Understanding Encoders-Decoders with an Attention-based mechanism

Harsh Sharma · Follow
Published in DataX Journal
9 min read · Feb 1, 2021

How attention-based mechanism completely transformed the working of neural machine translations while exploring contextual relations in sequences!

When it comes to applying deep learning principles to natural language processing, contextual information weighs in a lot! In the past few years, it has been shown that various improvement in existing neural network architectures concerned with NLP has shown an amazing performance in extracting featured information from textual data and performing various operations for a day to day life. One of the models which we will be discussing in this article is encoder-decoder architecture along with the attention model.

The encoder-decoder architecture for recurrent neural networks is actually proving to be powerful for sequence-to-sequence-based prediction problems in the field of natural language processing such as neural machine translation and image caption generation.

We will try to discuss the drawbacks of the existing encoder-decoder model and try to develop a small version of the encoder-decoder with an attention model to understand why it signifies so much for modern-day NLP applications!

Encoder-decoder — a brief introduction

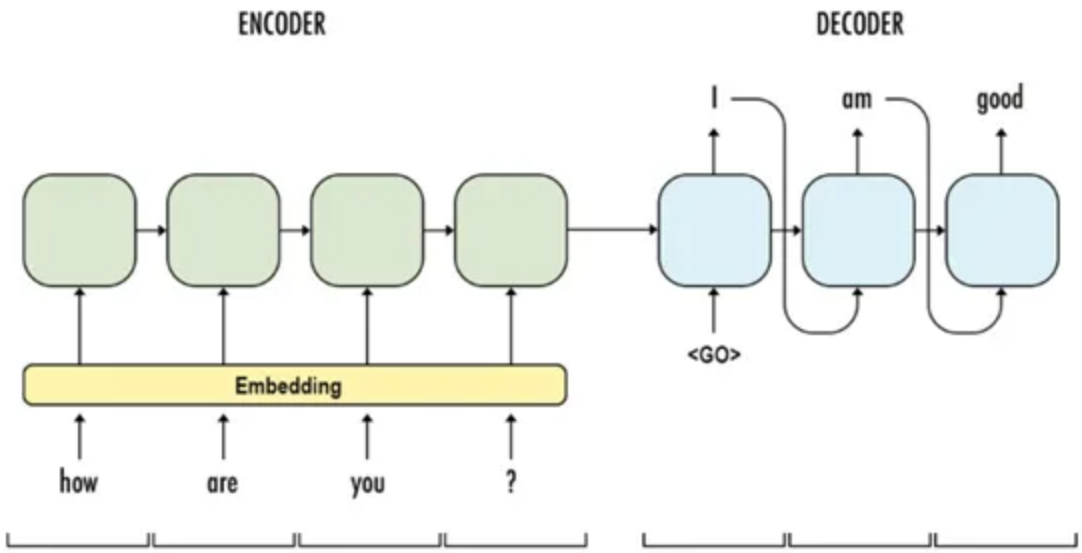
The encoder-decoder architecture with recurrent neural networks has become an effective and standard approach these days for solving innumerable NLP based tasks. The encoder-decoder model is a way of organizing recurrent neural networks for sequence-to-sequence prediction problems or challenging sequence-based inputs like texts [sequence of words], images [sequence of images or images within images] to provide many detailed predictions.

Encoder:

The encoder is a kind of network that **'encodes'**, that is obtained or extracts features from given input data. It reads the input sequence and summarizes the information in something called the **internal state vectors** or **context vector** (in the case of the LSTM network, these are called the hidden state and cell state vectors). We usually discard the outputs of the encoder and only preserve the internal states. This context vector aims to contain all the information for all input elements to help the decoder make accurate predictions. The hidden and cell state of the network is passed along to the decoder as input.

Decoder:

A decoder is something that **'decodes'**, interpret the context vector obtained from the encoder. The context vector of the encoder's final cell is input to the first cell of the decoder network. Using these initial states, the decoder starts generating the output sequence, and these outputs are also taken into consideration for future predictions. A stack of several LSTM units where each predicts an output (say y_hat) at a time step t.each recurrent unit accepts a hidden state from the previous unit and produces an output as well as its own hidden state to pass along the further network.



Classical sequence to sequence model — enocder -decoder

One of the main drawbacks of this network is its **inability to extract strong contextual relations from long semantic sentences**, that is if a particular piece of long text has some context or relations within its substrings, then a basic seq2seq model[short form for sequence to sequence] cannot identify those contexts and therefore, somewhat decreases the performance of our model and eventually, decreasing accuracy.

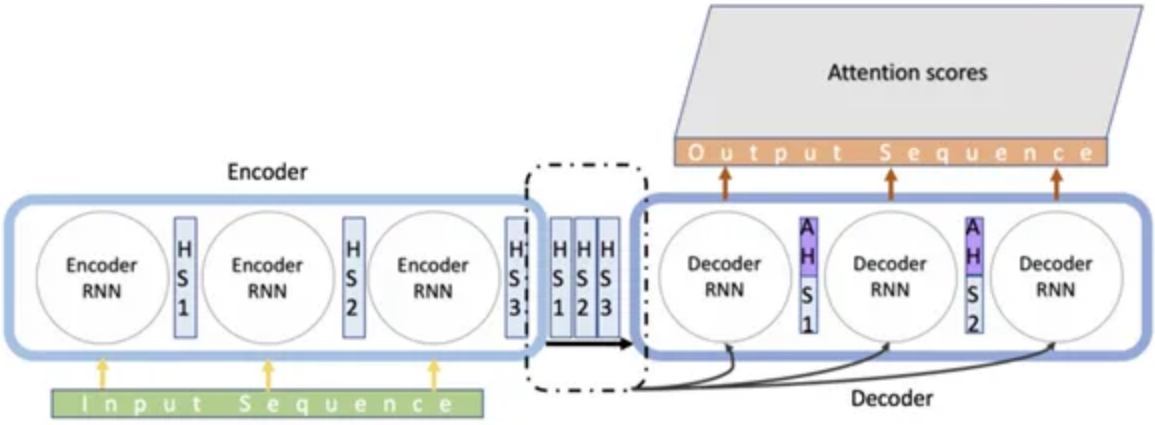
Keeping this in mind, a further upgrade to this existing network was required so that important contextual relations can be analyzed and our model could generate and provide better predictions.

