Sentiment Classifier Empowered With LSTM Autoencoder

Sayan Ghosh

ID-B1930063

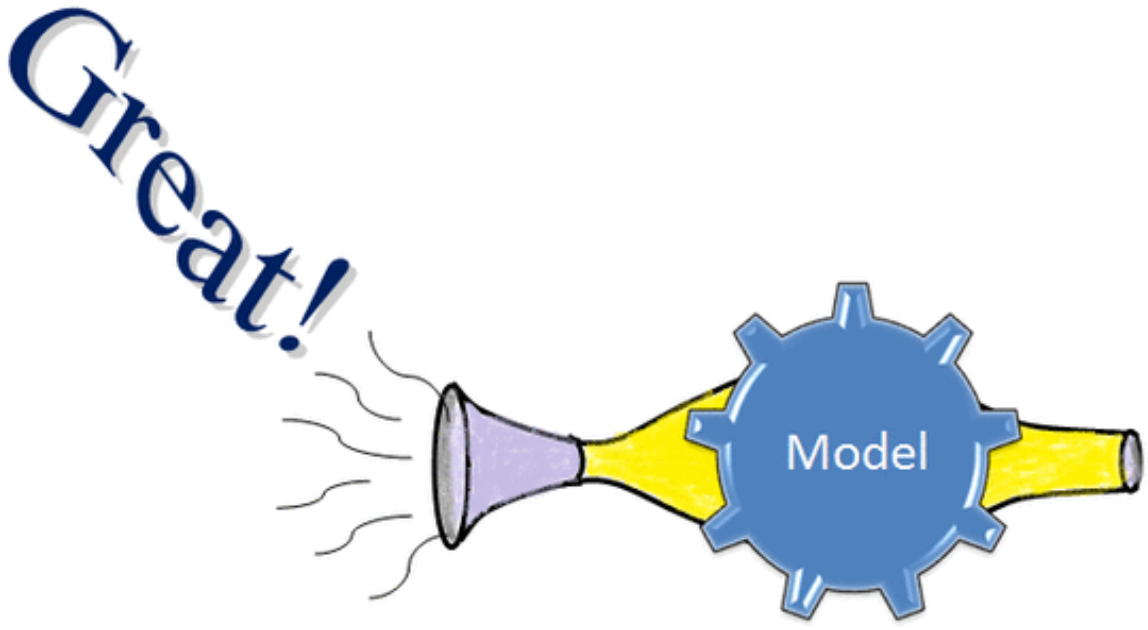
Big Data Analytics 2019-21

Computer Science Department

RKMVERI

Introduction

Overview of Sentiment Analysis



Description of Amazon Reviews Dataset

- **Source:** Kaggle
- **Features:**
  - Data is in *fastText*[5] format
  - Training dataset: Consists 36 lakh examples,
    Used 2 lakh for training, 1 lakh for validation
    Test dataset: Consists 4 lakh examples,
    Used 2 lakh for testing
  - Multilingual data (Consists of Spanish, French etc. along with English)
    Translated non-English language to English using Google translator [4]
  - Two labels: *__label__1* & *__label__2*
    __label__1 $\Longrightarrow$ 1 & 2 star ratings
    __label__2 $\Longrightarrow$ 4 & 5 star ratings

Description of Amazon Reviews Dataset

- **Source:** Kaggle
- **Features:**
  - Data is in *fastText*[5] format
  - Training dataset: Consists 36 lakh examples,
    Used 2 lakh for training, 1 lakh for validation
    Test dataset: Consists 4 lakh examples,
    Used 2 lakh for testing
  - Multilingual data (Consists of Spanish, French etc. along with English)
    Translated non-English language to English using Google translator [4]
  - Two labels: *__label__1* & *__label__2*
    __label__1 $\implies$ 1 & 2 star ratings
    __label__2 $\implies$ 4 & 5 star ratings

Description of Amazon Reviews Dataset

- **Source:** Kaggle
- **Features:**
  - Data is in *fastText*[5] format
  - Training dataset: Consists 36 lakh examples,
    Used 2 lakh for training, 1 lakh for validation
    Test dataset: Consists 4 lakh examples,
    Used 2 lakh for testing
  - Multilingual data (Consists of Spanish, French etc. along with English)
    Translated non-English language to English using Google translator [4]
  - Two labels: *__label__1* & *__label__2*
    __label__1 $\implies$ 1 & 2 star ratings
    __label__2 $\implies$ 4 & 5 star ratings

Description of Amazon Reviews Dataset

- **Source:** Kaggle
- **Features:**
  - Data is in *fastText*[5] format
  - Training dataset: Consists 36 lakh examples,
    Used 2 lakh for training, 1 lakh for validation
    Test dataset: Consists 4 lakh examples,
    Used 2 lakh for testing
  - Multilingual data (Consists of Spanish, French etc. along with English)
    Translated non-English language to English using Google translator [4]
  - Two labels: *__label__1* & *__label__2*
    __label__1 $\implies$ 1 & 2 star ratings
    __label__2 $\implies$ 4 & 5 star ratings

Description of Amazon Reviews Dataset

- **Source:** Kaggle
- **Features:**
  - Data is in *fastText*[5] format
  - Training dataset: Consists 36 lakh examples,
    Used 2 lakh for training, 1 lakh for validation
    Test dataset: Consists 4 lakh examples,
    Used 2 lakh for testing
  - Multilingual data (Consists of Spanish, French etc. along with English)
    Translated non-English language to English using Google translator [4]
  - Two labels: *__label__1* & *__label__2*
    __label__1 $\implies$ 1 & 2 star ratings
    __label__2 $\implies$ 4 & 5 star ratings

Literature Survey

Probabilistic Modelling

- Multinomial Naive Bayes [8]

RNN framework

- LSTM[12]/GRU[2]
- Bidirectional RNN [11]
- Hierarchical RNN [10]

Transformer-based network

- ELMo[9]
- BERT[3] and its variants

Literature Survey

Probabilistic Modelling

- Multinomial Naive Bayes [8]

RNN framework

- LSTM[12]/GRU[2]
- Bidirectional RNN [11]
- Hierarchical RNN [10]

Transformer-based network

- ELMo[9]
- BERT[3] and its variants

Literature Survey

Probabilistic Modelling

- Multinomial Naive Bayes [8]

RNN framework

- LSTM[12]/GRU[2]
- Bidirectional RNN [11]
- Hierarchical RNN [10]

