Arnaud Massenet
arnaudyoh@gmail.com

Lab session # 2
ALTEGRAD 2019

02/12/19

Question 1

We define "confident correct prediction"/"unsure correct prediction"/"strongly incorrect prediction" for a true label 1 as $p_i = 0.95/p_i = 0.6/p_i = 0.05$ respectively.

$$logloss = -\log 0.95 \approx 0.0222$$

$$logloss = -\log 0.6 \approx 0.2218$$

$$logloss = -\log 0.05 \approx 1.301$$

From the results above, we observe that the loss increases exponentially when p_i approaches 0. From this, we can assume that the same applies when we have a true label 0 and p_i approaches 1.

Question 2

To get the second output we do the following:

- Our inputs are the sub-inputs $[0, 2, 2]$ and $[1, 1, 2]$
- We apply the first filter $[0, -1, -1]$ to $[0, 2, 2]$, we get $[0, -2, -2]$
- We apply the second filter $[0, 0, 0]$ to $[1, 1, 2]$, we get $[0, 0, 0]$
- We sum up the elements together to get -4
- We add the bias and get -3.

So the missing value is -3.

Question 3

The other widely used activation function for a classification task is the sigmoid function.

If we are doing binary classification (class 1 and 0) we usually have two or one units in the output layer. We can use one unit to represent the probability $p_1 \in [0, 1]$ of outputting class 1 so the probability of class 0 would be $1-p_1$. If we use two output neurons, we can assign the probability of the two class p_1 and p_0 to each neuron.

If we are using softmax, then we will need to use two output units. Recall that the softmax function is defined as $\frac{e^{x_k}}{\sum_{i=1}^{N} e^{x_i}}$. This entails that we cannot use a single unit as we would always output a probability 1. If we are using the sigmoid function, we can instead use a single unit. This is because the sigmoid function $\sigma(x) = \frac{1}{1+e^x}$ needs only one input to be used, as such, if we use the sigmoid function with a single unit, we will need to train less parameters to achieve the same results than if we use softmax with two output units.

Question 4

Recall that we define n_h the number of filters, h the size of the filters. For each filter f_i we have an associated bias b_i. We assume that we are using maxpooling on all filters and a sigmoid activation function at the end of our branch.

For each filter we have $h * d + 1$ parameters, the $+1$ is for the bias. So for all filters we have $n_h * (h * d + 1)$ parameters to learn. Note that we do not need to train any parameters for the pooling process, the activation function or the output unit.

Thus, we will have to train a total of $n_h * (h * d + 1)$ parameters.