Attention mechanism- basic working

Attention is an upgrade to the existing network of sequence to sequence models that address this limitation. The simple reason why it is called 'attention ' is because of its ability to obtain significance in sequences.

First, it works by providing a more weighted or more signified context from the encoder to the decoder and a learning mechanism where the decoder can interpret were to actually give more 'attention' to the subsequent encoding network when predicting outputs at each time step in the output sequence.

We can consider that by using the attention mechanism, there is this idea of freeing the existing encoder-decoder architecture from the fixed-short-length internal representation of text. This is achieved by keeping the intermediate outputs from the encoder LSTM network which correspond to a certain level of significance, from each step of the input sequence and at the same time training the model to learn and give selective attention to these intermediate elements and then relate them to elements in the output sequence.



Basic Architecture of Attention Based Model

In simple words, due to few selective items in the input sequence, the output sequence becomes conditional,i.e., it is accompanied by a few weighted constraints. These conditions are those contexts, which are getting attention and therefore, being trained on eventually and predicting the desired results.

Attention-based sequence to sequence model demands a good power of computational resources, but results are quite good as compared to the traditional sequence to sequence model. Besides, the model is also able to show how attention is paid to the input sequence when predicting the output sequence. This can help in understanding and diagnosing exactly what the model is considering and to what degree for specific input-output pairs.

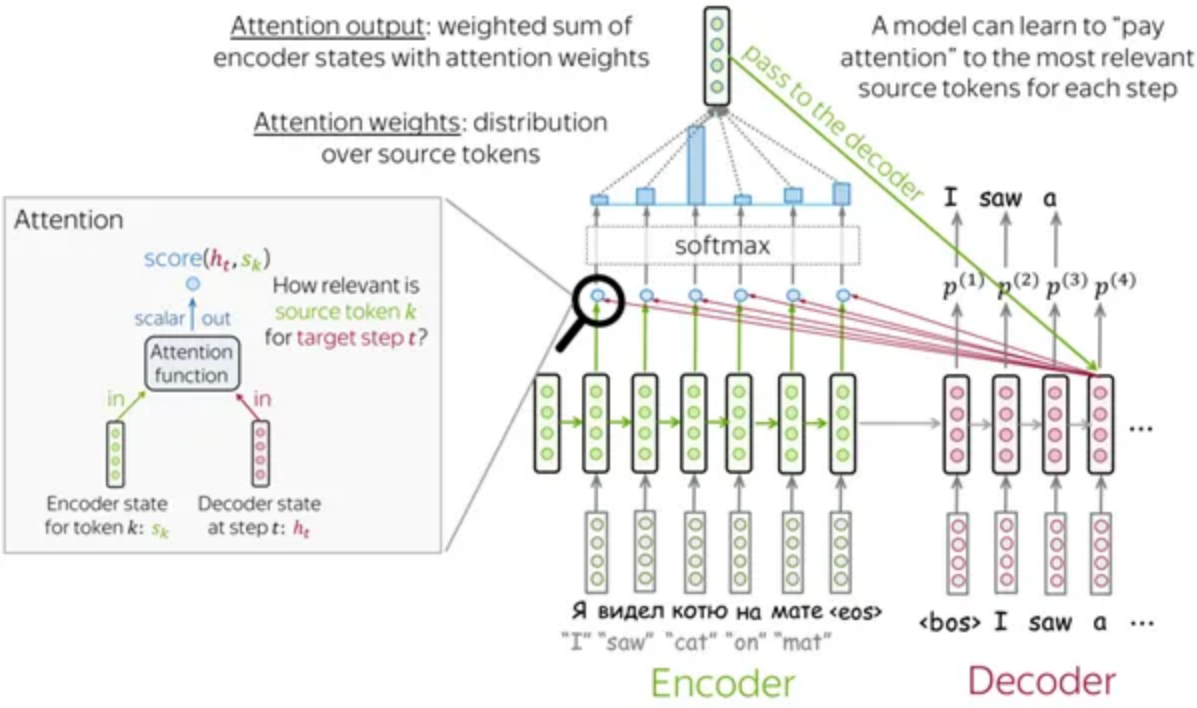
Rather than just encoding the input sequence into a single fixed context vector to pass further, the attention model tries a different approach. This model tries to

develop **a context vector that is selectively filtered specifically for each output time step,** so that it could focus and generate scores specific to those relevant filtered words and accordingly, train our decoder model with full sequences and especially those filtered words to obtain predictions.

Let us consider the following to make this assumption clearer.

Attention is proposed as a method to both align and translate for a certain long

of using the relevant information to select the appropriate output. With help of attention models, these problems can be easily overcome and provides flexibility to translate long sequences of information.



Let us try to observe the sequence of this process in the following steps:

1. In the encoder-decoder model, the input sequence would be encoded as a single fixed-length context vector. We will obtain a context vector that encapsulates the hidden and cell state of the LSTM network

2. The attention model requires access to the output, which is a context vector from the encoder for each input time step.

3. The decoder inputs need to be specified with certain starting and ending tags like <start> and <end>. These tags will help the decoder to know when to start and when to stop generating new predictions, while subsequently training our model at each timestamp.

4. The decoder outputs one value at a time, which is passed on to deeper layers further, before finally giving a prediction (say,y_hat) for the current output time step.

5. The alignment model scores (e) how well each encoded input (h) matches the current output of the decoder (s). The calculation of the score requires the output from the decoder from the previous output time step, e.g. S(t-1). When scoring the very first output for the decoder, this will be 0.

6. Scoring is performed using a function, let's say, a() is called the alignment model.

7. One of the very basic approaches for this network is to have one layer network where each input (s(t-1) and h1, h2, and h3) is weighted. With help of a hyperbolic tangent (tanh) transfer function, the output is also weighted.