Transformer-based network

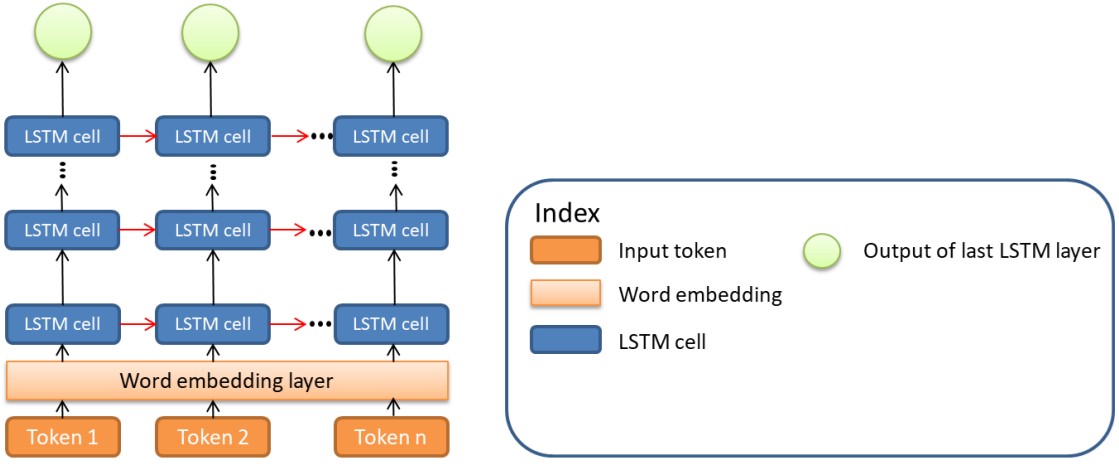
- ELMo[9]
- BERT[3] and its variants

Modelling Approach

Main idea

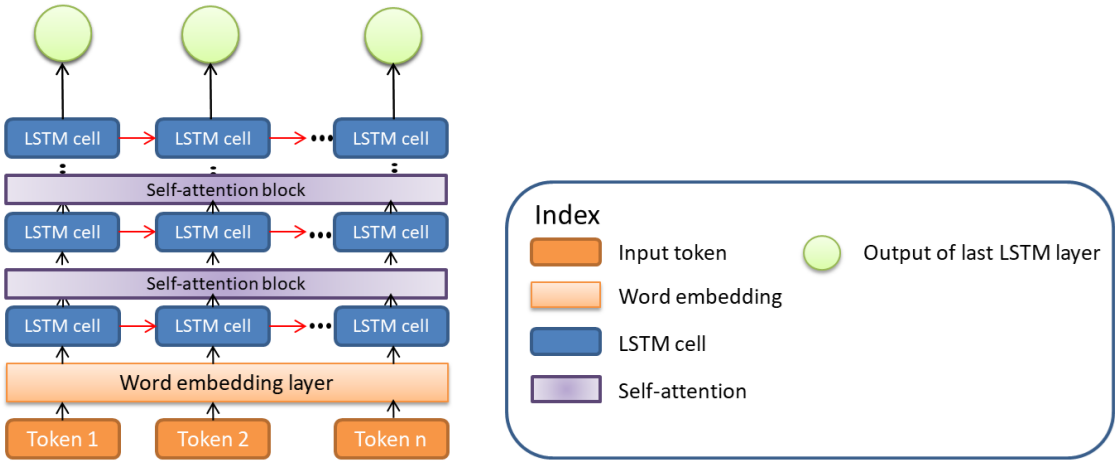
Introducing LSTM autoencoder with LSTM-based classifier network and simultaneously optimizing two objective functions to get better predictions

Modelling Approach



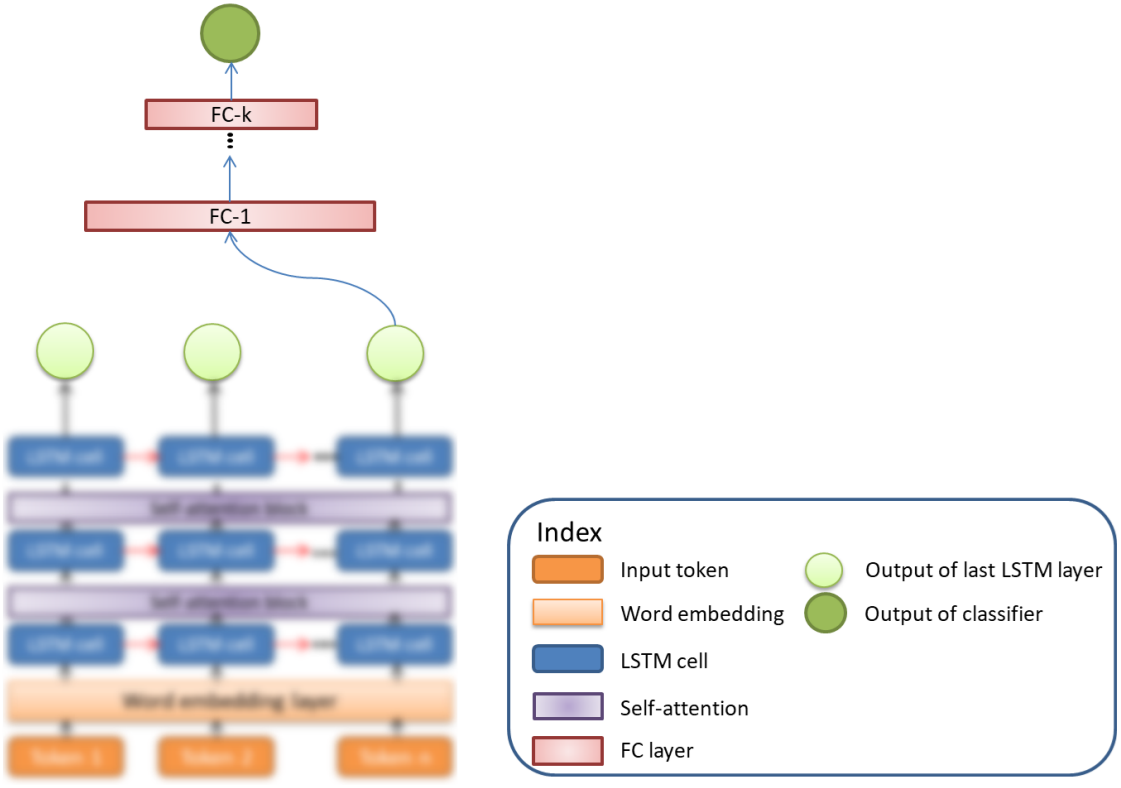
- Encoder: Usual rnn framework with word embedding layer followed by multiple lstm layers — which takes in as input the word tokens and outputs a vector at each time step.

