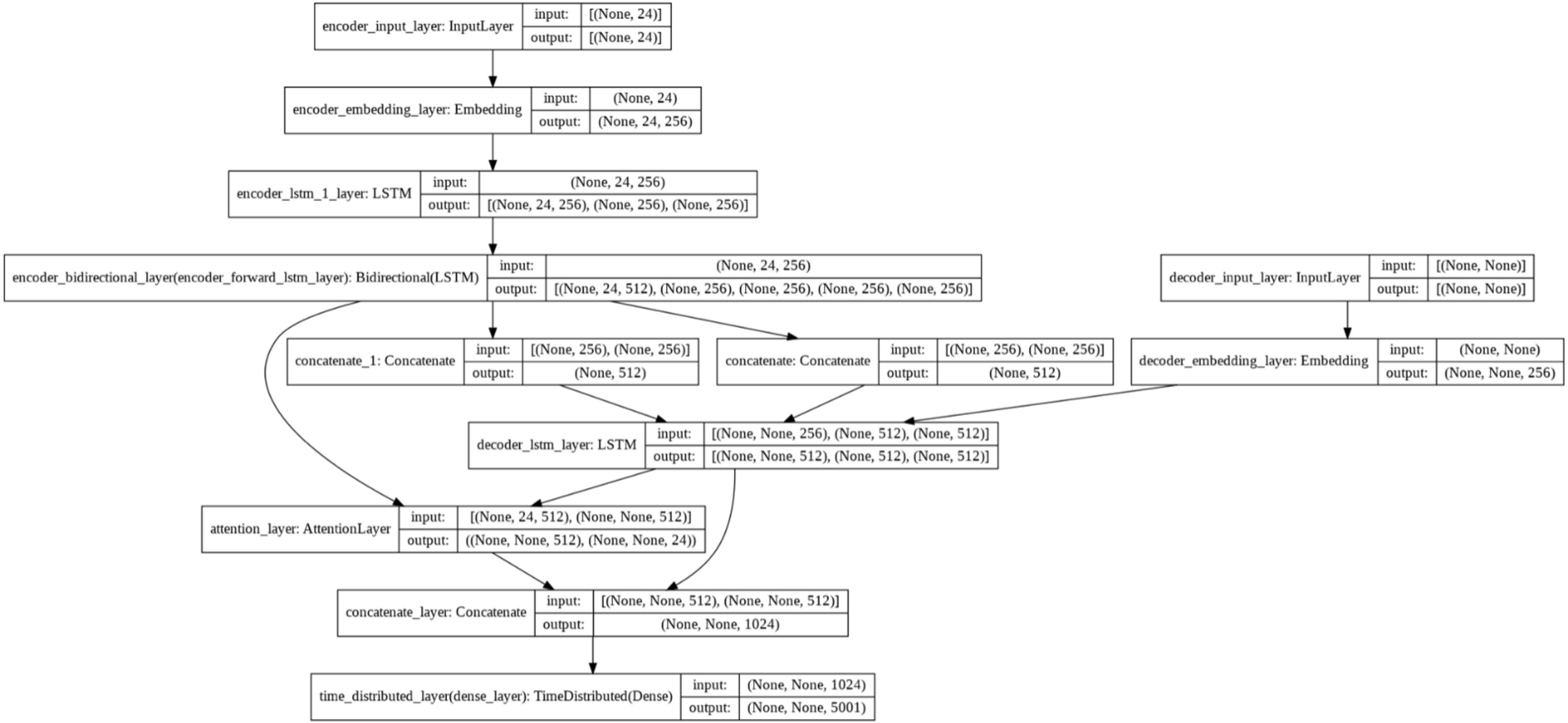
Problem:

- Translate German sentence to English sentences

Initial Proposal:

- Based on Bahdanau et al. (2015) making use of Bidirectional LSTM

Model



encoder_input_layer: InputLayer | input: [(None, 24)] | output: [(None, 24)]
encoder_embedding_layer: Embedding | input: (None, 24) | output: (None, 24, 256)
encoder_lstm_1_layer: LSTM | input: (None, 24, 256) | output: [(None, 24, 256), (None, 256), (None, 256)]
encoder_bidirectional_layer(encoder_forward_lstm_layer): Bidirectional(LSTM) | input: (None, 24, 256) | output: [(None, 24, 512), (None, 256), (None, 256), (None, 256), (None, 256)]
decoder_input_layer: InputLayer | input: [(None, None)] | output: [(None, None)]
concatenate_1: Concatenate | input: [(None, 256), (None, 256)] | output: (None, 512)
concatenate: Concatenate | input: [(None, 256), (None, 256)] | output: (None, 512)
decoder_embedding_layer: Embedding | input: (None, None) | output: (None, None, 256)
decoder_lstm_layer: LSTM | input: [(None, None, 256), (None, 512), (None, 512)] | output: [(None, None, 512), (None, 512), (None, 512)]
attention_layer: AttentionLayer | input: [(None, 24, 512), (None, None, 512)] | output: ((None, None, 512), (None, None, 24))
concatenate_layer: Concatenate | input: [(None, None, 512), (None, None, 512)] | output: (None, None, 1024)
time_distributed_layer(dense_layer): TimeDistributed(Dense) | input: (None, None, 1024) | output: (None, None, 5001)

Encoder

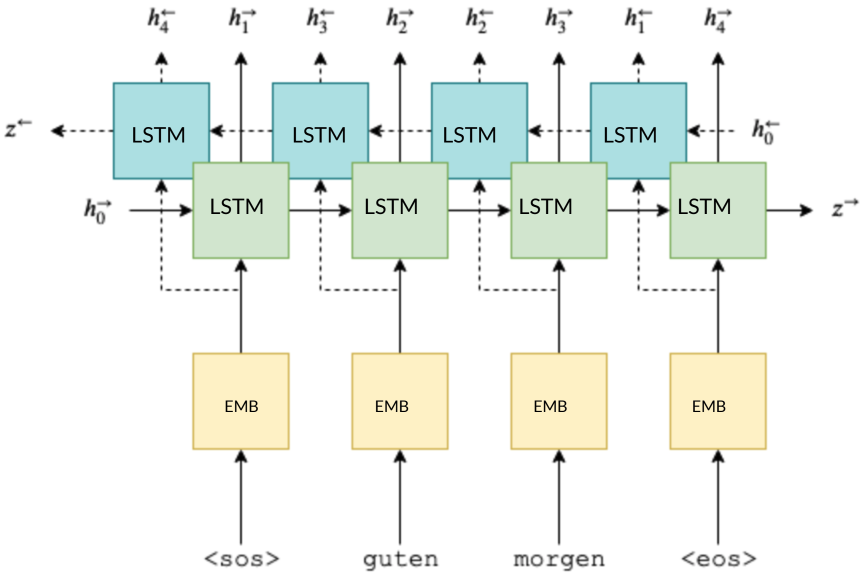
$z^{\rightarrow}$:Final Hidden Vector (forward direction)

$z^{\leftarrow}$:Final Hidden Vector (backward direction)

