Maxime Lepeytre
maxime.lepeytre@orange.fr

Lab session # 1
ALTEGRAD 2020

11/24/20

1 Question 1

Compute the partial derivatives of the loss w.r.t one positive example and one negative example, $\frac{\partial L}{\partial w_{c^+}}$ and $\frac{\partial L}{\partial w_{c^-}}$.

We know that:

$$L(t, C_t^+, C_t^-) = \sum_{c \in C_t^+} \log(1 + e^{-w_c \cdot w_t}) + \sum_{c \in C_t^-} \log(1 + e^{w_c \cdot w_t}) \tag{1}$$

Computation of $\frac{\partial L}{\partial w_{c^+}}$:

$$\frac{\partial L}{\partial w_{c^+}} = \frac{\partial \sum_{c \in C_t^+} \log(1 + e^{-w_c \cdot w_t})}{\partial w_{c^+}} \tag{2}$$

And we have,

$$\frac{\partial \log(1 + e^{-w_c \cdot w_t})}{\partial w_{c^+}} = \frac{-w_t}{1 + e^- w_c \cdot w_t} \tag{3}$$

So with (2) and (3),

$$\frac{\partial L}{\partial w_{c^+}} = (\frac{-w_t}{1 + e^{w_{c_1} \cdot w_t}}, \frac{-w_t}{1 + e^{w_{c_2} \cdot w_t}}, ..., \frac{-w_t}{1 + e^{w_{c_n} \cdot w_t}}) \tag{4}$$

We use the same process for the computation of Computation of $\frac{\partial L}{\partial w_t}$:

$$\frac{\partial L}{\partial w_{c^-}} = (\frac{w_t}{1 + e^{-w_{c_1} \cdot w_t}}, \frac{w_t}{1 + e^{-w_{c_2} \cdot w_t}}, ..., \frac{w_t}{1 + e^{-w_{c_n} \cdot w_t}}) \tag{5}$$

2 Question 2

Compute the partial derivative of the loss w.r.t. the target word, $\frac{\partial L}{\partial w_t}$

Computation of $\frac{\partial L}{\partial w_{c^+}}$ with (1) and linearity of differentiation:

$$\frac{\partial L}{\partial w_t} = \sum_{c \in C_t^+} \frac{-w_{c^+} e^{-w_{c^+} w_t}}{1 + e^{-w_{c^+} w_t}} + \sum_{c \in C_t^-} \frac{w_{c^-} e^{w_{c^-} w_t}}{1 + e^{w_{c^-} w_t}} \tag{6}$$

3 Question 3

Observe and interpret your similarity values and your plot. What can you say about the embedding space?

We computed the distance between two words, for that we use my_cos_ similarity:

1. $my\_cos\_similarity(\text{"}movie\text{"}, \text{"}banana\text{"}) = 0.4415$
2. $my\_cos\_similarity(\text{"}movie\text{"}, \text{"}film\text{"}) = 0.9923$
3. $my\_cos\_similarity(\text{"}movie\text{"}, \text{"}history\text{"}) = 0.9812$
4. $my\_cos\_similarity(\text{"}movie\text{"}, \text{"}children\text{"}) = 0.9785$
5. $my\_cos\_similarity(\text{"}movie\text{"}, \text{"}others\text{"}) = 0.9818$
6. $my\_cos\_similarity(\text{"}children\text{"}, \text{"}others\text{"}) = 0.9744$

With this list of distances, it is obvious that this representation is focused on film reviews. The meaning of ”film” and ”other” is different, but in the context of film review, these two words are close. For example, in a review, one can find ”other actors”, ”other films by this filmmaker”.....