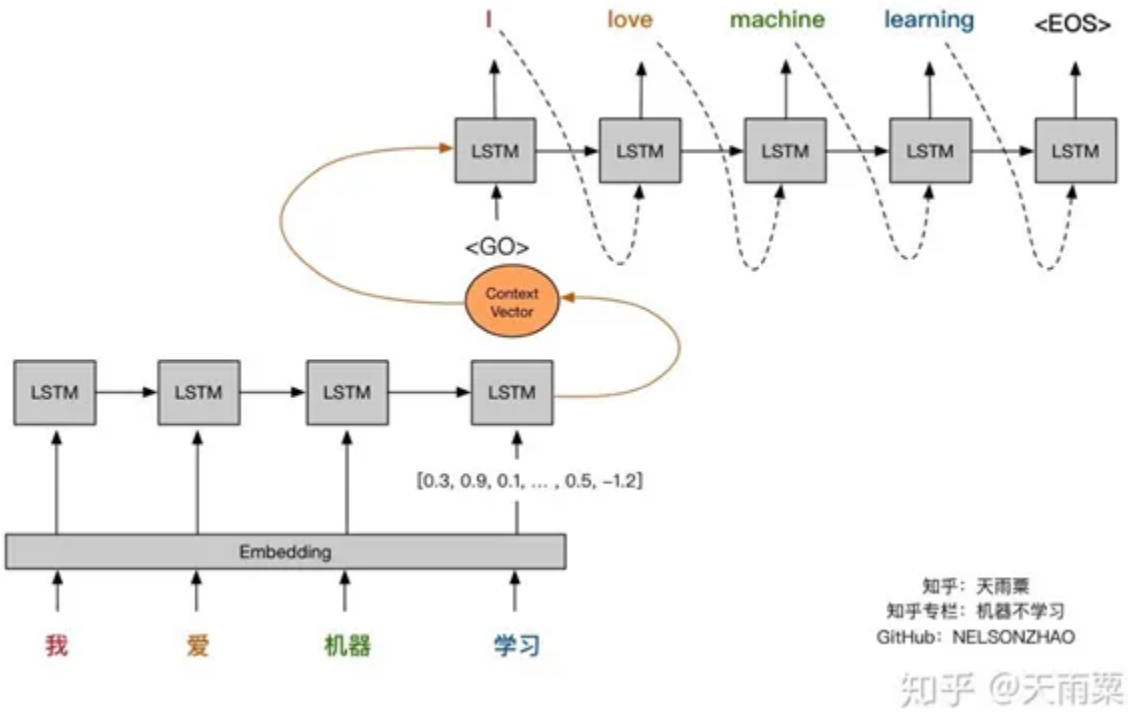
8. After obtaining the weighted outputs, the alignment scores are normalized using a **softmax function.** The normalization of the scores helps them to be treated like probabilities, indicating the likelihood of each encoded input time step (annotation) being relevant to the current output time step or not. These normalized scores are called annotation weights.

9. After obtaining annotation weights, each annotation, say,(h) is multiplied by the annotation weights, say, (a) to produce a new attended context vector from which the current output time step can be decoded. The context vector thus obtained is a weighted sum of the annotations and normalized alignment scores.

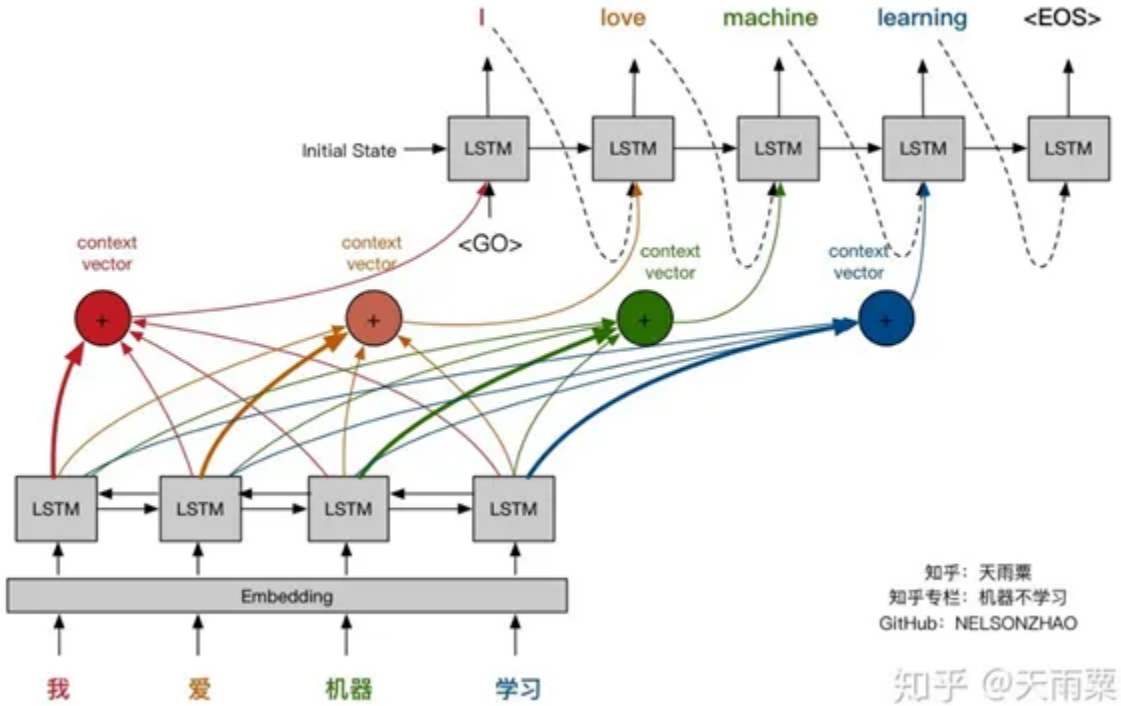
10. Finally, decoding is performed as per the encoder-decoder model, by using the attended context vector for the current time step. This attened context vector might be fed into deeper neural layers to learn more efficiently and extract more features, before obtaining the final predictions.

That being said, let's try to consider a very simple comparison of the model's performance between seq2seq with attention and seq2seq without attention model architecture.

