

Fig: Model architecture

**Architecture description**: I have designed the above architecture for solving the stated problem. I have used Glove Embeddings as pretrained embeddings ([Wikipedia 2014](http://dumps.wikimedia.org/enwiki/20140102/) + [Gigaword 5](https://catalog.ldc.upenn.edu/LDC2011T07) (6B tokens, 400K vocab, uncased, 50d, 100d, 200d, & 300d vectors, 822 MB download)). The embedding dimensions are 300.

The Bidirectional GRU is used as an encoder which encodes the given sentences. The encoder produced hidden state (dimensions of hidden state is 20). These hidden states are passed through linear layer 1 consisting of 20 neurons along with ReLU non linearity. The linear layer 2 produces a scalar score for each hidden state and these scores are passed through a SoftMax layer to form weights for each hidden state. Then the model produces a weighted average of the hidden state represented as H in the image. Now the model concatenates the sentiment of the sentence along with the normalized count of the columns 9 to 13 given in the dataset. Sentiment consists of polarity and subjectivity. Now this combined vector is passed through a linear layer 3(hidden state dimensions +7 neurons) with ReLU non linearity and then through linear layer 4(6 neurons) and lastly through SoftMax layer which produces the probability of all the class (in our case 6 classes). Now model apply argmax to the probabilities and output the predicted class.

The highest accuracy for multi-class classification is 25% and that for binary classification is 77%.

I have considered half-true, mostly true and true as true and pants on fire, false, mostly false as false for the binary classification. Seeing the accuracy for both i.e. for multi-class and binary classification it is evident that my model is not able to distinguish properly between half-true, mostly true and true and also between pants on fire, false, mostly false in the case of multi-class classification.

I have used the same model for calculating the binary classification accuracy. The method can be seen in the translator.py.

Preprocessing: - The sentences in the column 3,14 and 15 are concatenated using two separation tokens(<sep1>, <sep2>) like this “sentence from column 3 <sep1> sentence from column 14 <sep2> sentence from column 15”. Also, sentiment for this concatenated sentence is calculated and stored.

True casing was applied to these sentences in 3,14 and 15 columns and punctuations were removed. Using the NLTK library stop words were also removed from these sentences. Then the sentences were lemmatized using TextBlob library also using the same library the sentiments were calculated.

The libraries that were used are:

1. pytorch
2. nltk
3. numpy
4. textblob
5. sklearn
6. pickle
7. genism
8. pandas

The code is based on "Unsupervised Neural Machine Translation" by Mikel artetxe <https://github.com/artetxem/undreamt>

And you will find this line “This code is heavily modified version of the code used in the paper "Unsupervised Neural Machine Translation" by Mikel artetxe <https://github.com/artetxem/undreamt>” in the code if that code is from artetxe.