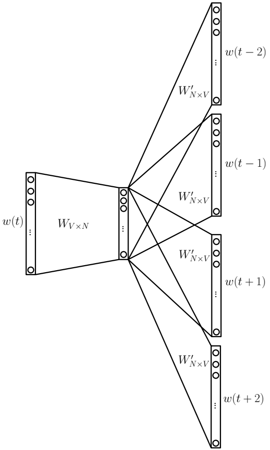
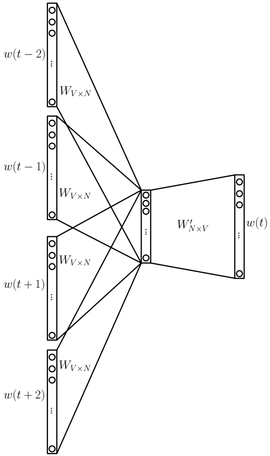
Selected Topics in NLP

By: Behzad Asadi

Word2Vec



Skip Gram: Considering a window, it is about predicting surrounding words given the centre word.



Continuous Bag Of Words: Considering a window, it is about predicting the centre word given surrounding words.

Word2Vec

Skip-Gram: Considering $\mathbf{v}_{w_i}$ as the ith row of the W matrix, and $\mathbf{v}'_{w_j}$ as the jth column of the W' matrix, the loss function is defined as

$$
\begin{aligned}
E &= -\log p(w_{O_1}, w_{O_2}, \cdots, w_{O_C} \mid w_I) \\
&= -\log \prod_{c=1}^{C} \frac{\exp({\mathbf{v}'}^T_{w_{O_c}} \mathbf{v}^T_{w_I})}{\sum_{j=1}^{V} \exp({\mathbf{v}'}^T_{w_j} \mathbf{v}^T_{w_I})} \\
&= -\sum_{c=1}^{C} {\mathbf{v}'}^T_{w_{O_c}} \mathbf{v}^T_{w_I} + C \cdot \log \sum_{j=1}^{V} \exp({\mathbf{v}'}^T_{w_j} \mathbf{v}^T_{w_I})
\end{aligned}
$$

CBOW Loss Function: Considering $\mathbf{x}_i$ as the one hot encoding column for the word i, $\mathbf{v}_{w_i}$ as the ith row of the W matrix, $\mathbf{v}'_{w_j}$ as the jth column of the W' matrix, and by defining $\mathbf{h}$ as

$$
\begin{aligned}
\mathbf{h} &= \frac{1}{C} W^T (\mathbf{x}_1 + \mathbf{x}_2 + \cdots + \mathbf{x}_C) \\
&= \frac{1}{C} (\mathbf{v}_{w_1} + \mathbf{v}_{w_2} + \cdots + \mathbf{v}_{w_C})^T,
\end{aligned}
$$

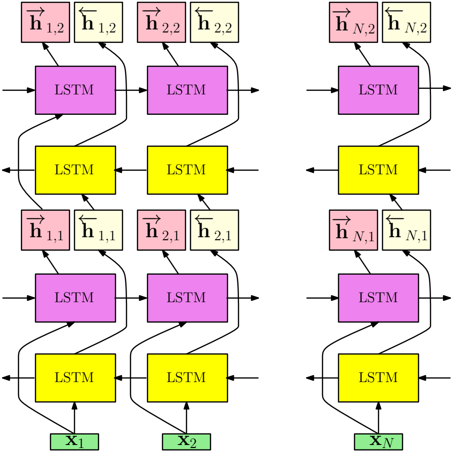
the loss function is defined as

$$
\begin{aligned}
E &= -\log p(w_O \mid w_{I_1}, w_{I_2}, \ldots, w_{I_C}) \\
&= -{\mathbf{v}'_{w_O}}^T \mathbf{h} + \log \sum_{j=1}^{V} \exp({\mathbf{v}'_{w_j}}^T \mathbf{h})
\end{aligned}
$$

ELMO: Embeddings from Language Models

It is a deep contextualized word representation. This means that the embedding corresponding to each word is a function of the entire sentence.

Bidirectional Language Model



Two-Layer BiLM

ELMO

Therefore, in ELMO, we have R_k which is a set of $2L+1$ representations corresponding to token k

$$
\begin{aligned}
R_k &= \left\{ \mathbf{x}_k, \overrightarrow{\mathbf{h}_{k,j}}, \overleftarrow{\mathbf{h}_{k,j}} \mid j = 1, 2, \dots, L \right\} \\
&= \left\{ \mathbf{h}_{k,j} \mid j = 0, 1, \dots, L \right\}
\end{aligned}
$$

where $\mathbf{h}_{k,0} = \mathbf{x}_k$, and $\mathbf{h}_{k,j} = \left[\overrightarrow{\mathbf{h}_{k,j}}, \overleftarrow{\mathbf{h}_{k,j}} \right]$.

In ELMO, the embedding corresponding to token k is computed as

$$
\mathbf{ELMO}_k = \gamma^{task} \sum_{j=0}^{L} s_j^{task} \mathbf{h}_{k,j}
$$

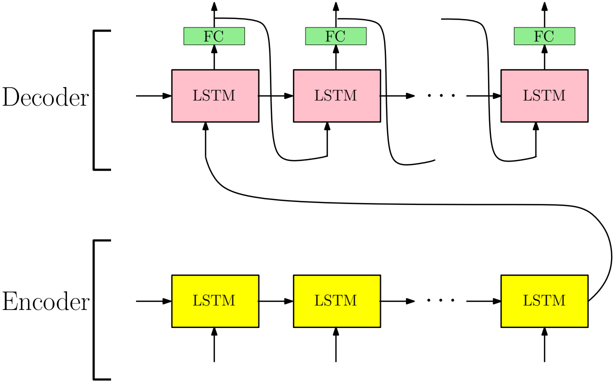
where $s_j^{task},\ j = 0, 1, \dots, L$ are softmax-normalized weights, and γ^{task} is a scaling factor. Considering a supervised NLP task in which embeddings are going to be utilised, these parameters are learned during the training process of that task.