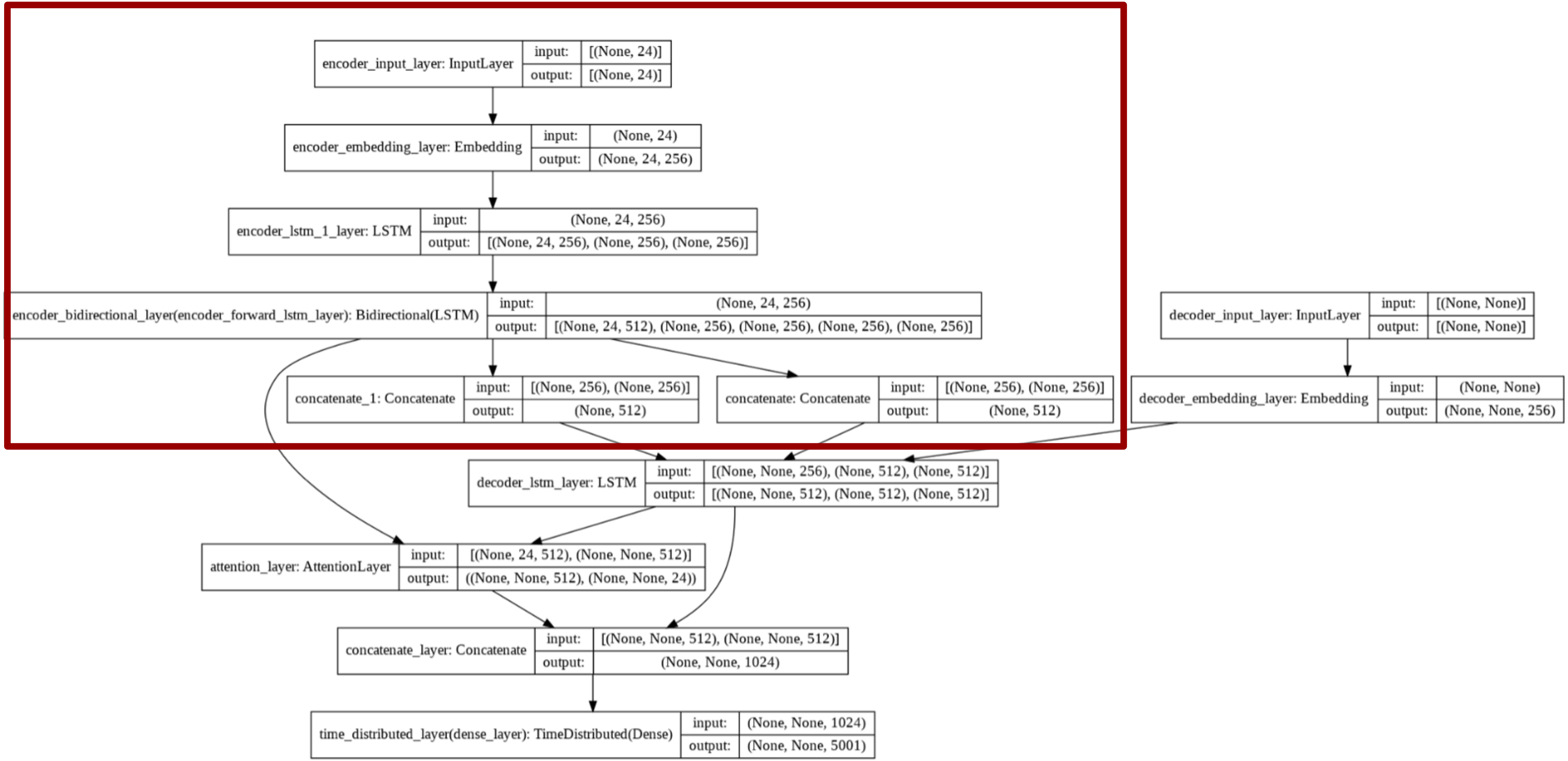
$h_n^{\rightarrow}$:Hidden vector for token n (forward)

$h_{N-n}^{\leftarrow}$:Hidden vector for token N-n (backward)

N :Total number of tokens



Source: https://github.com/bentrevett/pytorch-seq2seq/blob/master/3%20-%20Neural%20Machine%20Translation%20by%20Jointly%20Learning%20to%20Align%20and%20Translate.ipynb



encoder_input_layer: InputLayer
input: [(None, 24)]
output: [(None, 24)]
encoder_embedding_layer: Embedding
input: (None, 24)
output: (None, 24, 256)
encoder_lstm_1_layer: LSTM
input: (None, 24, 256)
output: [(None, 24, 256), (None, 256), (None, 256)]
encoder_bidirectional_layer(encoder_forward_lstm_layer): Bidirectional(LSTM)
input: (None, 24, 256)
output: [(None, 24, 512), (None, 256), (None, 256), (None, 256), (None, 256)]
decoder_input_layer: InputLayer
input: [(None, None)]
output: [(None, None)]
concatenate_1: Concatenate
input: [(None, 256), (None, 256)]
output: (None, 512)
concatenate: Concatenate
input: [(None, 256), (None, 256)]
output: (None, 512)
decoder_embedding_layer: Embedding
input: (None, None)
output: (None, None, 256)
decoder_lstm_layer: LSTM
input: [(None, None, 256), (None, 512), (None, 512)]
output: [(None, None, 512), (None, 512), (None, 512)]
attention_layer: AttentionLayer
input: [(None, 24, 512), (None, None, 512)]
output: ((None, None, 512), (None, None, 24))
concatenate_layer: Concatenate
input: [(None, None, 512), (None, None, 512)]
output: (None, None, 1024)
time_distributed_layer(dense_layer): TimeDistributed(Dense)
input: (None, None, 1024)
output: (None, None, 5001)

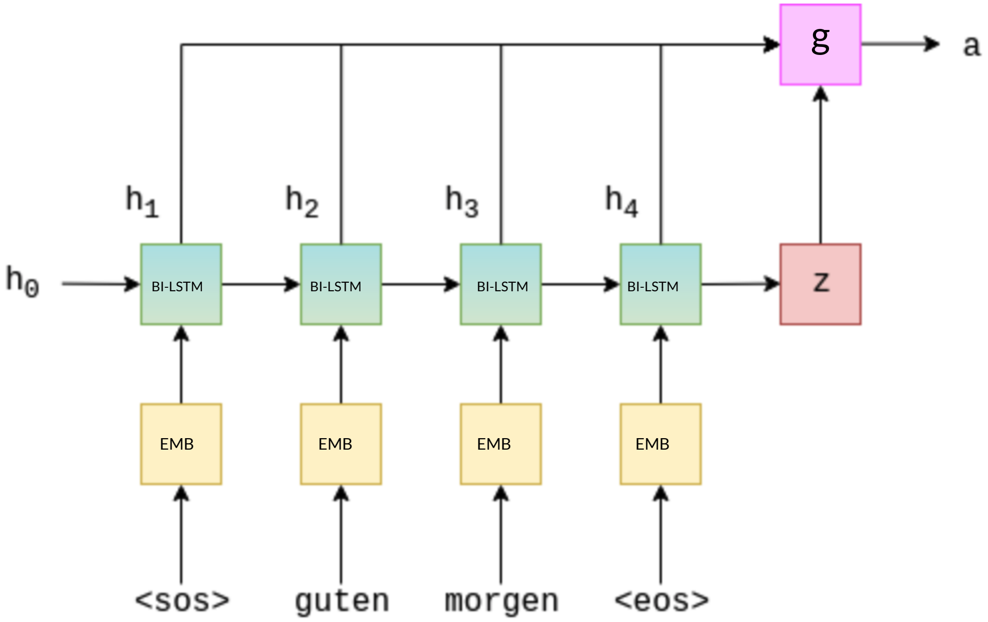
Attention

$$h_n = [h_n^{\rightarrow}; h_{N-n}^{\leftarrow}]$$

h_n: Concatenated hidden state

g: 2 input neural network

a : Attention weight

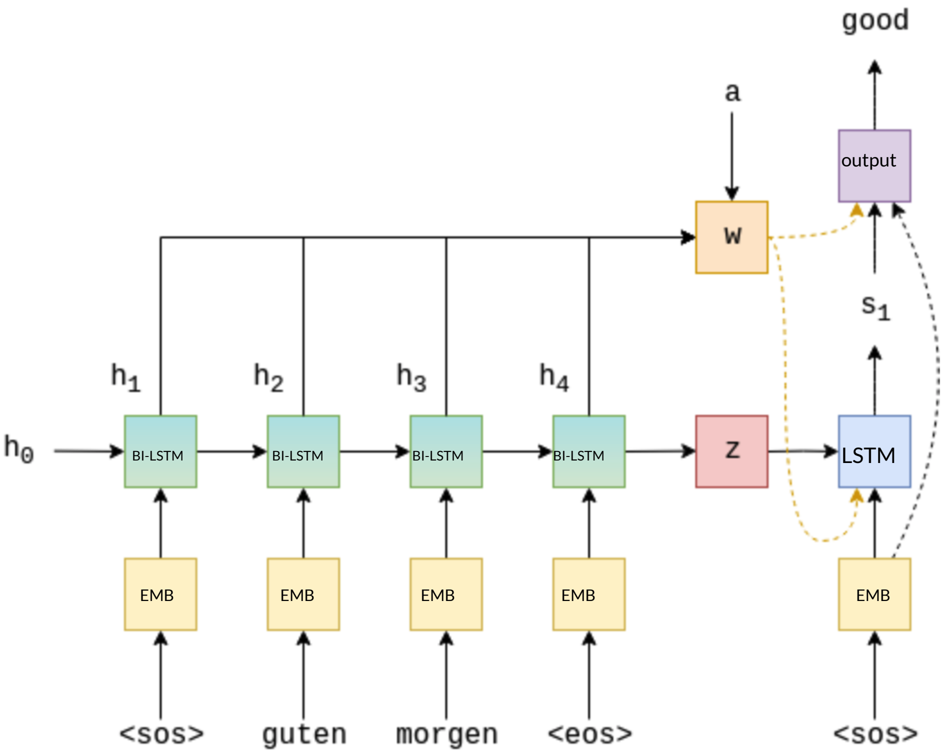


Source: https://github.com/bentrevett/pytorch-seq2seq/blob/master/3%20-%20Neural%20Machine%20Translation%20by%20Jointly%20Learning%20to%20Align%20and%20Translate.ipynb