Comparing attention and without attention-based seq2seq models



Encoder-Decoder with simple fixed context vector



Encoder-decoder with attention-based mechanism(https://zhuanlan.zhihu.com/p/37290775)

1. Unlike in the seq2seq model without attention, we used a fixed-sized context vector for all decoder time stamps but in the case of the attention mechanism,

we generate a context vector at every timestamp for filtered words with their respective scores.

2. Thanks to attention-based models, contextual relations are being much more exploited in attention-based models, the performance of the model seems very good as compared to the basic seq2seq model, given the usage of quite high computational power.

3. Using word embeddings might help the seq2seq model to gain some improvement with limited computational power, but long sequences with heavy contextual information might not get trained properly.

All this being given, we have a certain metric, apart from normal metrics, that help us understand the performance of our model — the BLEU score.

BLEU score

The bilingual evaluation understudy score, or BLEUfor short, is an important metric for evaluating these types of sequence-based models. It helps to provide a metric for a generated sentence to an input sentence being passed through a feed-forward model.

This score scales all the way from 0, being totally different sentence, to 1.0, being perfectly the same sentence.

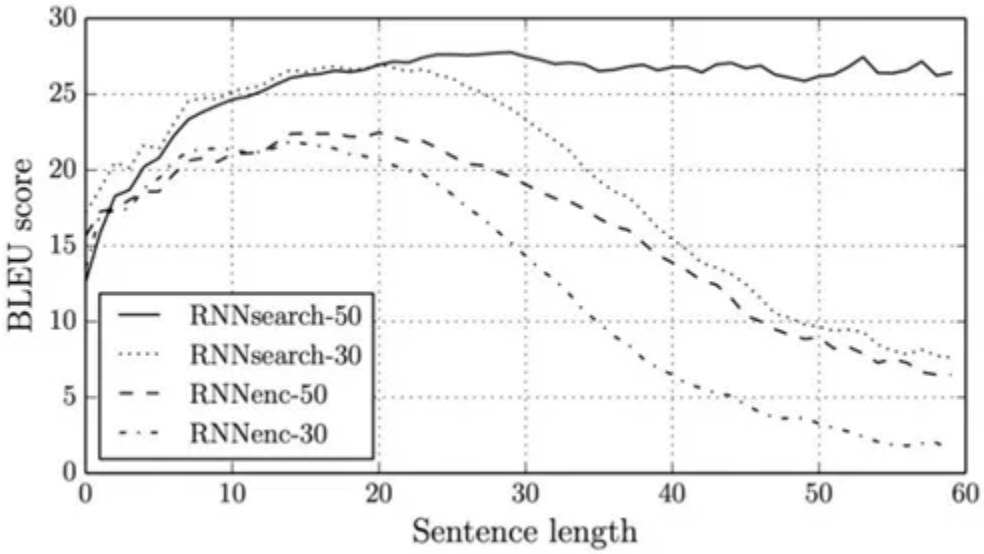
BELU score was actually developed for evaluating the predictions made by neural machine translation systems. Though is not totally perfect, but does offer certain benefits:

- It is quick and inexpensive to calculate.
- It is easy to understand.
- It is language independent.
- It correlates highly with human evaluation.
- It has been widely adopted.

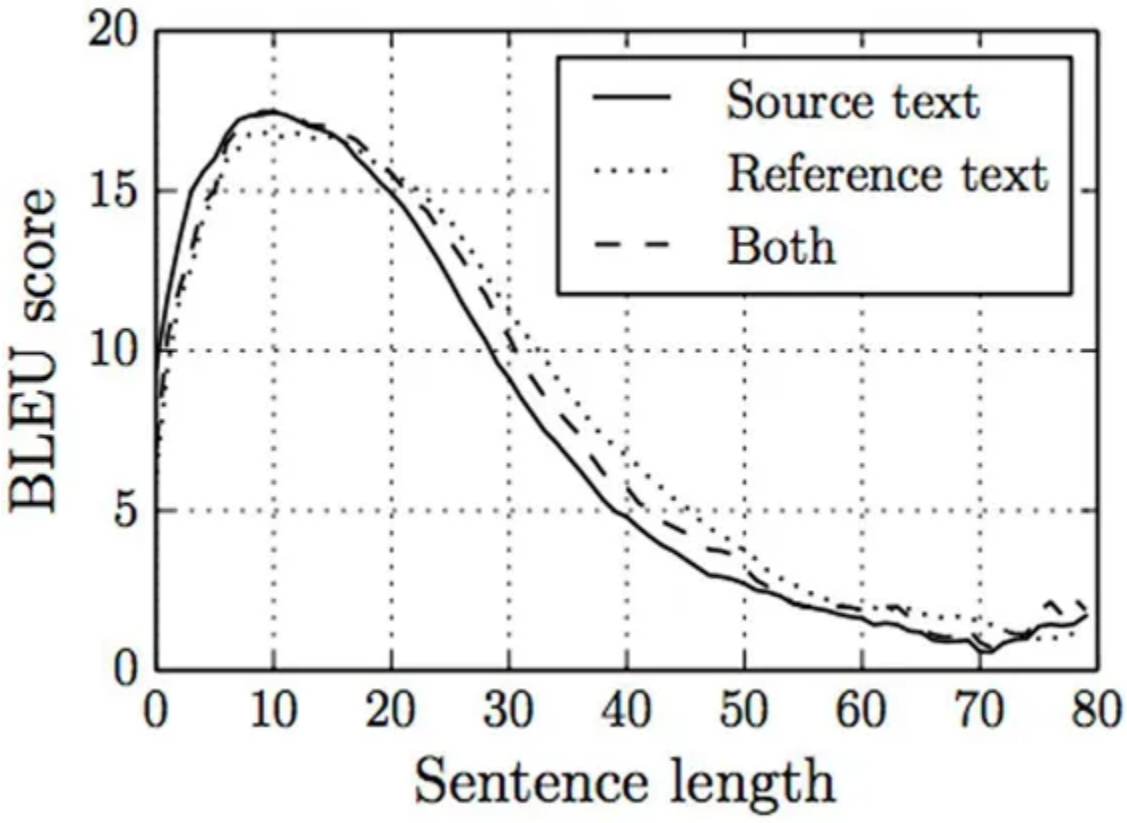
The python's own natural language toolkit library, or nltk, consists of the bleu score that you can use to evaluate your generated text against a given input text.nltk

provides the sentence_bleu() function for evaluating a candidate sentence against one or more reference sentences.