ULMFiT: Universal Language Model Fine-Tuning

ULMFiT consists of three stages:

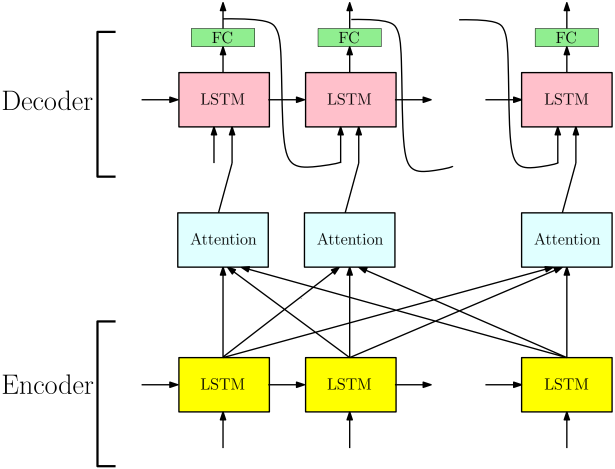
1. **Language Model Pre-Training**: In this part, a general corpus of text is used to train a language model.
2. **Language Model Fine-Tuning**: In this part, the language model is fine-tuned using the target task text. This is done using
   - Discriminative Fine-Tuning where different layers have different learning rates
   - Slanted Triangular Learning Rate where the learning rate first linearly increases, and then linearly decreases.
3. **Classifier Fine-Tuning**: In this part, first, two linear layers are added to the language model to form the classifier. Then, the following three techniques are used for fine-tuning
   - Discriminative Fine-Tuning
   - Slanted Triangular Learning Rate
   - Gradual Unfreezing

LSTM Encoder-Decoder Architecture



Encoder-Decoder

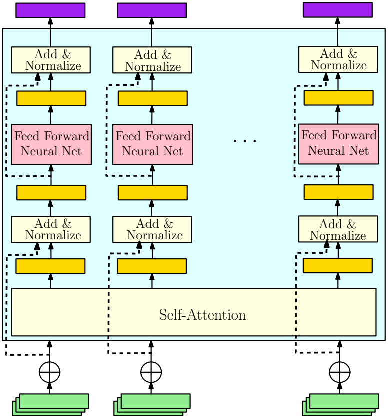
LSTM Encoder-Decoder Architecture with Attention



Encoder-Decoder with Attention

Transformer

Transformer consists of an encoder and a decoder. Here, we only review the encoder part which is a building block of BERT. [1] [2]



Transformer Encoder

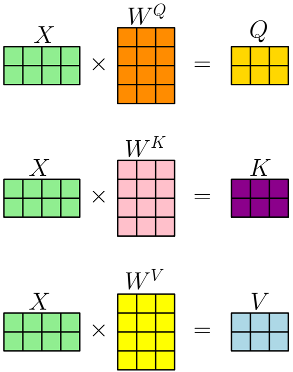
[1] Attention is All You Need

[2] http://jalammar.github.io/illustrated-transformer/

Self-Attention

$$\mathbf{q}_i = \mathbf{x}_i W^Q$$
$$\mathbf{k}_i = \mathbf{x}_i W^K$$
$$\mathbf{v}_i = \mathbf{x}_i W^V$$

$$Q = XW^Q$$
$$K = XW^K$$
$$V = XW^V$$



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

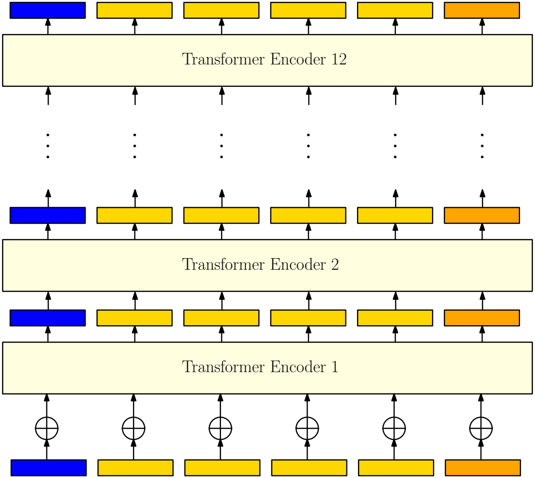
$$\text{MultiHead Attention} = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

where $\text{head}_i = \text{Attention}(Q_i, K_i, V_i)$, $Q_i = XW_i^Q$, $K_i = XW_i^K$, $V_i = XW_i^V$

BERT: Bidirectional Encoder Representations from Transformers

BERT Training Tasks:

- Masked Language Model
- Next Sentence Prediction



BERT

References

1. T. Mikolov et al. “Distributed Representations of Words and Phrases and their Compositionality".
2. X. Rong, “word2vec Parameter Learning Explained".
3. M. E. Peters et al., “Deep contextualized word representations".
4. J. Howard, and S. Ruder, “Universal Language Model Fine-tuning for Text Classification".
5. D. Bahdanau et al., “Neural Machine Translation by Jointly Learning to Align and Translate".
6. A. Vaswani et al., “Attention is All You Need".
7. J. Devlin et al., “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding".