RENNARD VIRGILE
Virgile@Rennard.org

Lab session # 3
ALTEGRAD 2019

11/01/19

1 Question 1

Let's first compute it with a p_i very close to the truth, equal to 0.99

$$logloss = -(1ln(0.99)) = -ln(0.99) = 0.0100 \tag{1}$$

then let's compute it with a p_i being a very incorrect prediction, equal to 0.1

$$logloss = -(1ln(0.1)) = -ln(0.1) = 2.3 \tag{2}$$

and finally with an unsure p_i, let's say a 0.5

$$logloss = -(1ln(0.5)) = -ln(0.5) = 0.69 \tag{3}$$

We can see that the binary crossentropy will effectively take into account the ”distance” from our correct prediction, unlike the '0-1' loss. This will effectively penalize a heavily incorrect prediction harder than an unsure incorrect prediction.

2 Question 2

Clearly, the missing value is -3, which is simply $0 \times 0 + 2 \times -1 + 2 \times -1 + 0 \times 1 + 0 \times 1 + 0 \times 2 + 1 = -3$

3 Question 3

Here we use the softmax function which to a x_i associates $\frac{e^{x_i}}{\sum_{j=1}^{K} e^{x} x_j}$ which is commonly used for multi class classification problems. Here, as we work on a binary classification problem, we can simply use the $sigmoid$ function. Instead of having as an output the number of different classes which we are interested in, we will only get one unit on the prediction layer, representing simply the probability of $y_i = 1$. The Softmax uses one hot representation of labels, contrary to the sigmoid

4 Question 4

Our model is defined by different layers, the first with trainable parameters is the Embedding layer, where every word in the vocabulary has an embedding of size d, and those are all parameters. We therefore have on the first embedding layer a $|V| * d$ where $|V|$ is the length of the dictionary. Following this, we get a Convolution layer that goes on two different branches. The layer has parameters $|nf|$ the number of filters, as well as $|lf|$ the filters length. It has therefore a number $|lf| * |nf| * |d|$ parameters. After this, the Max Pool layer has no parameters. Finally, the Dense layer has $|lf|$ parameters We have therefore a number of $|V| * d + |nf| * |lf| * |d| + |lf|$

5 Question 5

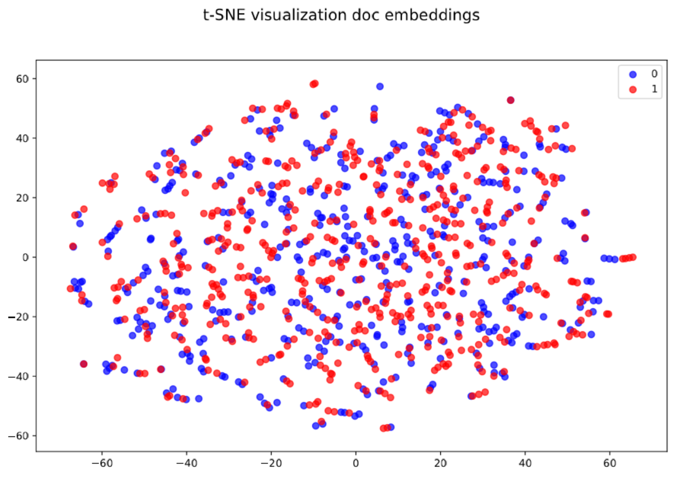

Figure 1: Before the Training

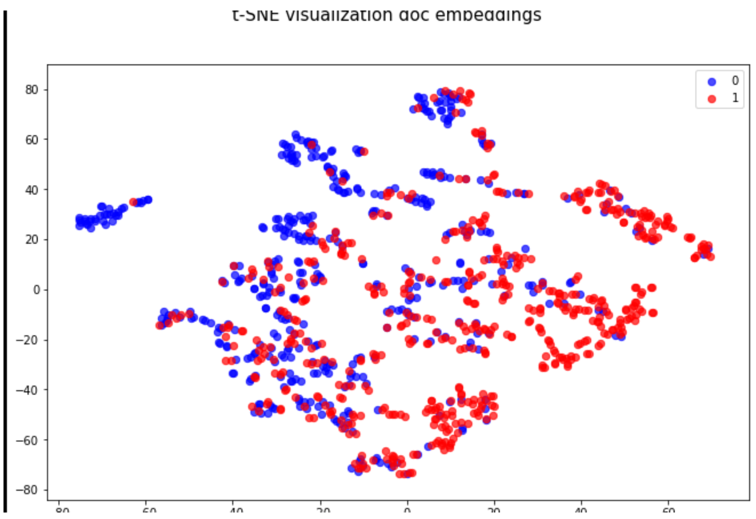

Figure 2: After the training (1 Epochs)

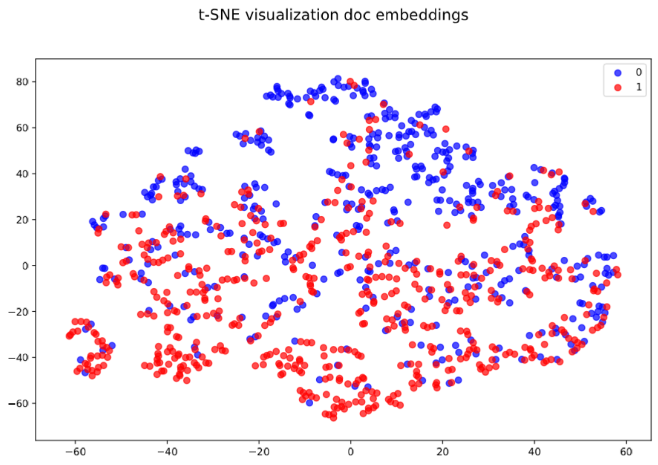

Figure 3: After the training (4 Epochs)

Here, we're using the T-SNE method, who is, as we've already seen in the last practical assignment, a way to project our data into a 2D space in order to better visualize what happened during the training. We can see on the figure 1 that before the training, the reviews which were liked and disliked are completely randomly assigned to parts of the 2D plane. This is normal as we haven't worked on the data yet. After only one Epochs, we can see on o2, that the Liked/Disliked have already started to clump together. After 4 Epochs, we can see on 3 that the Liked and Disliked reviews have been separated pretty well.

6 Question 6

In [1], they explain what Saliency maps are, and give context as to why they are used in the case of Image classification. In this case. They explain that the magnitude of the derivative will show which pixels needs to be changed the least to affect the class score the most. This will effectively give us the location of the object which we try to find.
In our case, we do not work on Image classification, but on Sentiment classification, with words instead of images. The resulting saliency map is given in the figure below, for both a training of 4 epochs and less

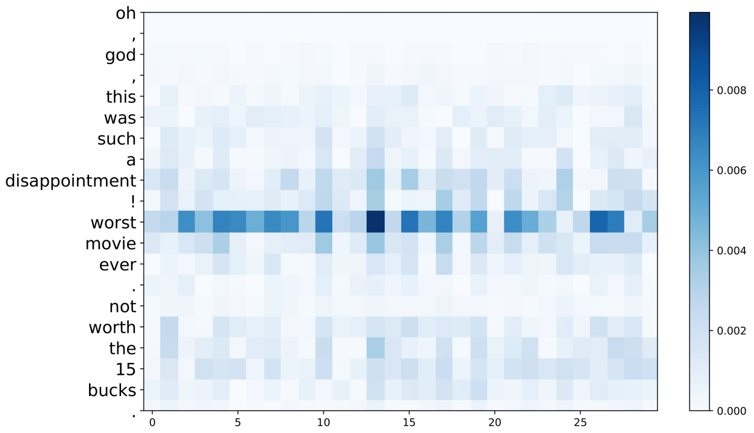

Figure 4: After some training

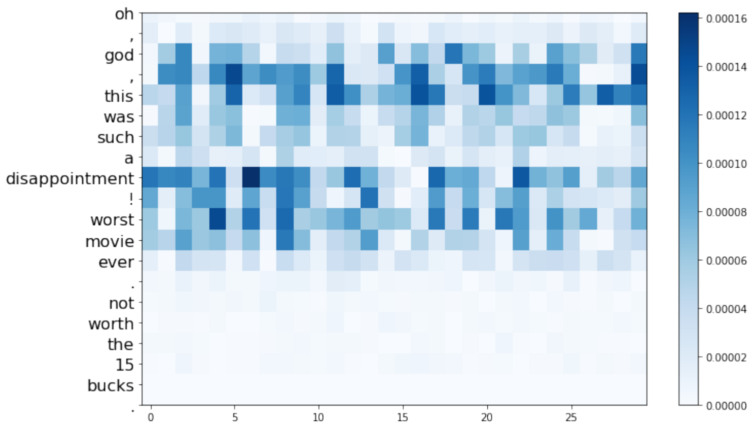

Figure 5: After some more training (4 Epochs)

By the same token, we can see that ”disappointment”, ”worst” are the words in the review which needs to be changed the least in order to change the classification the most. This seems reasonable as disappointment and worst are words that accurately give whether the review is positive or negative. We can see that overfitting the data will make it so that ”,” is a great indicator of wether or not the movie was good, which is not a good indicator

7 Question 7

CNN’s are by construction good for extracting local as well as position invariant features, and are thus a good choice for Sentiment analysis, as the result is often found in some keywords in the phrase, as seen in 5. Moreover, they are faster than the alternatives. They are however relatively weak for determining long range dependencies, and would be an exceptionally bad alternative to LSTM’s for things like machine translation. Some of this points are developped in [2]. Another weak points of CNN’s are that the sequence must all be of same length, unlike for RNN’s, which makes us have to pad the data. More generally, CNN also need a bigger dataset

References

[1] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

[2] Wenpeng Yin, Katharina Kann, Mo Yu, and Hinrich Schütze. Comparative study of cnn and rnn for natural language processing, 2017. cite arxiv:1702.01923Comment: 7 pages, 11 figures.