Given below is a comparison for the seq2seq model and attention model's bleu score:



Attention-based model BLEU score



Seq2Seq based model BLEU Score

Conclusion

After diving through every aspect, it can be therefore concluded that sequence to sequence-based models with the attention mechanism does work quite well when compared with basic seq2seq models. Exploring contextual relations with high semantic meaning and generating attention-based scores to filter certain words actually help to extract the main weighted features and therefore helps in a variety of applications like neural machine translation, text summarization, and much more. Implementing attention models with bidirectional layer and word embedding can actually help to increase our model's performance but at the cost of high computational power.

Though with limited computational power, one can use the normal sequence to sequence model with additions of word embeddings like trained google news or wikinews or ones with glove algorithm to explore contextual relationships to some extent, dynamic length of sentences might decrease its performance after some time, if being trained on extensively.

Deep Learning | Attention Model | Seq2seq | Naturallanguageprocessing | Data Science

Follow

Written by Harsh Sharma

50 Followers · Writer for DataX Journal

Applied NLP | Deep Learning Developer/ Research

More from Harsh Sharma and DataX Journal

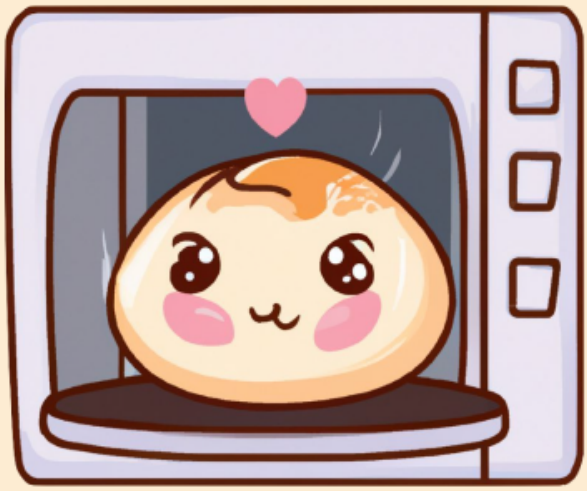
Harsh Sharma

My Transformative Experience as a MITACS Globalink Research Intern in Canada From India—2022

Summer 2022—was one of the most amazing times—I got an amazing opportunity to visit the University of British Columbia as a Research...

20 min read · Mar 26

44 2



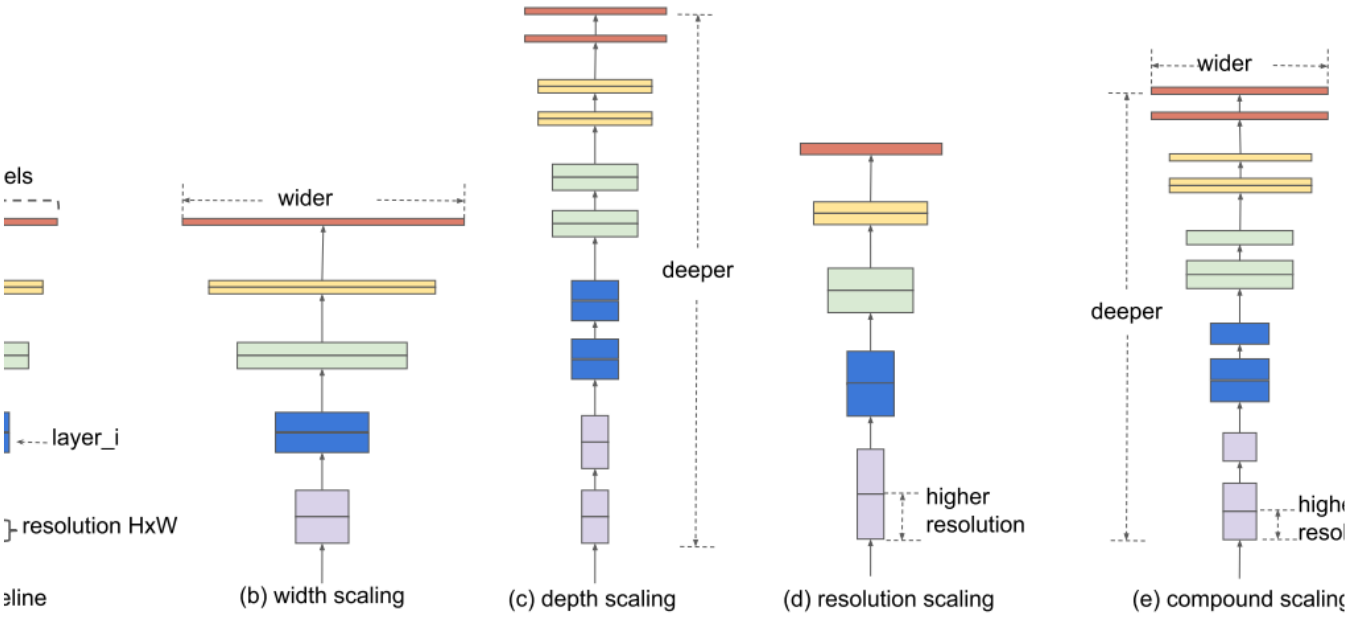
Vaishnav Manoj in DataX Journal

Unleashing the Speed: Exploring the Power of Bun.js and the Future of JavaScript Runtimes

Supercharging JavaScript: Unveiling the Speed and Innovation of Bun.js, the Game-Changing Runtime

