Sequence to Sequence

Krystian Derhak, Michael Lee, Yuvraj Singh

KEYWORDS

Seq2Seq, Recurrent Neural Networks, Gated Recurrent Unit, Long Short Term Memory, Language Translation, Speech Recognition, Chemical Synthesis

1 INTRODUCTION

To the layman, the field of artificial intelligence is one big block of marketing buzzwords that sound intelligent and recognizable yet difficult to explain their whole meaning. For example, Artificial Intelligence itself is a familiar phrase that would indicate the field is creating some form of robot or machine capable of thinking for itself, where the idea is conveyed correctly, but the particulars are fuzzy at best. Even those within artificial intelligence are constantly striving to learn the newer techniques that allow new and better solutions. This paper will discuss a specific subsection of Neural Networks known as sequence-to-sequence, hereafter referred to as seq2seq, some of the most recent use cases, and the research into improving these types of neural networks.

2 SEQUENCE TO SEQUENCE MODELS

Before discussing the particulars of Sequence to Sequence (seq2seq) models, it is necessary to have a basic foundation of neural networks. At a high-level view, a neural network has an input layer, an output layer, and several hidden layers, as shown in the top network of Figure 1. The idea is that one can input information into the network, and the hidden layers make educated decisions about the output. Once a network has been trained on a large enough test set and enough connections have been made, the neural network can receive new information and make the correct decision on data it's never seen before. A simple example would be financing, where a neural network can watch infinitely more transactions and look for fraudulent transactions.

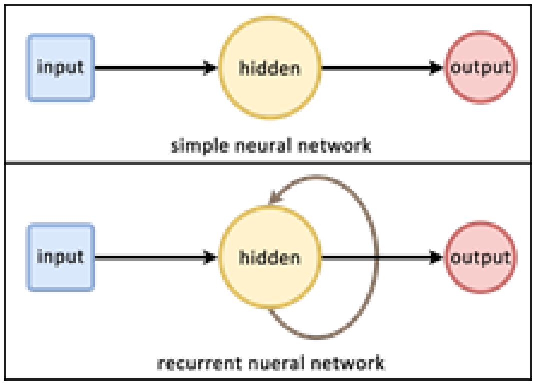


Figure 1: Structures of neural network and recurrent neural network

2.1 Recurrent Neural Networks

While neural networks seem like a magic bullet, they struggle in a few areas that require specialization and consideration. For example, although neural networks can perform text prediction, the memory of former inputs is necessary for making predictions that make sense. Solutions such as Recurrent Neural Networks (RNNs), in which the network can retain some information called memory in the form of states, were thus developed.

Recurrent Neural Networks are practically the same as a regular neural network, except for a slight difference in the utilization of a feedback loop, as seen in Figure 1. This feedback loop saves the output of the processing node and feeds the result back into the hidden layer. For a standard neural network, the information is passed in one direction. Each node within the hidden layers of RNNs act as a memory cell, where data is retained as computation is continued. RNNs are helpful in performing time-series prediction because of their feature to remember previous inputs, therefore working great with sequential data.

Even with the knowledge of states, traditional recurrent neural networks suffer from short-term memory issues caused by the phenomenon known as vanishing gradient. Figure 2 illustrates how vanishing gradient gives rise to short-term memory issues. Vanishing gradient is when the gradient, or value used to adjust weights throughout the neural network, loses its value over time, thereby diminishing the network's ability to learn.