Question 5

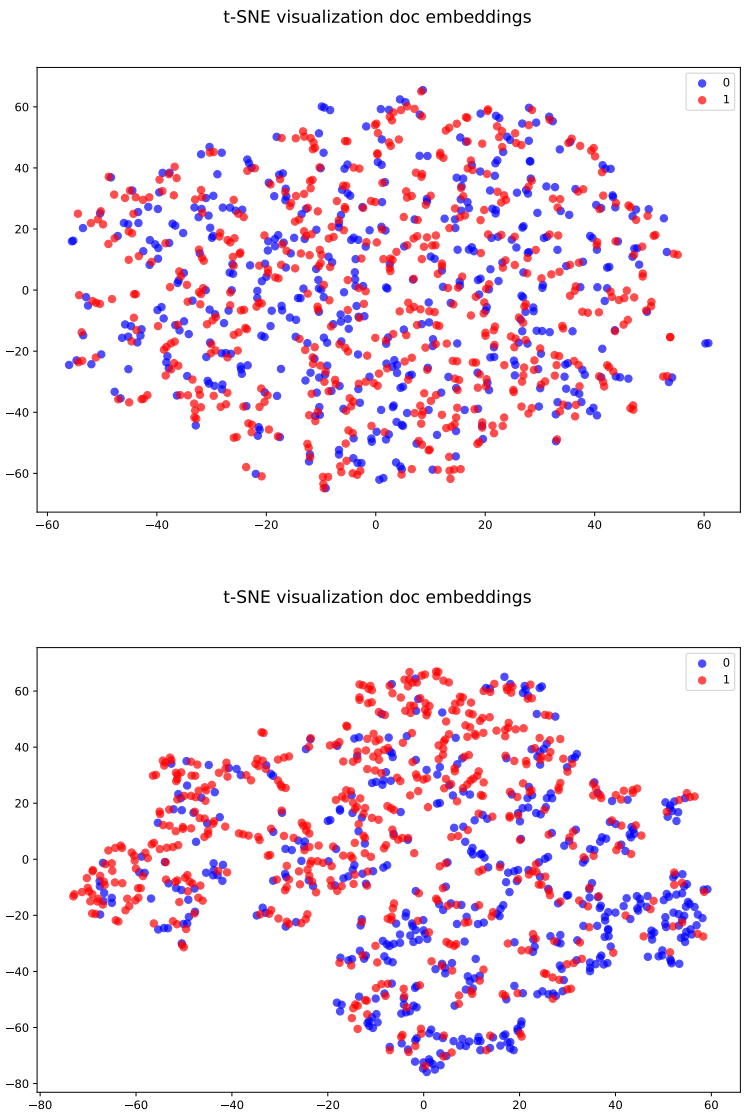

Figure 1: Comparison of the embedding space before and after the training of the network.

We can observe that after the training of the network, we the distribution of the two classes over the embedding space have shifted away. Indeed, in Fig. 1 we can see on the top figure , which represents vector space before the training, that the words are more or less uniformly distributed over the whole space. On the contrary, in the bottom figure, we can see that words of class 1 are usually found on the top right corner while class 0 word are found on the left hand corner. This result shows that the training of the network created a transformation of the space that allow to better separate words of different classes.

Question 6

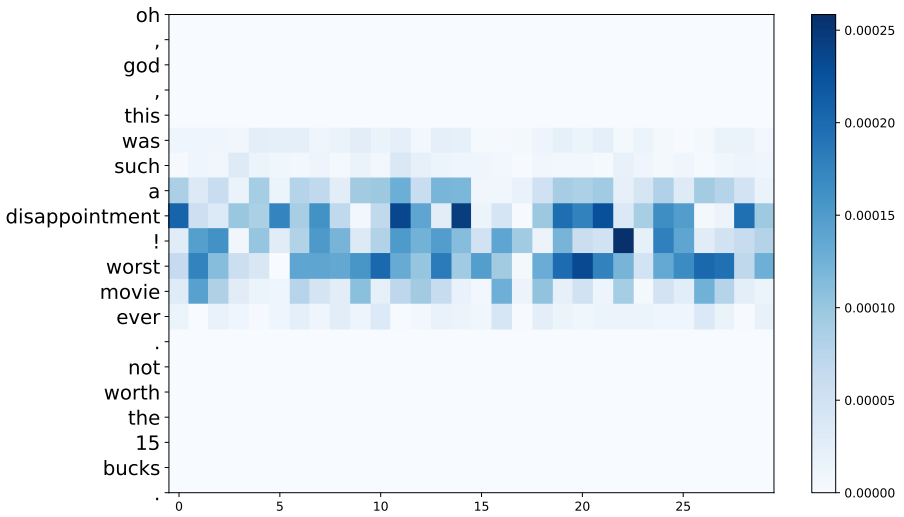

Figure 2: Saliency map for the sentence: *Oh , god , this was such a disappointment ! Worst movie ever . Not worth the 15 bucks .*

In Fig. 2 we can observe that the gradients are higher around the tokens ”!” and ”worst”. This is expected since we obtained similar results when we computed the norms of the region embeddings. The regions with higher norms were those that included the words with high saliency as shown in Fig. 2. Also, the results are not very surprising if we follow our intuition since ”worst movie” and ”disappointment” clearly highlight the negative sentiment of the review.

Question 7

While CNNs allow to capture words and their surrounding context it cannot link words that are far from each others. For example, if we mention a person a the beginning of a book, a CNN model will struggle to recognize that words such as ”He/She” might refer to that person later in the book. Thus, CNNs struggle to capture long-distance relations between words.