17 min read · May 15

101 2



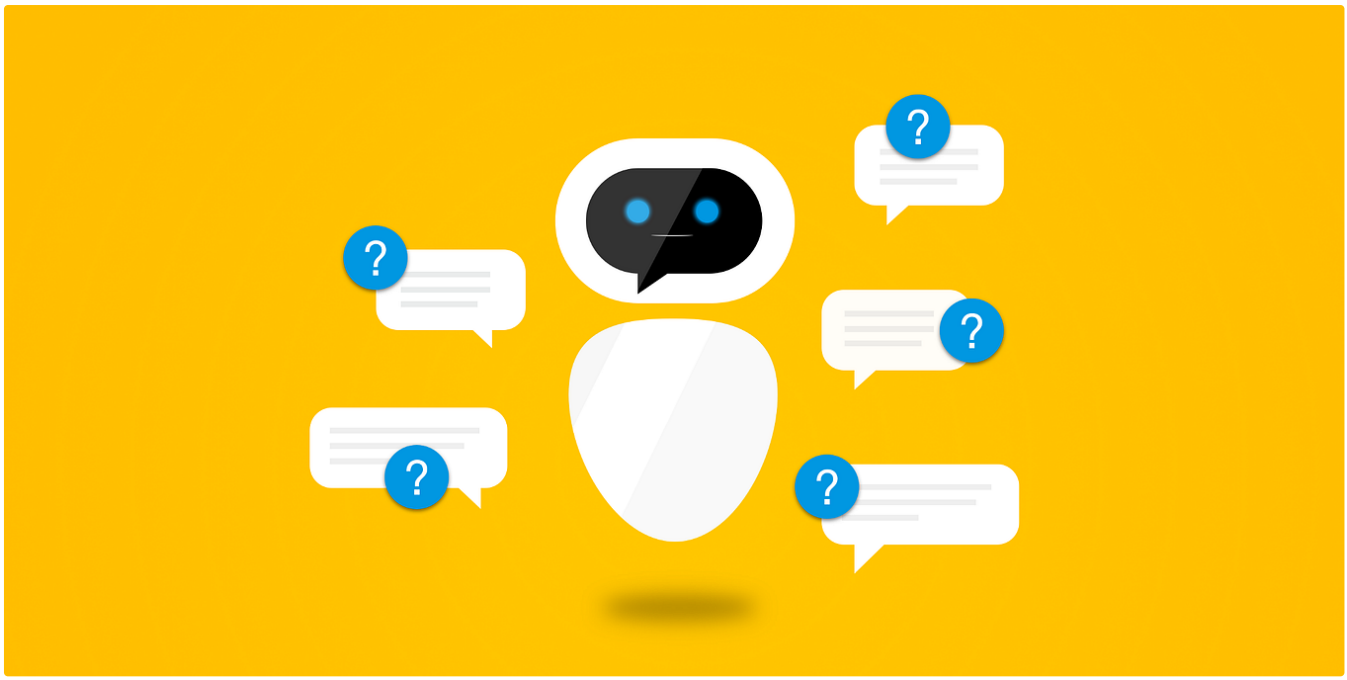
Aryan Raj in DataX Journal

EfficientNet : A New Approach to Model Scaling

Convolutional Neural Networks (CNNs), likewise referred to as ConvNets, are a major category in the field of neural networks and can been...

4 min read · 6 days ago

5 2



Harsh Sharma in DataX Journal

Developing Chatbots with RASA- Intuition to Implementation

"To the user, chatbots seem to be "intelligent" due to their informative skills. However, chatbots are only as intelligent as the...