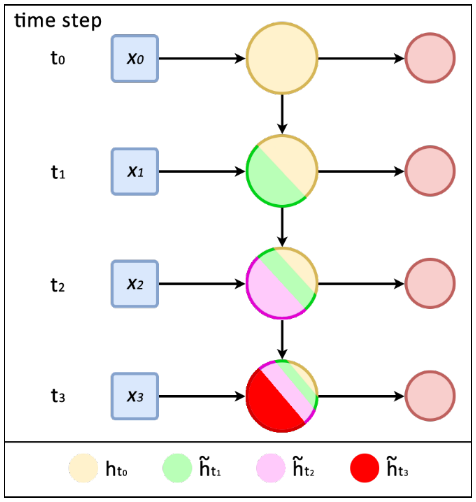
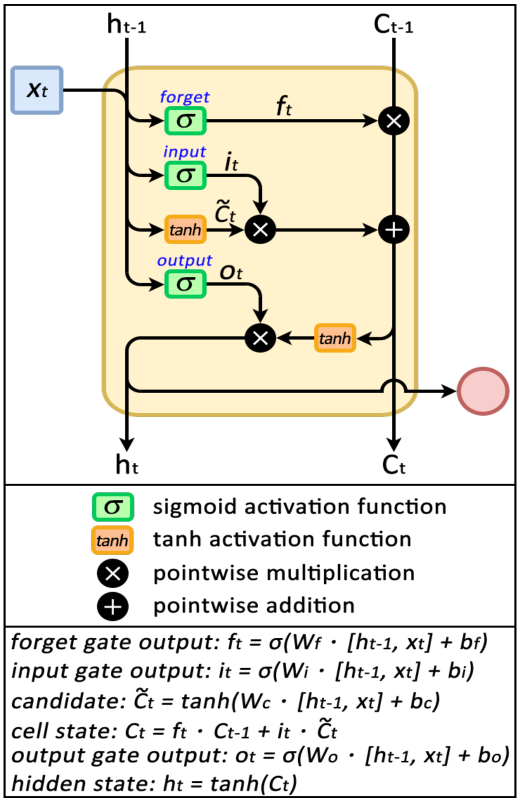


Figure 2: Diagram of short-term memory due to vanishing gradient

2.2 Long Short-Term Memory

To overcome the issue of vanishing gradients, two specialized versions of Recurrent Neural Networks were proposed. The first solution is a model called Long Short-Term Memory (LSTM). Figure 3 depicts the structure of an LSTM network. Within LSTM networks, more information is used when deciding the hidden states such as the cell states and gate mechanism, including forget gate, input gate, and output gate.


ht-1
Ct-1
xt
forget
σ
ft
input
σ
it
C̃t
tanh
output
σ
ot
tanh
ht
Ct
σ sigmoid activation function
tanh tanh activation function
⊗ pointwise multiplication
⊕ pointwise addition

forget gate output: $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
input gate output: $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
candidate: $\tilde{C}_t = \tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$
cell state: $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$
output gate output: $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$
hidden state: $h_t = \tanh(C_t)$

Figure 3: Structure of LSTM and formulas of the internal networks

The previous cell state contains the information of all previous inputs. Three gates are composed of a neural network and a sigmoid activation function with a point-wise multiplication operator, which outputs a value from zero to one, as seen in Figure 4 Left. The function of the forget gate is to decide how much previous information should be thrown away or kept according to the current input and the previous hidden state (the input vector [hprevious, x]); the input gate decides how much current input should be used for updating the cell state. The hyperbolic tangent activation function before the input gate is to regulate all values from the input vector for avoiding producing astronomical values, as seen in Figure 4 Right. Cell state is then updated based on the outputs of forgetting and input gates; the output gate determines the amount of cell state used as the hidden state passed onto the next layer of the network or used as the output. With these gates in place, the effects of vanishing gradient are mitigated. [6]

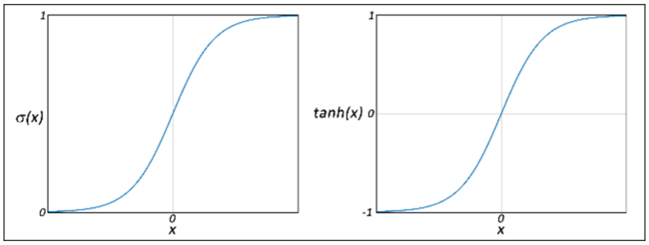
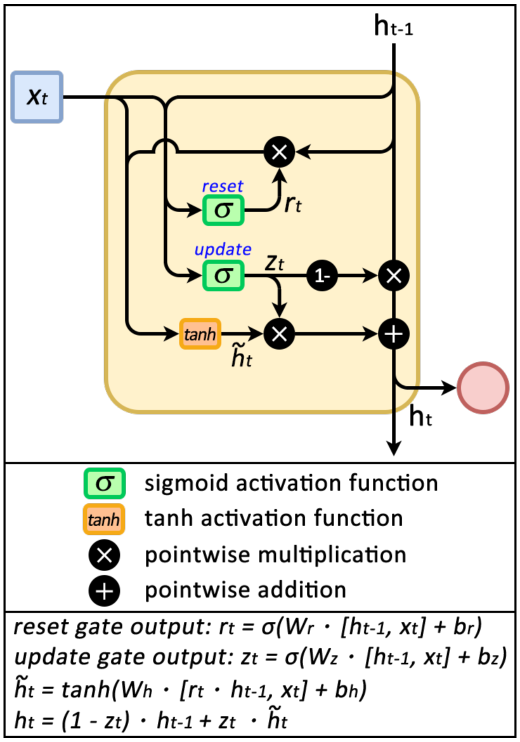


Figure 4: Left- Sigmoid (σ) activation function as gate mechanism with 0 = close, 1 = open, and any value in between = partially open; Right- hyperbolic tangent (tanh) activation function to regulate vector values

2.3 Gated Recurrent Unit

Another solution to the issue of vanishing gradients is the introduction of the Gated Recurrent Unit (GRU) model. GRUs are similar to LSTMs, but the difference is the operations performed within the unit itself. GRUs have only two gates, a reset gate and an update gate. Also, GRUs remove the cell states and use only the hidden states to transport information instead. Figure 5 depicts the structure of a GRU network.


ht-1
xt
reset
σ
rt
update
σ
zt
1-
tanh
h̃t
ht
σ sigmoid activation function
tanh tanh activation function
⊗ pointwise multiplication
⊕ pointwise addition

reset gate output: $r_t = \sigma(W_r \cdot [h_{t-1}, x_t] + b_r)$
update gate output: $z_t = \sigma(W_z \cdot [h_{t-1}, x_t] + b_z)$
$\tilde{h}_t = \tanh(W_h \cdot [r_t \cdot h_{t-1}, x_t] + b_h)$
$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t$

Figure 5: Structure of a GRU and formulas of the internal networks

Gates are neural networks themselves, where each network has its own weights. Each loop of the feedback loops is considered the cell states. The update gate is similar to the combination of the forget gate and the input gate from an LSTM model, and it decides what information should be thrown away or added to the hidden state. [2]. The reset gate is used to determine how much the previous information (the hidden state) to keep [2]. The introduction of these gates allows a better control flow of the gradients as they are updated throughout the model, making both solutions to short-term memory. Since the structure of GRUs are less complex than LSTMs', GRUs are more suitable for smaller data sets, and they are a little faster to train. Researchers usually apply both of them to determine which one is better for their cases.

2.4 Seq2Seq Models Overview

With these specific types of networks, there is still a struggle when the sequence of the input matters. Large swathes of problems rely on this stipulation, one of the largest being language-based problems where the meaning of sentences can change based on the sequence. For example, the sentence, "Hello, how are you?" can be scrambled to "how hello? you are" and while all words can be found in a dictionary and are spelled correctly, and the average human reader might be able to conceptually understand the message, the sentence is technically meaningless. This is where seq2seq comes into play.

As a subsection of neural networks, seq2seq models have many of the same components with a few more considerations. Beyond having an input and an output, the seq2seq has an encoder and a decoder. The encoder is responsible for converting the input sequence into a vector encompassing the input item and any context. The decoder reverses the process by taking that vector and feeding it through itself in stages to produce human-readable values.

3 APPLICATIONS

The theory is needed to understand more of the buzzwords that are discussed in the field of artificial intelligence, but it is not the main point. Why people are excited about artificial intelligence is the impressive results that are exhibited in the specific applications. The following subsections discuss three examples where seq2seq is utilized in real-life applications. As previously stated, due to the highly sequential nature of speech, two of the examples focus on speech. The first application is language translation, English to Japanese, for example, and the second application is speech recognition which involves speech recognition. The final application is the use of seq2seq in the field of chemical synthesis.

3.1 Language Translator

Language translation is a subfield within computational linguistics used to translate text from one language to another. Because artificial intelligence and machine learning models are often taught using numerical values, the use of vernacular data can be problematic. With seq2seq models, such linguistic information can be converted to numerical values, which will allow translation of words in one language to words in another. Figure 6 shows a simplified encoder with a corresponding encoder. The encoder is used to convert the input information into machine-readable in the form of a context vector that has lower-dimensional than its original form and thus occupies fewer resources. After the model's training, the vectorized input can then be converted back into a linguistic form using the decoder. While this can be effective on a simple word-to-word translation, the effectiveness declines drastically when using whole sentences or phrases since the translation requires comprehension of the entire sentence and its counterpart in the target language. The use of natural language processing techniques (NLP) allows neural networks to utilize sentence structures such as parts of speech and figures of speech to provide a more effective sentence translation. Figure 7 represents a basic seq2seq model utilizing an LSTM network to convert words from an input language to the corresponding words in the target language.