Modelling Approach



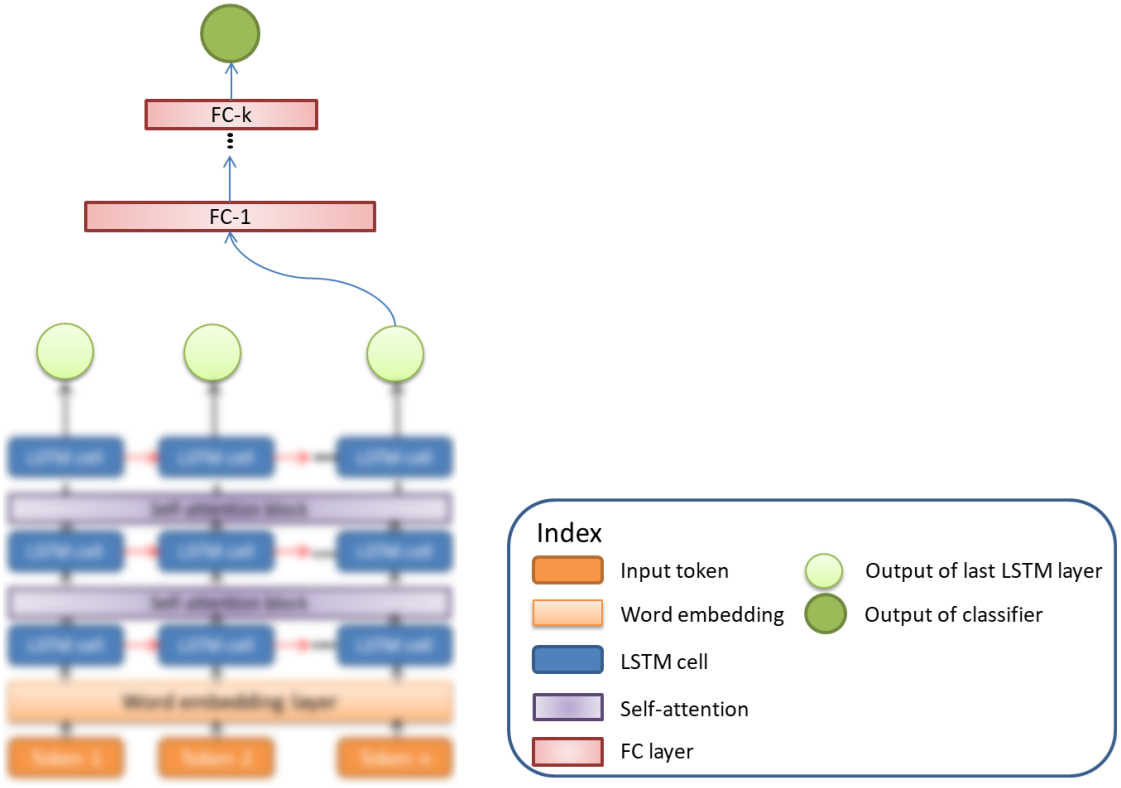
- Encoder: Introduced self-attention block in between two LSTM layers to help the encoder learn better.

Modelling Approach



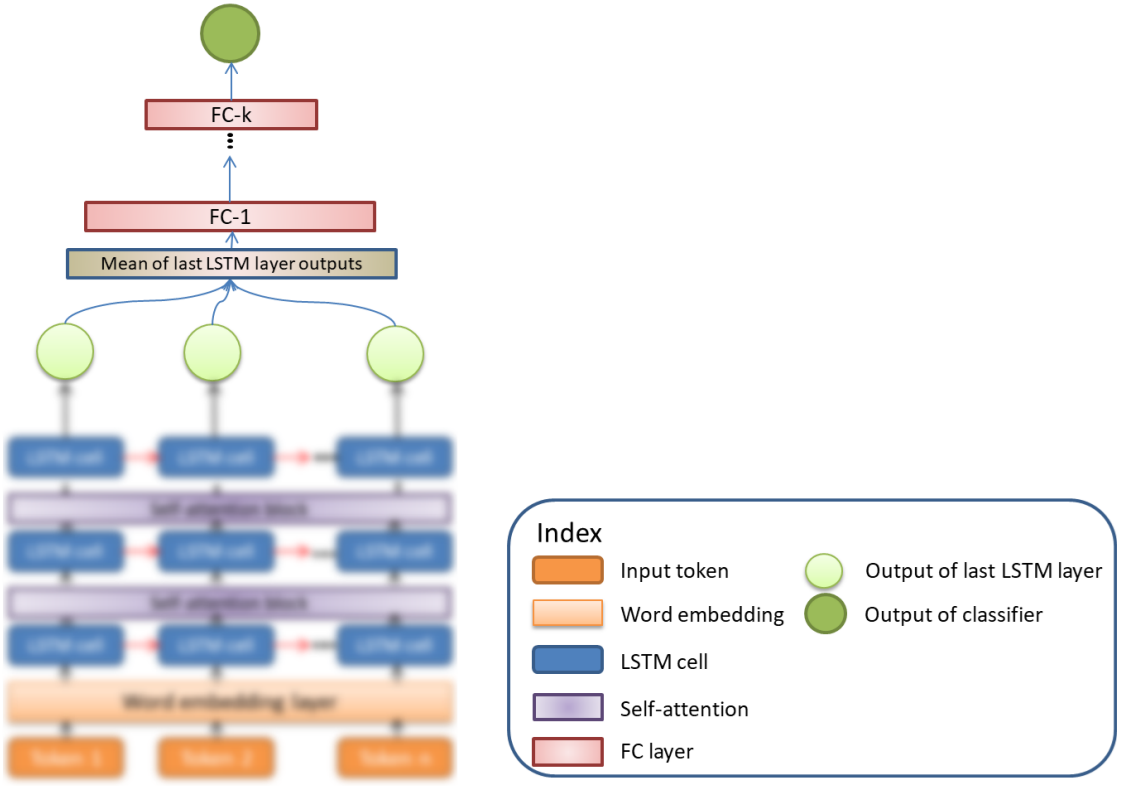
- Classifier: Feed-forward neural network with multiple fully connected layers — which takes in as input the output of last LSTM layer and outputs the probability of a particular example being in a particular class.

Modelling Approach



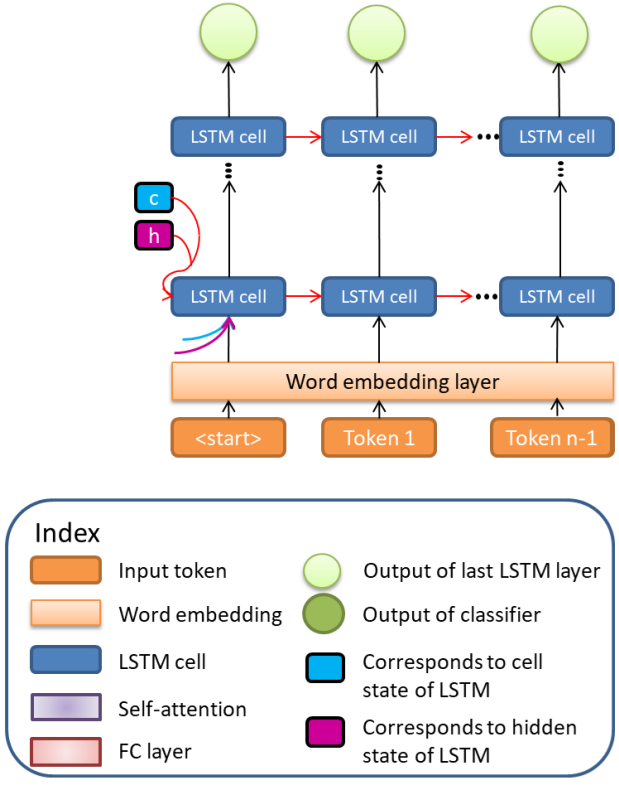
- Classifier: Usually the last time step output of encoder is taken as input.

Modelling Approach



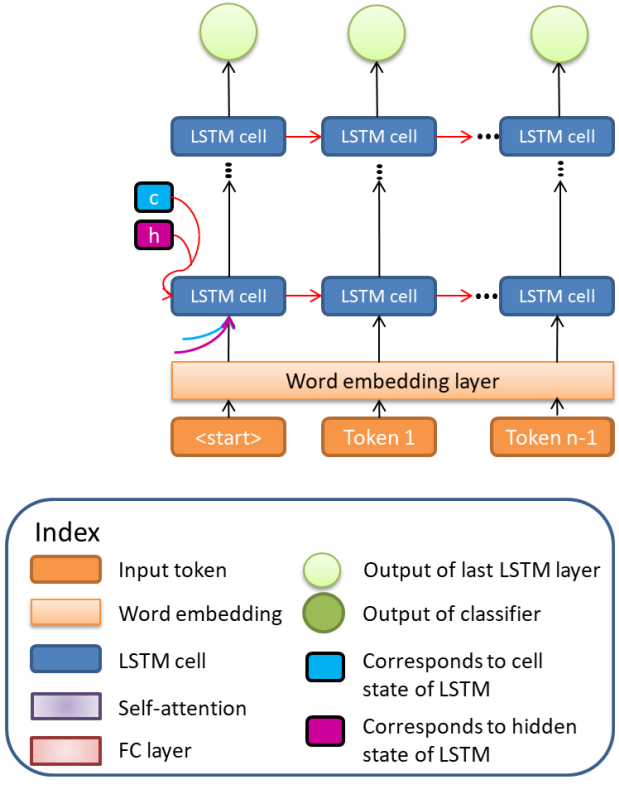
- Classifier: To account for information loss average of all time-step output has been taken as input to classifier.

Modelling Approach



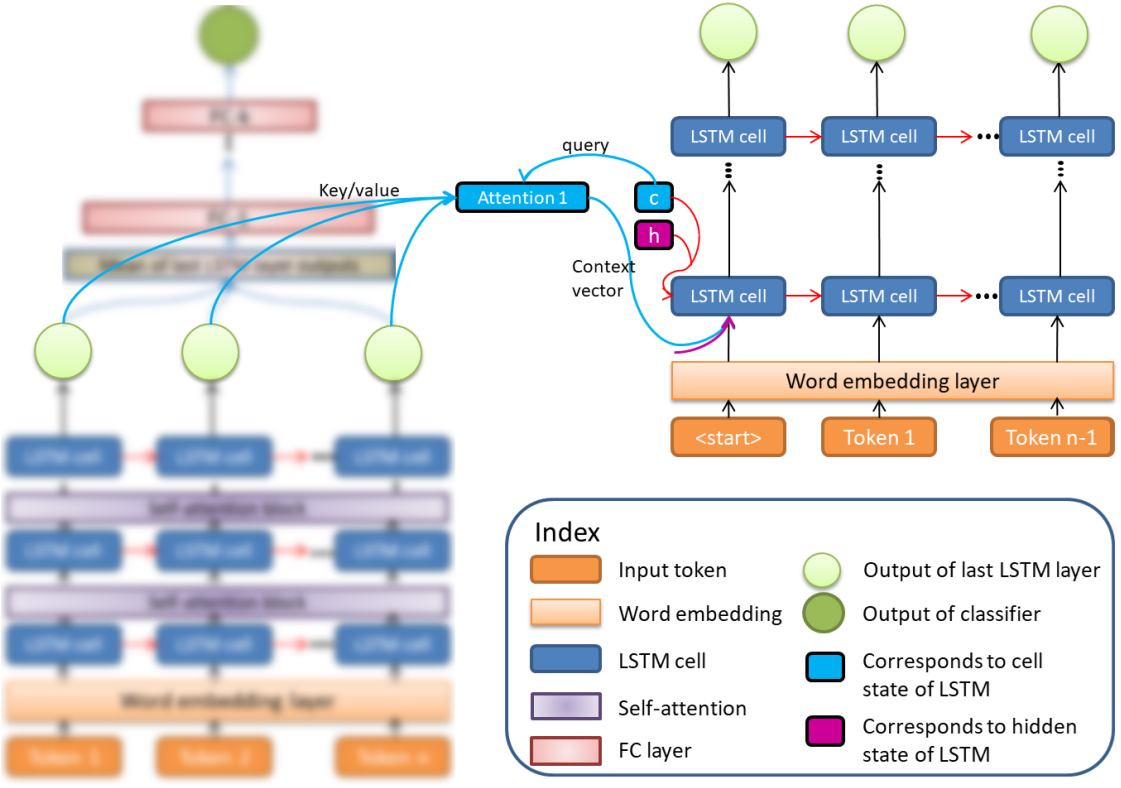
- Decoder: RNN framework with word embedding layer followed by multiple LSTM layers, initiated with <start> token in the first time step.

Modelling Approach



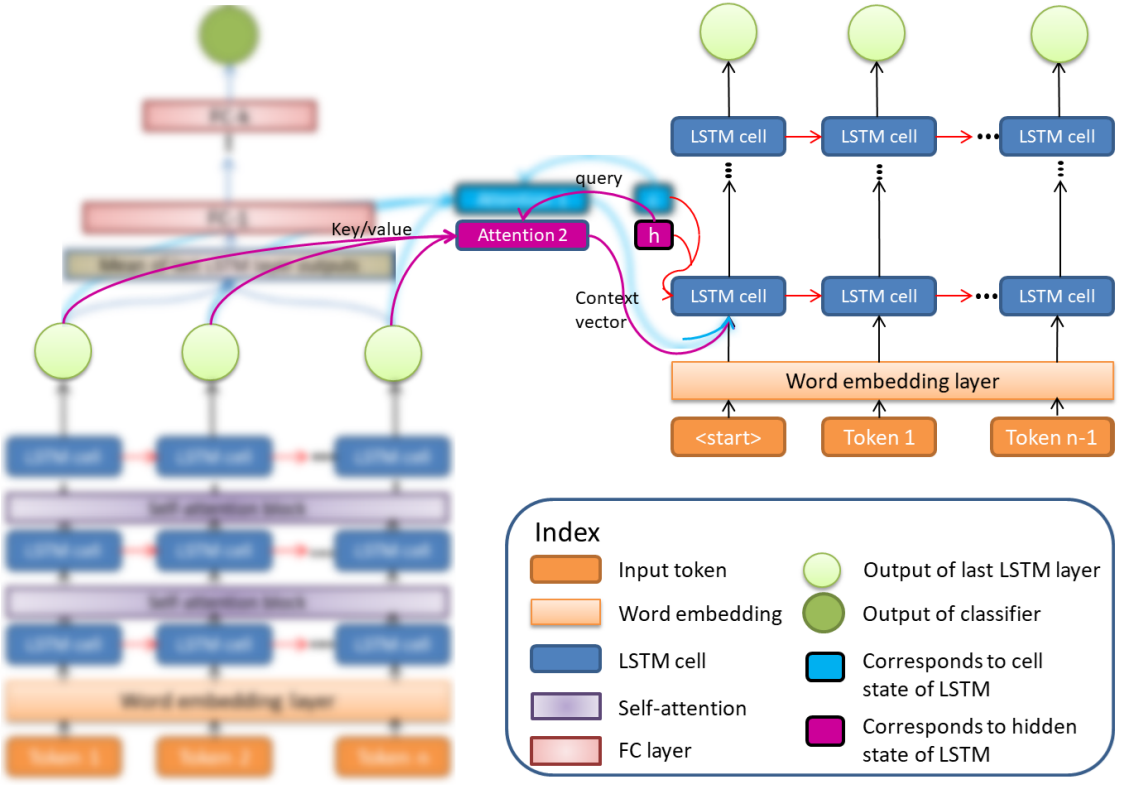
- Decoder: Three inputs to initial LSTM layer — embedded token, context vectors coming from two attention [1] modules where cell state / hidden state of decoder LSTM cells acts as query and outputs of last LSTM layer of encoder act as keys/values.

Modelling Approach



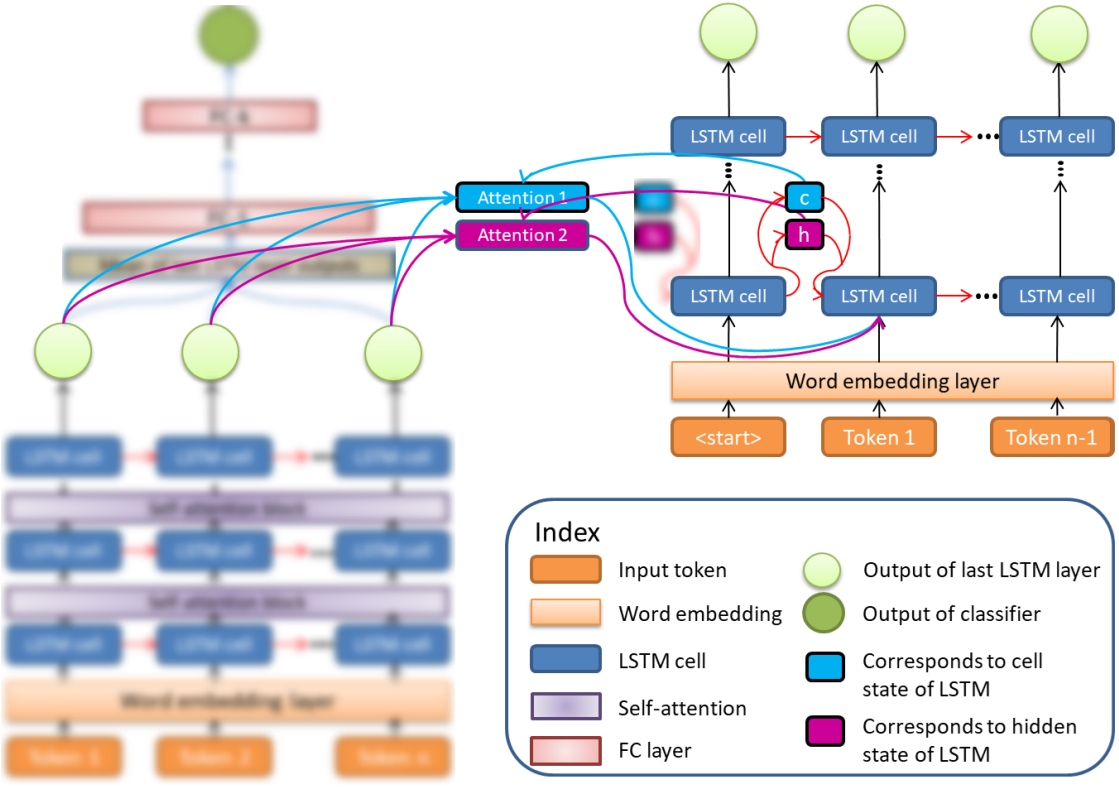
- Decoder: Three inputs to initial LSTM layer — embedded token, context vectors coming from two attention [1] modules where cell state / hidden state of decoder LSTM cells acts as query and outputs of last LSTM layer of encoder act as keys/values.

Modelling Approach



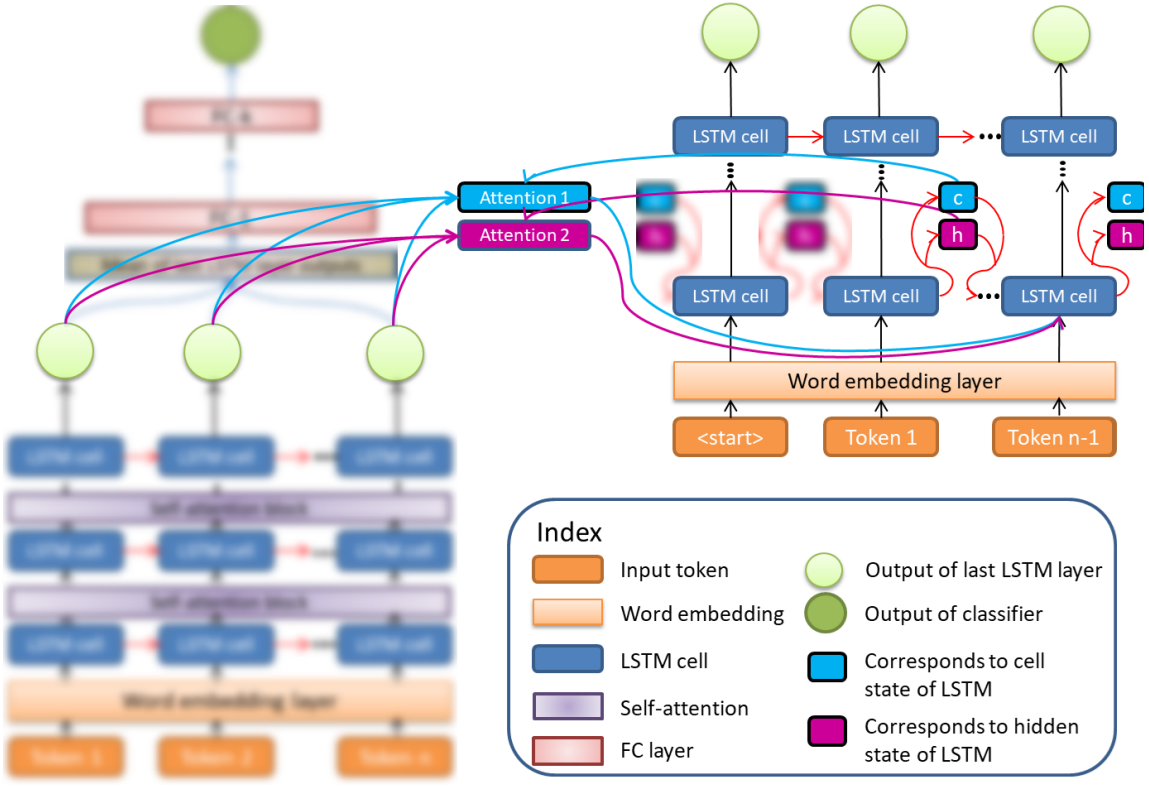
- Decoder: Three inputs to initial LSTM layer — embedded token, context vectors coming from two attention [1] modules where cell state / hidden state of decoder LSTM cells acts as query and outputs of last LSTM layer of encoder act as keys/values.