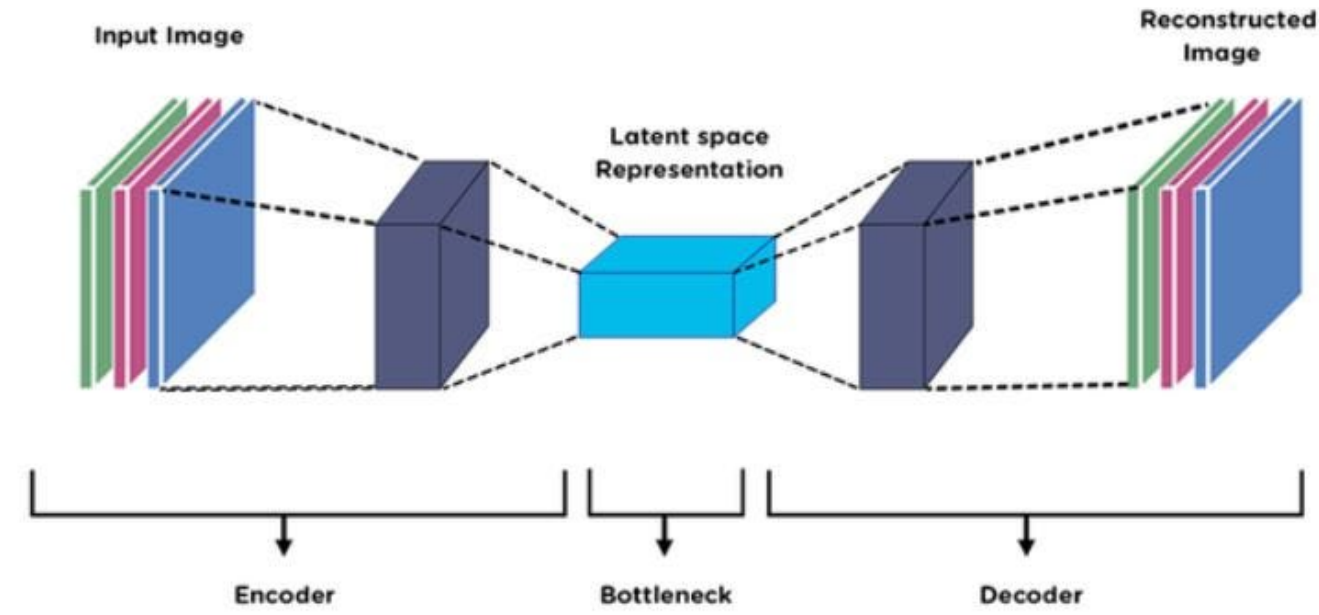
30 min read · Oct 6, 2020

161 1

See all from Harsh Sharma

See all from DataX Journal

Recommended from Medium

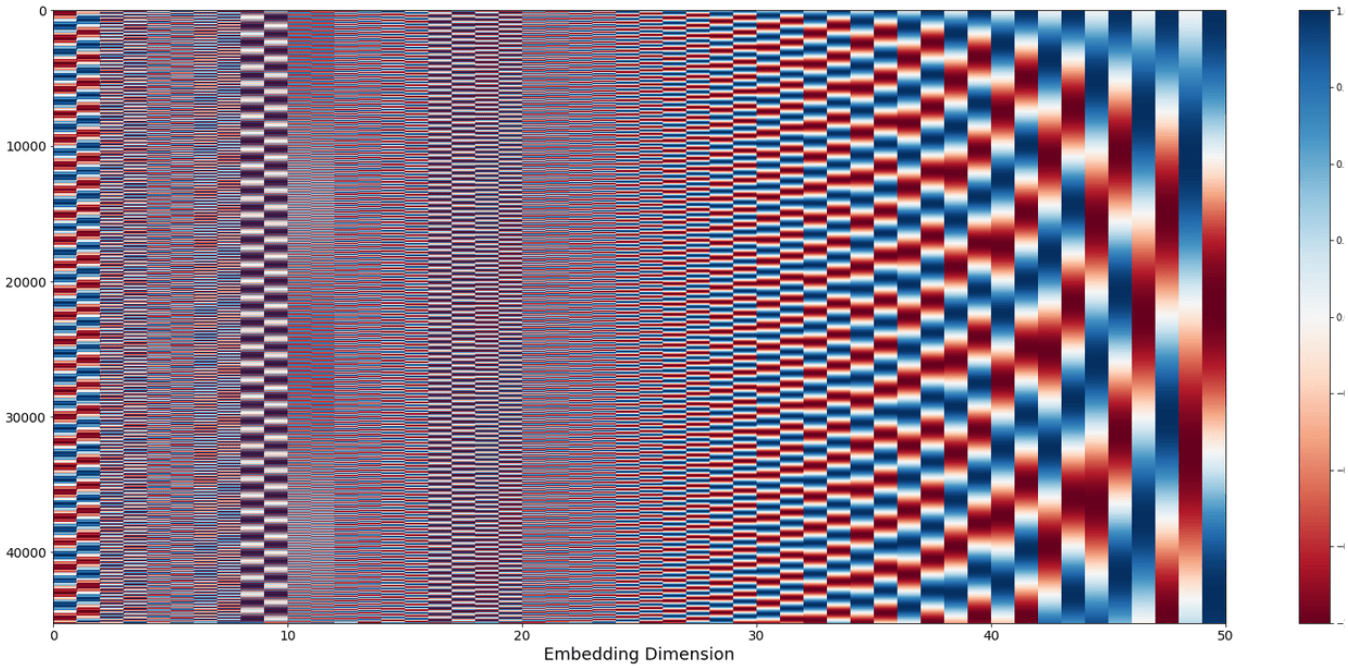


Ahmadsabry

A Perfect guide to Understand Encoder Decoders in Depth with Visuals

Introduction

6 min read · Jun 24



Eugene Ku

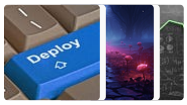
Transformer Architecture (Part 1—Positional Encoding)

Nowadays, arguably the most popular and influential model behind the hypes of deep learning comes from a model architecture called the...

4 min read · Aug 22

14

Lists

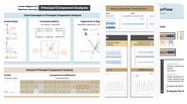


Predictive Modeling w/ Python

20 stories · 428 saves

New_Reading_List

174 stories · 125 saves



Practical Guides to Machine Learning

10 stories · 492 saves

Natural Language Processing

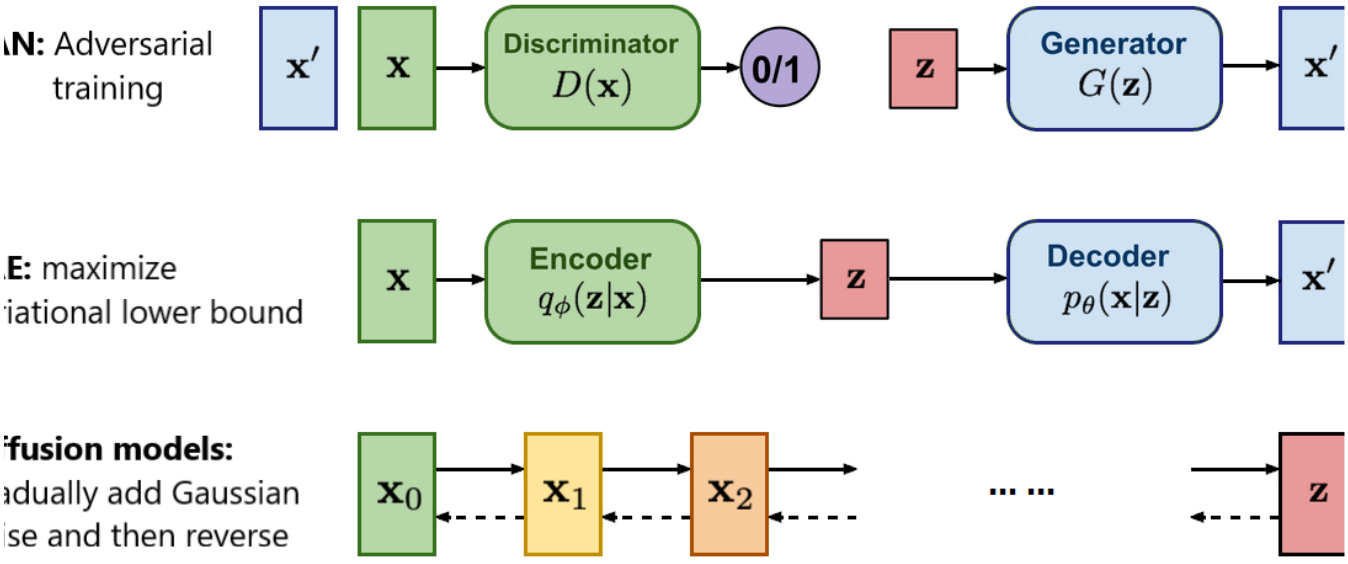
657 stories · 259 saves

Nimrita Koul

A simple example of computing Self-attention scores in Transformer model's Encoder block

Here I will show computation of simple self-attention scores (using dot product) for words in a small sentence. Assuming that positional...