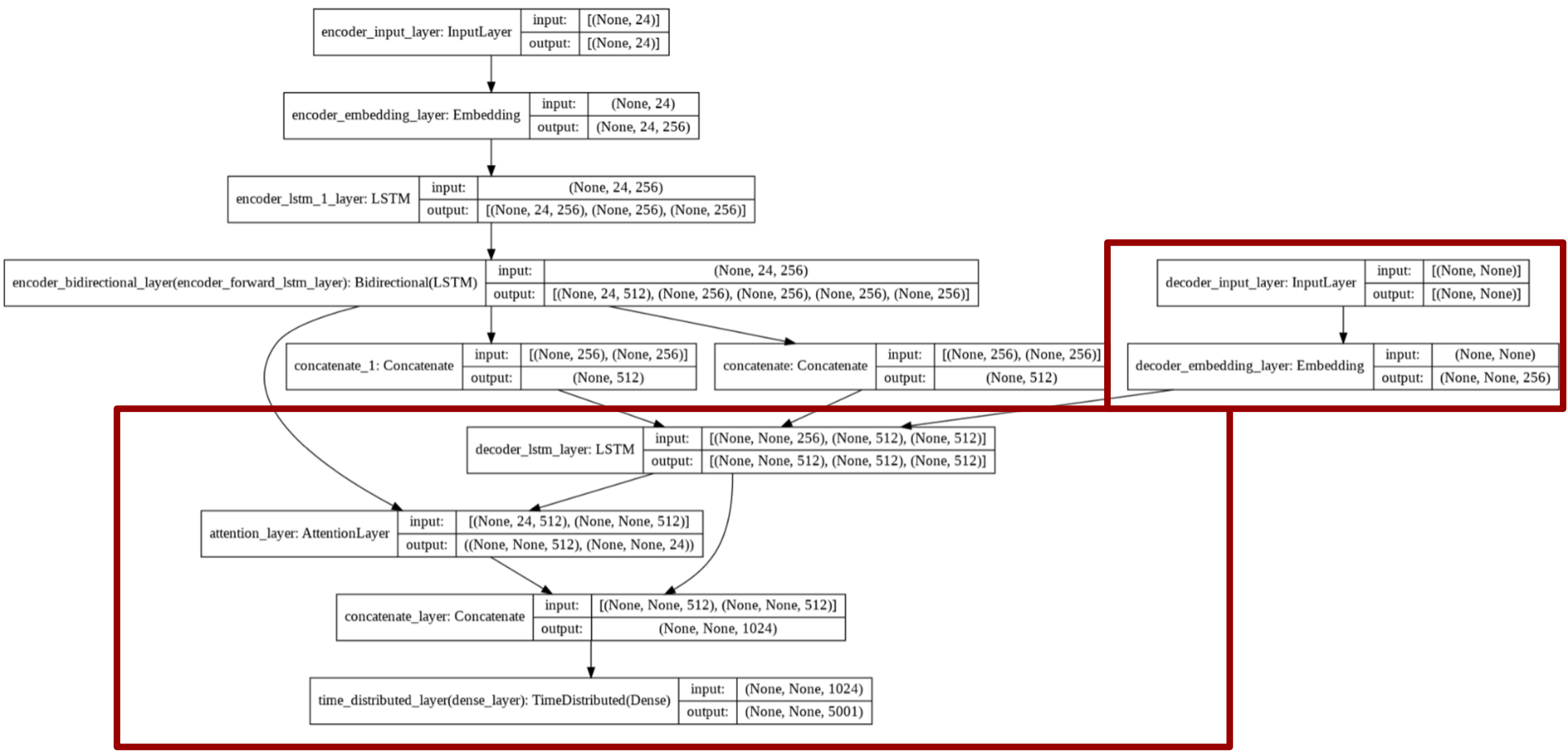
Decoder

W: Context vector

S_n: Decoder hidden state



Source: https://github.com/bentrevett/pytorch-seq2seq/blob/master/3%20-%20Neural%20Machine%20Translation%20by%20Jointly%20Learning%20to%20Align%20and%20Translate.ipynb



encoder_input_layer: InputLayer
input: [(None, 24)]
output: [(None, 24)]
encoder_embedding_layer: Embedding
input: (None, 24)
output: (None, 24, 256)
encoder_lstm_1_layer: LSTM
input: (None, 24, 256)
output: [(None, 24, 256), (None, 256), (None, 256)]
encoder_bidirectional_layer(encoder_forward_lstm_layer): Bidirectional(LSTM)
input: (None, 24, 256)
output: [(None, 24, 512), (None, 256), (None, 256), (None, 256), (None, 256)]
decoder_input_layer: InputLayer
input: [(None, None)]
output: [(None, None)]
concatenate_1: Concatenate
input: [(None, 256), (None, 256)]
output: (None, 512)
concatenate: Concatenate
input: [(None, 256), (None, 256)]
output: (None, 512)
decoder_embedding_layer: Embedding
input: (None, None)
output: (None, None, 256)
decoder_lstm_layer: LSTM
input: [(None, None, 256), (None, 512), (None, 512)]
output: [(None, None, 512), (None, 512), (None, 512)]
attention_layer: AttentionLayer
input: [(None, 24, 512), (None, None, 512)]
output: ((None, None, 512), (None, None, 24))
concatenate_layer: Concatenate
input: [(None, None, 512), (None, None, 512)]
output: (None, None, 1024)
time_distributed_layer(dense_layer): TimeDistributed(Dense)
input: (None, None, 1024)
output: (None, None, 5001)

Design Choices

- Embedding size: 256
- Hidden units: 512