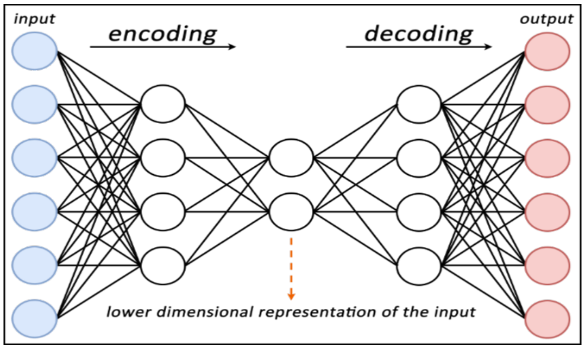


Figure 6: A simple diagram illustrates the structure of an encoder and a decoder

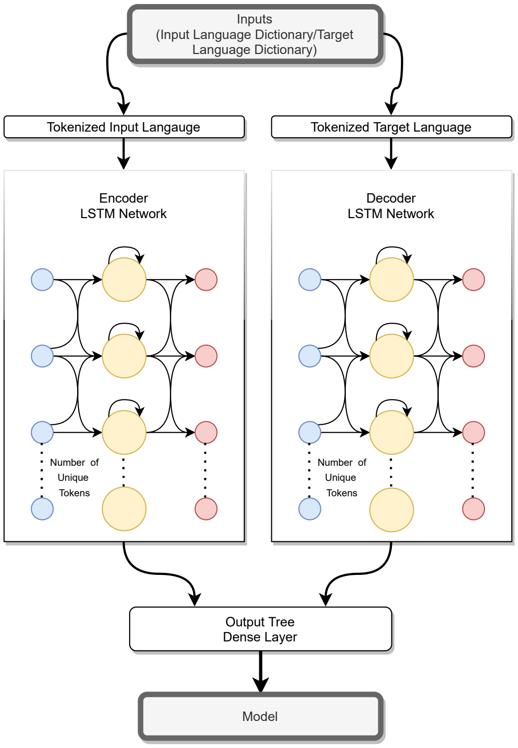


Figure 7: Example of a Long Short-Term Memory Model utilizing seq2seq for language translation

Using a seq2seq model similar to the one proposed in Figure 7, an English to Japanese translation model can be built. With such a model, a phrase such as "I love you" can be translated to the Japanese kanji representation of such phrase, as seen in Figure 8. Typically, when such seq2seq models are built, a set of data is retained for testing purposes like other models. Part of the testing process is to ensure that the encoder and decoder are functioning correctly by verifying that a specific word is encoded to the exact numerical vector each time and that such vector is decoded back to the proper word used during the encoding. The application's overall functionality can be evaluated at a systems level where users can test various inputs to ensure that they are being translated to the correct output.

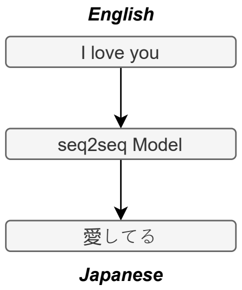


Figure 8: Language Translation: "I love you"

3.2 Speech Recognition

Speech recognition, also known as automatic speech recognition (ASR), is a capability similar to language translation, where the software can process human speech into a written format. With speech recognition, systems such as speaker labeling, profanity filtering, emotion recognition, and much more can be created. Speech recognition requires the use of analyzing the audio, categorizing the audio into separate parts, and then digitizing it into a computer-readable form. The audio is encoded similarly to how text is for language translation. Seq2Seq models created to handle ASR are called acoustic models, which maps segments of the encoded audio to phonemes [9]. These phonemes are connected together to form words, in which the seq2seq model used has to generate hypotheses to determine what the speaker said. Thus, the most suitable text representation is generated by decoding the most accurate hypothesis into its linguistic form. Overall, speech recognition applications are created very similarly to how language translation tools are. The main difference between the two is the data one needs to create a speech recognition application and how that data is formatted into context vectors that can be trained on.

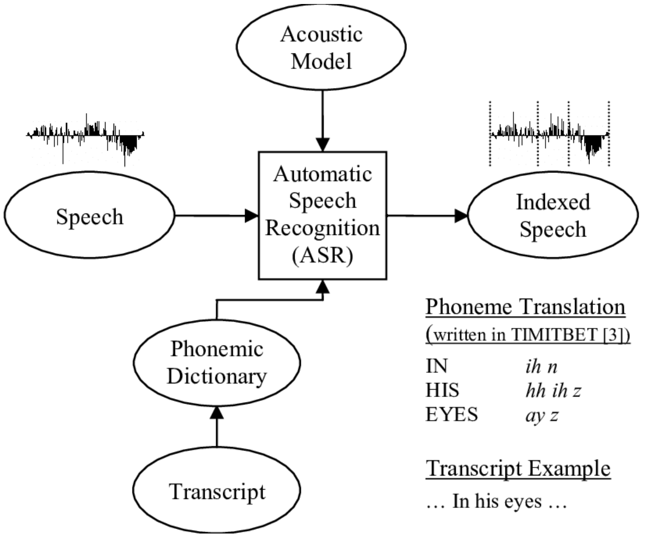


Figure 9: Overview of an Automatic Speech Recognition System [8]

RNNs are great compared to traditional neural networks due to the feedback loop system that allows previous data to be used in the next steps of the network. With this, models can be created where recent information can be used to make predictions about the future, which will allow ASR models to accurately select the correct hypothesis generated based on the previously selected. The problem with RNNs, as stated previously, is the lack of memory. With the addition of models like LSTMs and GRUs, speech recognition applications can hone in on specific pieces of important information and add/delete information with the aid of the logic gates within the networks [1].

The application of creating models to perform speech recognition is valuable and has many use cases given the technological advancements with this present day. While it can be a useful tool, there are many challenges when it comes to automatic speech recognition. The first of which is presented by two major factors, distance, and environment. One of the initial processes of automatic speech recognition is collecting the audio itself. An increase of distance from a microphone to the actual source of the speech allows biasing and noise to be included within the data itself. Loud environments also contribute to this as well; thus this calls for an even more precise system. Further difficulties arise from accents. Even though an individual might be speaking the same language a network is trained on, accents provide enough variance that the network will classify incorrectly the word spoken, which raises questions as to the efficacy of the training sets. This is an active area of research with many theories with few concrete results, for example networks trained on the Georgian language handle accents in that language better than most other networks trained on other languages [3]

3.3 Chemical Reaction Synthesis

Much of the modern world can be attributed to advances in the chemical realm. Society has been able to stronger buildings, smaller computer chips, better vaccines, warmer clothing, grow better foods, increasingly thanks to the ability to combine the basic building blocks of life, periodic elements, and biological molecules into new substances. The difficulty and danger when doing this are due to the fact that these reactions are unknown; they carry a significant risk to them as well as reward. Due to the complex way that many large molecules can break down, there are multiple different ways that an experiment can go. Previous attempts to use algorithms and deep learning were met with minimal success as they required prohibitively costly mathematical calculations to run or were simply inaccurate due to being rule-based and could only do basic molecules. Additionally, previous attempts needed large training sets to begin to be accurate. Traditionally chemical molecules and reactions are drawn as 2D and 3D graphs, but recently, there is an international system known as the Simplified molecular-input line-entry system, or SMILES, that conveys these molecules as lines of text. With this sequential textual-based input, a seq2seq LTSM based neural network can return accurate output as seen in Figure 10 [5].

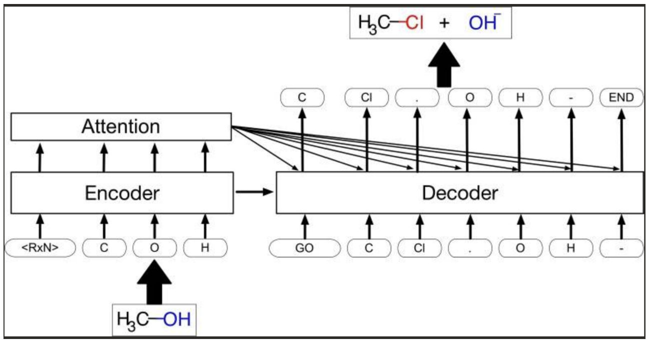


Figure 10: Example of a encoded simplified molecular-input line-entry system or SMILE encoding [5]

Initial results are promising. In most areas where traditional models do well, the seq2seq models also do well but require fewer resources. The seq2seq model also performs better than the traditional model when there are more complex molecules in question. While this work is promising, it is still relatively new in the world of chemistry. Further work and advancements will focus on optimizing the network and improving accuracy, training on more diverse data sets where fewer known reactions are documented as well as increasing the difficulty of the reactions presented to the network. The current working neural networks provide results too simplistic to realistically push the boundaries synthetic chemists face daily, but they are more broadly being used amongst less skilled scientists and technicians. Other researchers are focusing their efforts on unsupervised learning which would expand the capabilities of networks exponentially while coming at the cost of large training sets and computation time [11].

4 IMPROVEMENTS

As previously stated, most seq2seq networks rely upon LSTMs or GRUs as their base, which performs significantly better than a typical RNN. This is due to their ability to have longer-term memory, which allows for better learning and better performance. While LTSMs and GRUs have more memory, they still suffer from memory loss over significantly long sequences of text. A newer type of network called Transformers is being developed that overcome these shortcomings while retaining many similarities to these previously mentioned networks. Transformers still have an encoder and a decoder but contain additional components. The encoder includes attention information or information that tells the decoder which aspects of the sequence it should focus on and how it relates to other portions of the sequence. The decoder is unique in that the output of the decoder loops around and is also an input to the decoder, thereby retaining more information concerning the encoded input. Transformer networks are relatively new but already show promise and improvements being able to handle larger tasks while requiring less training [10].

5 CONCLUSIONS

We have shown the basics of neural networks and how specific subsections work, namely the Long Short-Term Memory (LSTM) and Gate Re-current Networks (GRU) operate. The main benefit that they utilize is their ability to remember state, or memory, which is critically important for sequence-based problems. As shown above, models using seq2seq networks are already employed in several language and scientific problems. While the improvements of LSTMs and GRUs are admirable, newer networks such as Transformer networks that incorporate further memory and improvements are beginning to be implemented that excel in these same fields [4] [7].

REFERENCES

[1] Recep Arslan. 2019. Development of Output Correction Methodology for Long Short Term Memory-Based Speech Recognition. *Sustainability* 11 (08 2019), 4250. https://doi.org/10.3390/su11154250

[2] Rahul Dey and Fathi M. Salem. 2017. Gate-variants of Gated Recurrent Unit (GRU) neural networks. In *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*. 1597–1600. https://doi.org/10.1109/MWSCAS.2017.8053243

[3] Irakli Kardava, Jemal Antidze, and Nana Gulua. 2016. Solving the problem of the accents for speech recognition systems. *International Journal of Signal Processing Systems* 4, 3 (2016), 235–238.

[4] Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K. Reddy. 2020. Deep Reinforcement Learning for Sequence-to-Sequence Models. *IEEE Transactions on Neural Networks and Learning Systems* 31, 7 (July 2020), 2469–2489.

[5] Bowne Liu, Bharath Ramsundar, Prasad Kawthekar, Jade Shi, Joseph Gomes, Quang Luu Nguyen, Stephen Ho, Jack Sloane, Paul Wender, and Vijay Pande. 2017. Retrosynthetic Reaction Prediction using Neural Sequence-to-Sequence Models. *ACS Central Science* 3, 10 (October 2017), 1103–1113.

[6] Christopher Olah. 2015. Understanding LSTM Networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[7] Flavio Padau, Rodrigo Carceroni, Geraldo Santos, and Kiriakos Kutulakos. 2010. Linear Sequence-to-Sequence Alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 2 (February 2010), 304–320.

[8] David Rossiter, Gibson Lam, and Brian Mak. 2006. Automatic Audio Indexing and Audio Playback Speed Control as Tools for Language Learning. 290–299. https://doi.org/10.1007/11925293_26

[9] Anuroop Sriram, Heewoo Jun, Sanjeev Satheesh, and Adam Coates. 2017. Cold Fusion: Training Seq2Seq Models Together with Language Models. *CoRR* abs/1708.06426 (2017). arXiv:1708.06426 http://arxiv.org/abs/1708.06426

[10] Wood Thomas. 2020. Transformer Neural Network. https://deepai.org/machine-learning-glossary-and-terms/transformer-neural-network

[11] Zheng Xu, Sheng Wang, Feiyun Zhu, and Junzhou Huang. 2017. Seq2seq fingerprint: An unsupervised deep molecular embedding for drug discovery. In *Proceedings of the 8th ACM international conference on bioinformatics, computational biology, and health informatics*. 285–294.