2 min read · 6 days ago



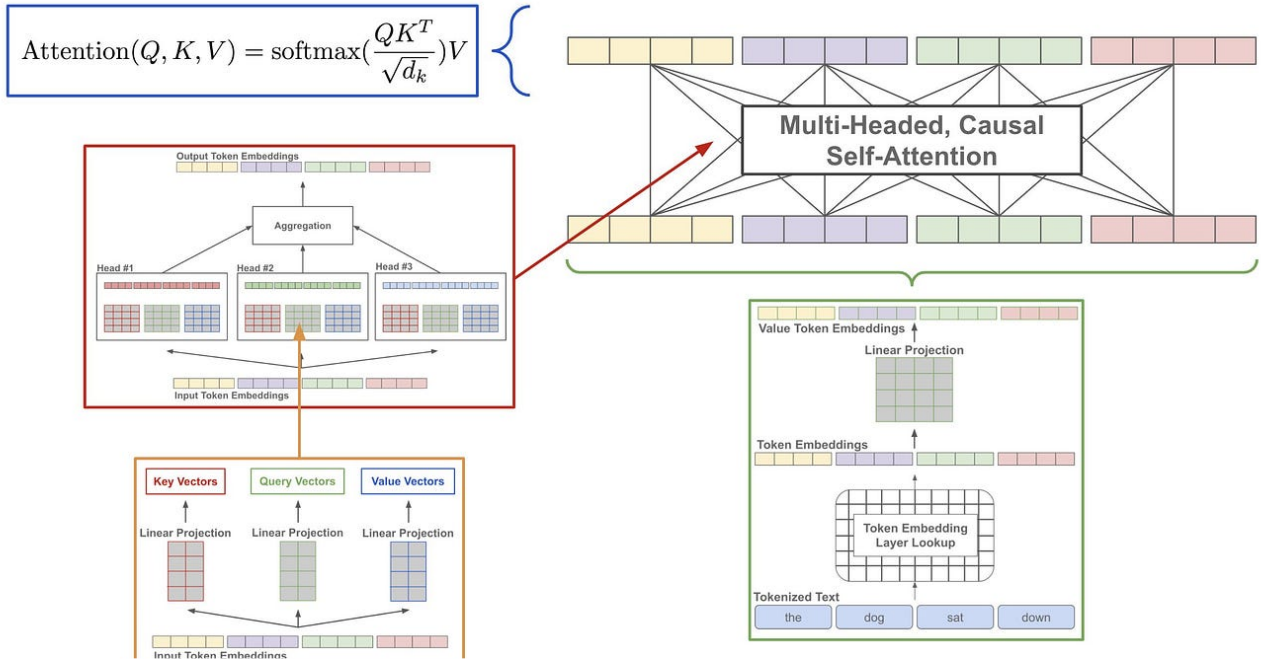
Ainur Gainetdinov in Towards AI

Diffusion Models vs GANs vs VAEs: Comparison of Deep Generative Models

Deep generative models are applied to diverse domains such as image, audio, video synthesis, and natural language processing. With the...

6 min read · May 12

249 2



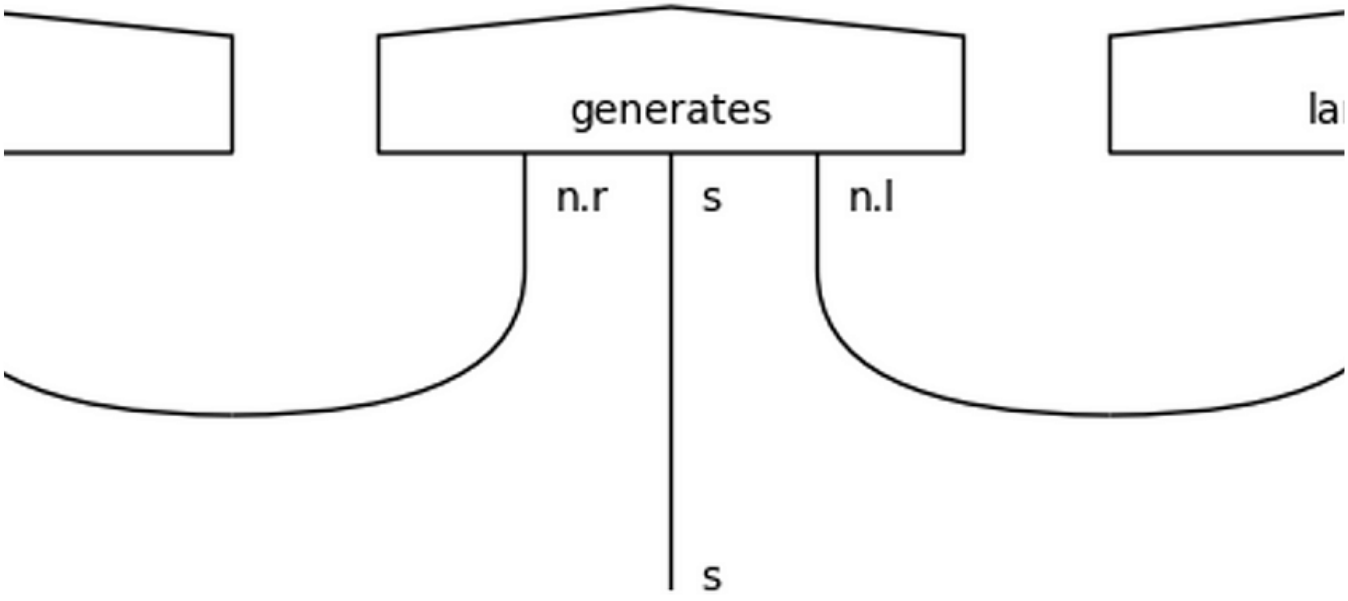
Akash Kesrwani

Multi-Head Self Attention: Short Understanding

Each “block” of a large language model (LLM) is comprised of self-attention and a feed-forward transformation. However, the exact...

3 min read · Sep 8

6



Qiskit in Qiskit

An introduction to Quantum Natural Language Processing

By Amin Karamlou, Marcel Pfaffhauser, and James Wootton

10 min read · Nov 4, 2022

 226  

See more recommendations