Modelling Approach



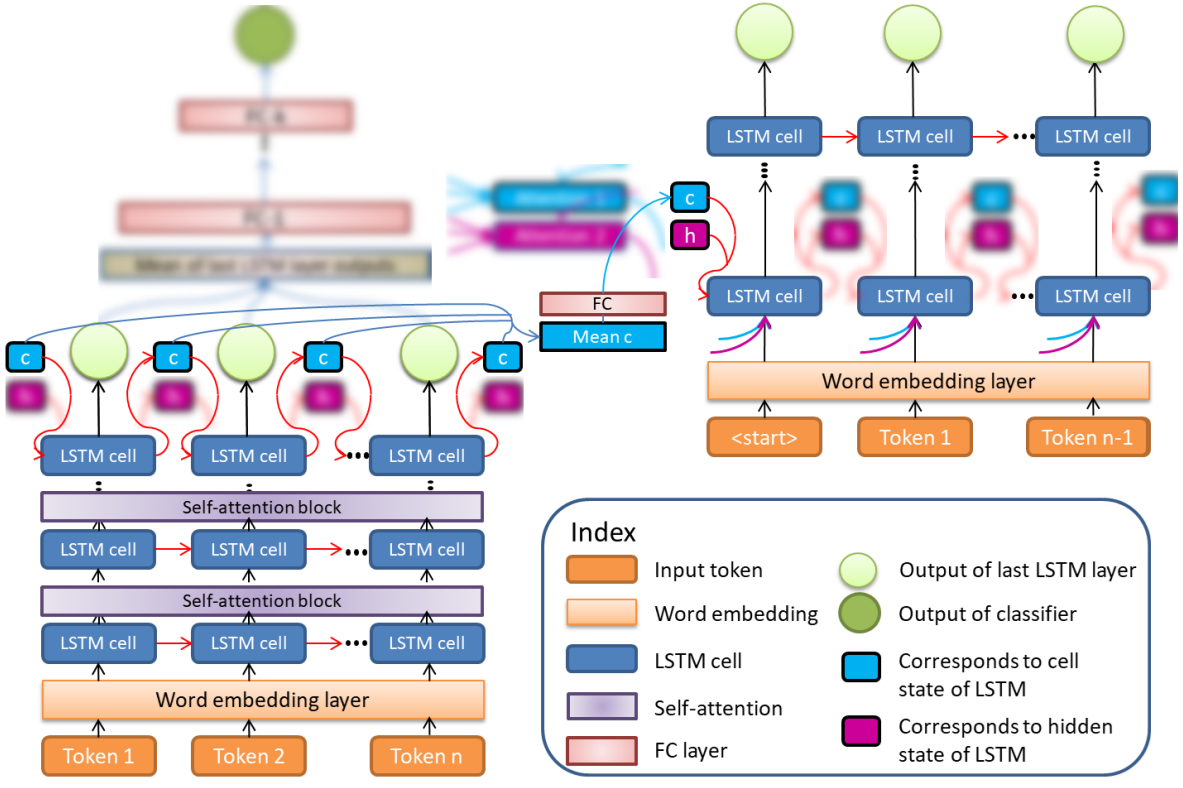
- Decoder: Three inputs to initial LSTM layer — embedded token, context vectors coming from two attention [1] modules where cell state / hidden state of decoder LSTM cells acts as query and outputs of last LSTM layer of encoder act as keys/values.

Modelling Approach



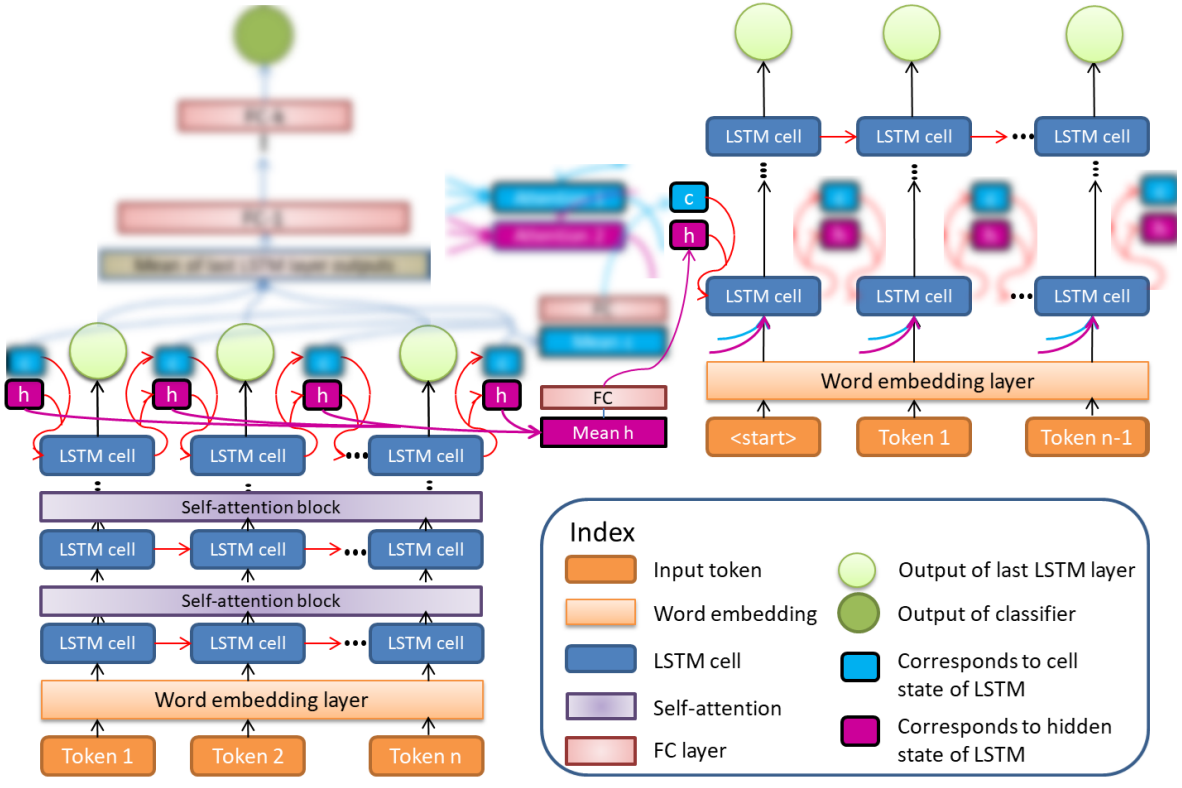
- Decoder: Three inputs to initial LSTM layer — embedded token, context vectors coming from two attention [1] modules where cell state / hidden state of decoder LSTM cells acts as query and outputs of last LSTM layer of encoder act as keys/values.