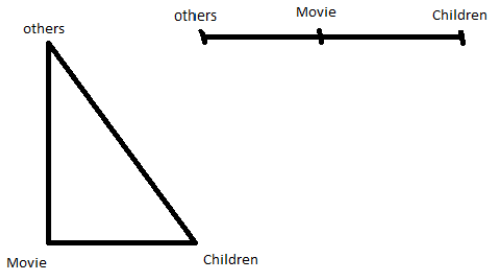

Figure 1: Center on Movie

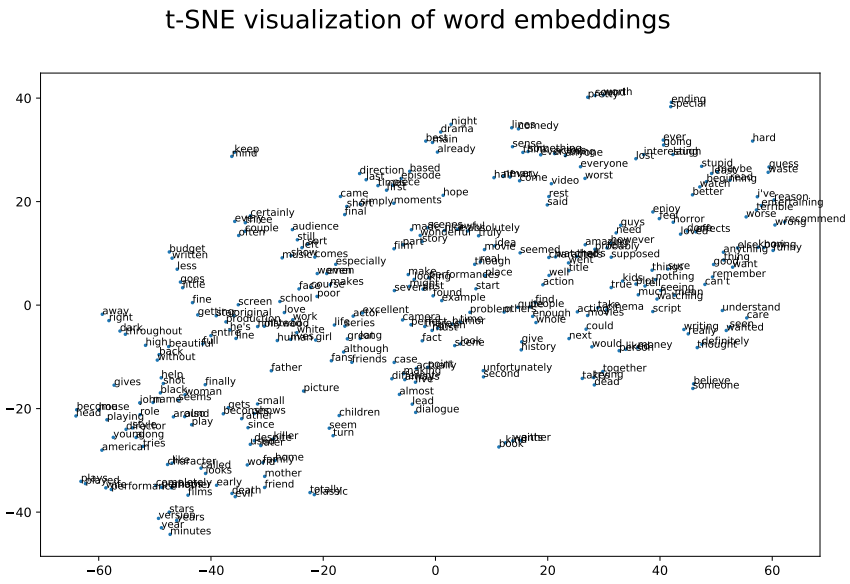

Figure 2: Our model

W, word embedding matrix is focus on film point of view.

Why my cos similarity(”children”, ”others”) = 0.9744 ? As everything is compare to movie, film... if two words are close to movie then by consequence the distance between these two words is small. As we can see on fig (1). Movie, film is the center of our representation so movie can be the center of a segment [others, children] or a corner of the triangle (children,movie,others).

W matrix is represented on fig (2) through PCA and t-SNE projection.

This representation is a 2D projection of our W embedding matrix.

4 Question 4

What changes should we make to our pipeline (preprocessing and training) to learn document vectors jointly with the word vectors? Base your answer on [1].

The main idea under this article [1] is to outperform bag-of-word approach by adding context with a paragraph matrix. An issue in NLP models is to take into account the context.

Using this idea, our new model can be:

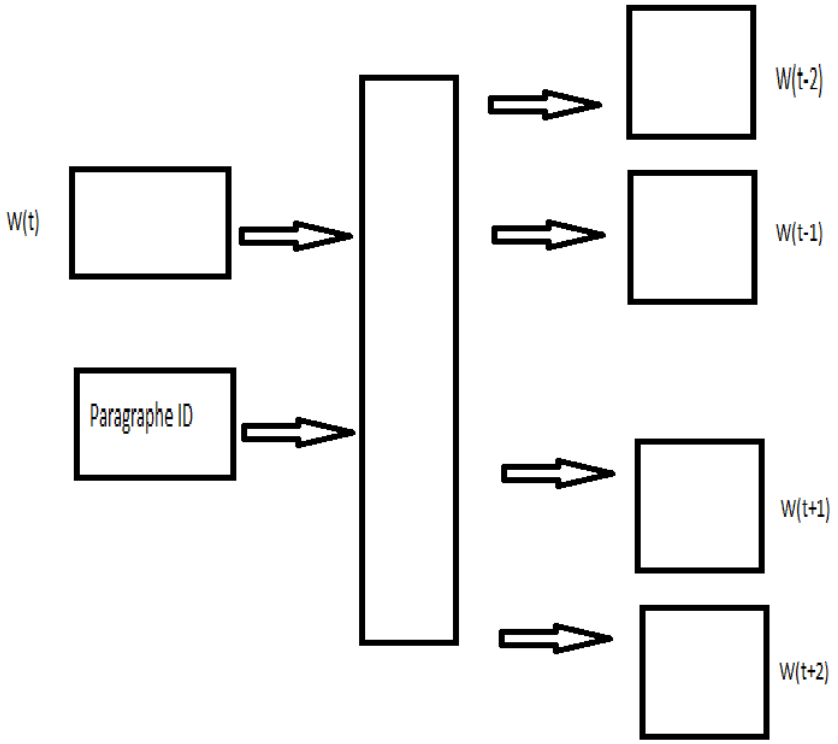

Figure 3: New model with a paragraph matrix

Paragraphe_id is the index review. It is important to modify "sample_example()" in the script because we need to return window and index of the review (windows=[word_window, index_review]). Wp is a new matrix, $n * n$ n is the number of reviews.

It is necessary to also modify compute_dot_products $Wp * ei = Wp[i, :]$ ei is a column vector (n,1) witch has 0 everywhere except at i it is 1.

The Loss function need to be modified to take into account Wp learning.

With this new loss we can change compute_gradients() function with $\frac{\partial L}{\partial w_p}$.

During the train we need to create the vector ei using windows[1] (windows=[word_window, index_review]) witch is the index of the non zero entry .

References

[1] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International conference on machine learning*, pages 1188–1196, 2014.