Del Gallo Adriano
delgallo.adriano@gmail.com

1 Question 1

Our label is $y_i = 1$.
For a confident correct prediction:

$$logloss = -\log(p_i) \xrightarrow{p_i \to 1} 0 \tag{1}$$

For an unsure correct prediction:

$$logloss = -\log(p_i) \xrightarrow{p_i \to \frac{1}{2}^+} log(2) \tag{2}$$

For a strong incorrect prediction:

$$logloss = -\log(p_i) \xrightarrow{p_i \to 0} \infty \tag{3}$$

We notice that the loss is a decreasing function of p_i. This means that when the predictions are accurate, the loss will penalise uncertainty. Also the penalisation is the highest for incorrect predictions (which translates by $p_i < 1/2$) and will tend to $+\infty$ when the prediction is is strongly incorrect.

2 Question 2

We are looking for the second output after applying the filter W_0 which has depth 2. We note W_{01} (resp. W_02) the first (resp. second) layer of the filter W_0, and I_1 (resp. I_2) the first (resp. second) layer of the input. The output is :

$$\begin{aligned} O_2 &= I[2:2+h,:] * W_0 + b_0 \\ &= I_1[2:2+h] * W_{01} + I_2[2:2+h] * W_{02} + b_0 \\ &= (0 \times 0 + 2 \times -1 + 2 \times -1) + (1 \times 0 + 1 \times 0 + 2 \times 0) + 1 = -3 \end{aligned}$$

The missing value is -3.

3 Question 3

We could use a sigmoid to do a logistic regression on the extracted features. In this case there is only one unit in the final layer. We affect class 1 if the output is greater than 0.5, class 0 otherwise.

4 Question 4

The word embeddings count for sd parameters. For a filter of size h used to sample a word vector of length d, there is hd convolution parameters (the filter must be of depth d to preserve the word embedding) and 1 bias. For n_f filters in one branch this gives $n_f(hd+1)$ parameters. After MaxPooling layer and Concatenation layer we have a vector of size n_f as input of a fully connected layer with 2 output units. This counts for $2(n_f+1)$ parameters. We get the following formula for the number of parameters p:

$$p(s,d,h,n_f) = sd + hd + n_f(hd+1) + 2(n_f+1)$$

5 Question 5

At first reviews are randomly scattered since the embeddings have not yet learned to represent the documents. However, after training we see the formation of clusters in the embedding space. The reviews are regrouped by label and the document embedding space learns regions of positive and negative reviews, making the two classes practically linearly separable.
We notice that some regions of the embedding space are populated by clusters of a unique class of documents, representing regions of confidently positive (resp. confidently negative reviews). Regions where both classes exist are regions where the review does not lean towards a strong opinion.

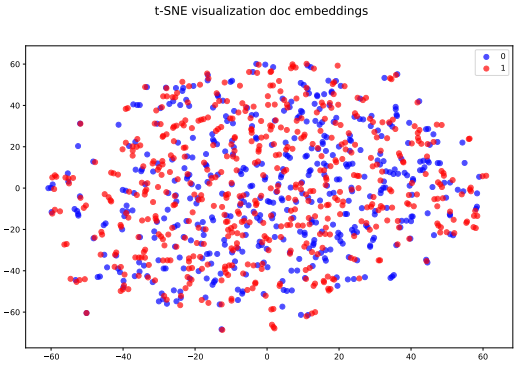

Figure 1: t-SNE of document embeddings before training.

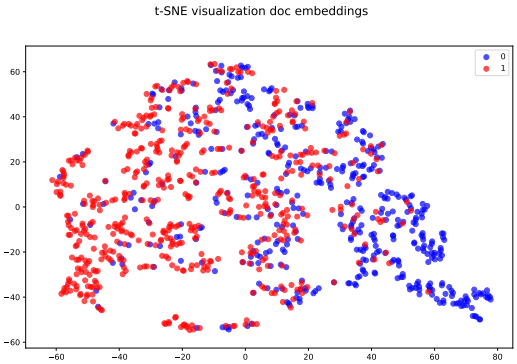

Figure 2: t-SNE of document embeddings after training.

6 Question 6

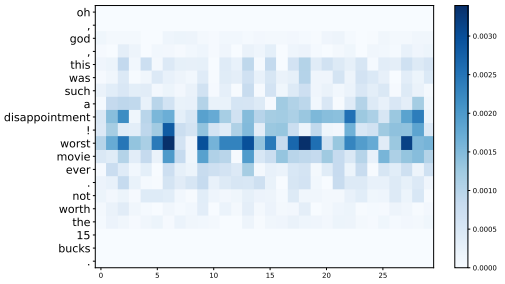

Figure 3: Saliency map of our review

We see that in our review, the words 'worst' and 'disappointment' have the highest saliency. This corresponds to our intuition of which words will influence the class the most. The other words are not very helpful for determining the class which is reflected by low saliency. This shows that this model relies on time invariant keywords to perform classification.

7 Question 7

The CNN model is based on having receptive fields of different sizes, however within the receptive fields, word order is not taken into account (same words in different orders might have a different meaning). Also since we use windows to sample our text, we cannot take into account long term dependencies between words if they occur within two different time frames. For tasks that require long term dependencies and word order preservation, one might prefer the use of RNN.