Modelling Approach



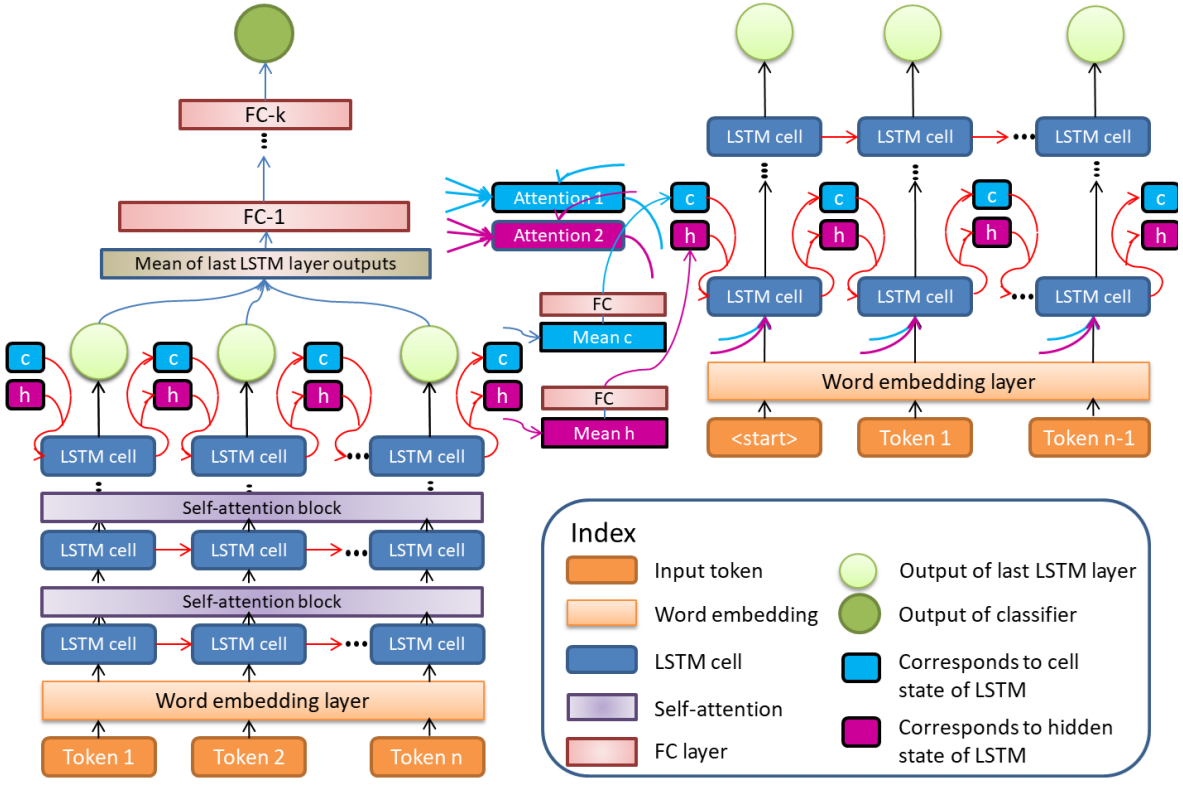
- Decoder: Intermodule cell state / hidden state conversion using one fully connected layer opted instead of random/zero initialization of cell state / hidden state.

Modelling Approach



- Decoder: Intermodule cell state / hidden state conversion using one fully connected layer opted instead of random/zero initialization of cell state / hidden state.

Modelling Approach



Overall architecture

Modelling Approach

Choice of hyperparameters:

- Number of LSTM layers in encoder is greater than that of decoder: Good reconstruction with weaker decoder requires the later layers of encoder to learn something really meaningful. [13]
- SELU activation [6] in classifier: Does away with the need of batch normalization, resulting in fewer parameters and less training time.
- AdamW optimizer [7]: Variant of Adam, but known to generalize well than Adam.
- Dropout of 20% throughout the network: Acts as regularizer and also stabilizes the training process.

Modelling Approach

Choice of hyperparameters:

- Number of LSTM layers in encoder is greater than that of decoder: Good reconstruction with weaker decoder requires the later layers of encoder to learn something really meaningful. [13]
- SELU activation [6] in classifier: Does away with the need of batch normalization, resulting in fewer parameters and less training time.
- AdamW optimizer [7]: Variant of Adam, but known to generalize well than Adam.
- Dropout of 20% throughout the network: Acts as regularizer and also stabilizes the training process.

Modelling Approach

Choice of hyperparameters:

- Number of LSTM layers in encoder is greater than that of decoder: Good reconstruction with weaker decoder requires the later layers of encoder to learn something really meaningful. [13]
- SELU activation [6] in classifier: Does away with the need of batch normalization, resulting in fewer parameters and less training time.
- AdamW optimizer [7]: Variant of Adam, but known to generalize well than Adam.
- Dropout of 20% throughout the network: Acts as regularizer and also stabilizes the training process.

Modelling Approach

Choice of hyperparameters:

- Number of LSTM layers in encoder is greater than that of decoder: Good reconstruction with weaker decoder requires the later layers of encoder to learn something really meaningful. [13]
- SELU activation [6] in classifier: Does away with the need of batch normalization, resulting in fewer parameters and less training time.
- AdamW optimizer [7]: Variant of Adam, but known to generalize well than Adam.
- Dropout of 20% throughout the network: Acts as regularizer and also stabilizes the training process.

Modelling Approach

Training Scheme:

Training is conducted in two phases in every epoch.

1. Phase 1: Train autoencoder with cross-entropy loss averaged over all time-steps but masking out the padded tokens.
2. Phase 2: Train encoder and classifier together with binary cross-entropy loss.

Modelling Approach

Training Scheme:

Training is conducted in two phases in every epoch.

1. Phase 1: Train autoencoder with cross-entropy loss averaged over all time-steps but masking out the padded tokens.
2. Phase 2: Train encoder and classifier together with binary cross-entropy loss.