- Sequence length: 24
- Vocabulary size: 5000

Results

Training

- Parameters:
    - Training size: 700K samples
    - Validation size: 150K samples
    - Test size: 150K samples
    - Epochs: 10
    - Batch size: 256
    - Optimizer: RMS prop
    - Loss: Sparse categorical cross-entropy

Datasets

- Parallel corpus of German and English sentences
- Europarl:
  - 1.9M samples
  - Contains political speeches (complicated, nested sentences)
- ELRC:
  - 53K samples
  - contains German-English texts extracted from the website of the Federal Foreign Office Berlin.
- Rapid:
  - 1.5M samples
  - Contains news reports
- Custom:
  - 1M samples
  - mixed from the above datasets

Human Evaluation

- Variables
  - A: Number of sentences that were correct
  - B: Similarly score given by human evaluator
  - C: Grammatical correctness score given by human evaluator
- Formula

$$Z = \frac{1}{100}\left[A \times \frac{B+C}{2}\right]$$

Examples

Good:

Review German: technische hilfe aus <unk> fur kroatien im vorfeld des <unk>
Original English: technical assistance extended to croatia ahead of eu accession
Predicted English: technical assistance for croatia ahead of the accession

Bad:

German: mit blick auf die <unk> <unk> , die es auf allen seiten gegeben hat , ist hier jede <unk> erforderlich , um
Original English: given the serious willingness to negotiate shown by all sides , every effort is needed to reach an overall outcome.
Predicted English: with regard to the terrorist of the , the us of the people , we are now here to do ,

Results

	Europarl 1M	***Europarl 30K***	***Rapid 1M***	***Rapid 30K***	***ELRC 50K***	***Custom 1M***
Training Time	1h 9min	7min	1h 9min	4min	0h 10min	2h 10min
BLEU-1	<u>71.82</u>	63.88	67.89	58.05	58.86	**<u>72.69</u>**
BLEU-2	<u>61.40</u>	49.41	59.90	47.62	46.17	**<u>63.57</u>**
BLEU-3	53.92	39.01	<u>54.58</u>	41.46	37.78	**<u>57.41</u>**
BLEU-4	48.31	31.87	<u>50.52</u>	37.31	31.99	**<u>52.78</u>**
Human Eval.	10.0	7.0	27.0	26.5	9.0	**<u>35.0</u>**

(Best results marked in **<u>bold+underlined</u>**, Second-best results <u>underlined</u>)

Results (Transfer Learning)

Trained on: RAPID 1M
Evaluated on: EUROPARL 1M

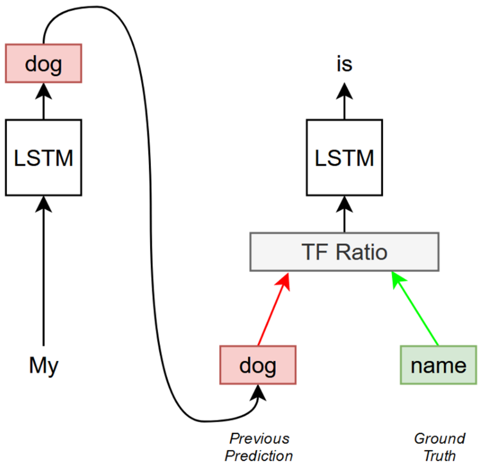
	Rapid 1M	***Europarl 1M***
BLEU-1	**67.89**	55.18
BLEU-2	**59.90**	34.99
BLEU-3	**54.58**	21.46
BLEU-4	**50.52**	15.37

Challenges & Review

- Adapting architecture to memory requirements:
  - When training on larger parameters, model ran out of memory.
- Difficulties in finding suitable vocab-size
- Consider implementing Teacher Forcing Ratio
  - Considering we have limited vocab size (5000)
- Dealing with Unknown tokens:
  - We chose the second most probable token when encountered.

Challenges & Review

- Adapting architecture to memory requirements:
  - When training on larger parameters, model ran out of memory.
- Difficulties in finding suitable vocab-size
- Consider implementing Teacher Forcing Ratio
  - Considering we have limited vocab size (5000)
- Dealing with Unknown tokens:
  - We chose the second most probable token when encountered.



Challenges & Review

- Adapting architecture to memory requirements:
  - When training on larger parameters, model ran out of memory.
- Difficulties in finding suitable vocab-size
- Consider implementing Teacher Forcing Ratio
  - Considering we have limited vocab size (5000)
- Dealing with Unknown tokens:
  - We chose the second most probable token when encountered.

German: `mit blick auf die <unk> <unk> , die es auf allen seiten gegeben hat , ist hier jede <unk> erforderlich , um`
Original English: `given the <unk> willingness to negotiate shown by all sides , every effort is needed to reach an overall <unk> .`
Predicted English: `with regard to the terrorist of the , the us of the people , we are now here to do ,`

Take Home Messages

- **Larger embedding** size **encode more** information but take more training time.
  - Restrict embedding size between 100 to 300 depending on dataset size.
- Model works better with a Bidirectional LSTM than just forward LSTM layers.
  - **Bidirectional LSTM encode vicinity** of word to help in prediction.
- Datasets that have **larger variety**.
  - **Positively affects** results.
- Training text style influences predictions.
  - For different text style, prediction done in training style.
- One needs to think about how to deal with unknown vocabulary.
  - We took the second highest probable token as a design choice to deal with this.

Sources

- Images:
  - https://github.com/bentrevett/pytorch-seq2seq/blob/master/3%20-%20Neural%20Machine%20Translation%20by%20Jointly%20Learning%20to%20Align%20and%20Translate.ipynb
- Guideline:
  - https://towardsdatascience.com/neural-machine-translation-nmt-with-attention-mechanism-5e59b57bd2ac