Modelling Approach

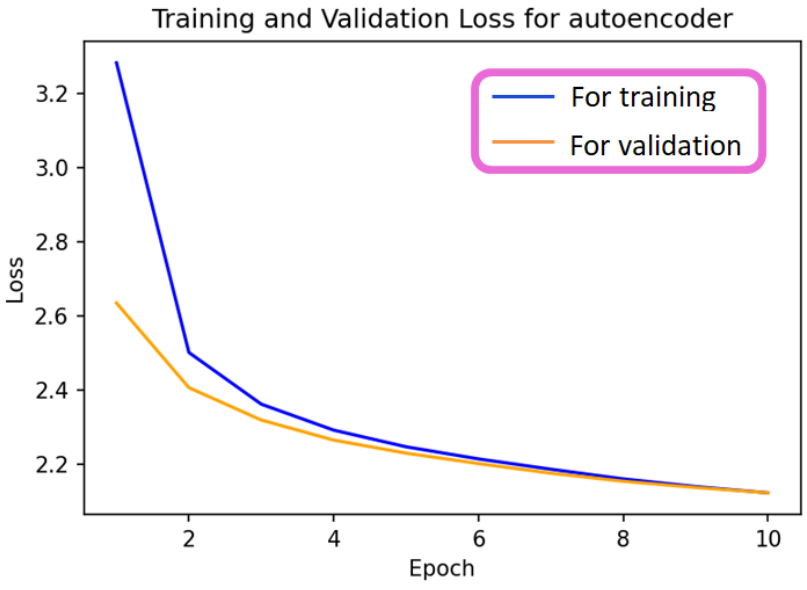
Training Scheme:

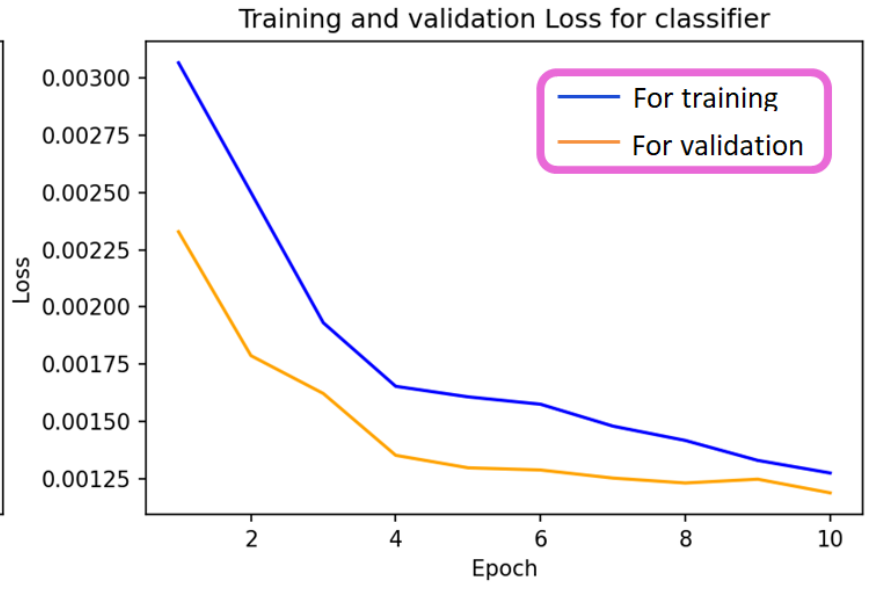
Training is conducted in two phases in every epoch.

1. Phase 1: Train autoencoder with cross-entropy loss averaged over all time-steps but masking out the padded tokens.
2. Phase 2: Train encoder and classifier together with binary cross-entropy loss.

Result

Training and validation loss





Test results

Accuracy: 90.5215%, Precision: 91.1331%, Recall: 89.7154%

Future Work

- Checking if weak decoder helps encoder in gaining better encoded representation of a sequence.
- Checking the performance of the model by tweaking some hyperparameters (for example, changing number of LSTM layers, replacing SELU with other activation functions like SERLU or DELU etc.)
- Trying out bidirectional LSTM layers
- Modifying Attention mechanisms to improve performance

Future Work

- Checking if weak decoder helps encoder in gaining better encoded representation of a sequence.
- Checking the performance of the model by tweaking some hyperparameters (for example, changing number of LSTM layers, replacing SELU with other activation functions like SERLU or DELU etc.)
- Trying out bidirectional LSTM layers
- Modifying Attention mechanisms to improve performance

Future Work

- Checking if weak decoder helps encoder in gaining better encoded representation of a sequence.
- Checking the performance of the model by tweaking some hyperparameters (for example, changing number of LSTM layers, replacing SELU with other activation functions like SERLU or DELU etc.)
- Trying out bidirectional LSTM layers
- Modifying Attention mechanisms to improve performance

Future Work

- Checking if weak decoder helps encoder in gaining better encoded representation of a sequence.
- Checking the performance of the model by tweaking some hyperparameters (for example, changing number of LSTM layers, replacing SELU with other activation functions like SERLU or DELU etc.)
- Trying out bidirectional LSTM layers
- Modifying Attention mechanisms to improve performance

References


Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. *Neural Machine Translation by Jointly Learning to Align and Translate*. 2016. arXiv: 1409.0473 [cs.CL].


Kyunghyun Cho et al. *Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation*. 2014. arXiv: 1406.1078 [cs.CL].


Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].


SuHun Han. *googletrans 3.0.0*. 2020. URL: https://pypi.org/project/googletrans/.

- Armand Joulin et al. “Bag of Tricks for Efficient Text Classification”. In: *arXiv preprint arXiv:1607.01759* (2016).
- Günter Klambauer et al. *Self-Normalizing Neural Networks*. 2017. arXiv: 1706.02515 [cs.LG].
- Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG].
- Muhammad et al. “Multinomial Naive Bayes Classification Model for Sentiment Analysis”. In: (2019). URL: http://paper.ijcsns.org/07_book/201903/20190310.pdf.
- Matthew E. Peters et al. *Deep contextualized word representations*. 2018. arXiv: 1802.05365 [cs.CL].

- Salah et al. “Hierarchical Recurrent Neural Networks for Long-Term Dependencies”. In: (). URL: https://papers.nips.cc/paper/1995/file/c667d53acd899a97a85de0c201ba99be-Paper.pdf.
- Mike Schuster and Kuldip K. Paliwal. “Bidirectional Recurrent Neural Networks”. In: *IEEE Transactions on Signal Processing* (1997).
- Sepp, Jurgen Hochreiter, and Schmidhuber. “Long short-term memory”. In: (1997). URL: https://www.bioinf.jku.at/publications/older/2604.pdf.
- *Should autoencoders really be symmetric?* 2020. URL: https://www.reddit.com/r/MachineLearning/comments/ef1xe8/d_should_autoencoders_really_be_symmetric/.

THANK YOU