TTIC 31210 ASSIGNMENT 1
SIMPLE NEURAL NLP

Zewei Chu
Department of Computer Science
The University of Chicago
Chicago, IL 60637, USA
zeweichu@uchicago.edu

1 WORD AVERAGING SENTENCE CLASSIFIER

1.1 EXPOSITION

Each word $w \in \mathcal{V}$ is embedded to a d-dimensional word vector by a lookup table $e(w) \in \mathbb{R}^d$. For a sequence of words s, we use s_i to denote the ith word of the sequence. The average vector of the sequence is then $v(s) = \frac{1}{|s|}\sum_{i=1}^{|s|} e(s_i)$. Then we compute the dot product of the average word vector and a d-dimensional parameter vector $p \in \mathbb{R}^d$, and then get through as logistic sigmoid layer to get the probability of being label 1, $r = \sigma(p \cdot v(s))$. Let $t \in \{0, 1\}$ be the correct label, then the cross entropy loss is defined as:

$$loss = t \log r + (1 - t) \log(1 - r)$$

1.2 IMPLEMENTATION

I implement the model in PyTorch with python 2.7.

1.3 EXPERIMENTATION

TEST accuracy: 0.740253

DEV accuracy: 0.754587

1.4 ANALYSIS

The word with the largest norm is "distasteful" and the word with minimum norm is ",". The top 10 words with largest norms are "distasteful, gallery, efficient, putrid, lousy, implausible, mile, Gone, dupe, subzero", and the top 10 words with smallest norms are ",, the, Heavy, burlesque, Insanely, reclaiming, UNK, Dunst, and, rolls".

My observation is that short and simple words usually have small norms while long and complicated words tend to have large norms.

2 ATTENTION AUGMENTED WORD AVERAGING MODEL

2.1 EXPOSITION

Following the notation of 1.1, we denote

$$v(s) = \sum_{i=1}^{|s|} w_i e(s_i)$$

where

$$w_i = \frac{exp(a \cdot e(s_i))}{Z}$$

and

$$Z = \sum_{i=1}^{|s|} exp(a \cdot e(s_i))$$

The remaining classification step and loss function are the same as 1.1.

2.2 Implementation and Experimentation

The code is written in PyTorch.

2.3 Analysis

Words with high variance attention weights: "We", "stomach", "cinema", "cliches", "Bad", "Too", "moviemaking", "Peter", "loud", "Cool". These words sometimes have strong impact on the sentence sentiment, while sometimes not.

Examples of high variance attention weights	
Sentence	label
If you can **stomach** the rough content , it 's worth checking out for the performances alone .	1
The movie 's accumulated force still feels like an ugly knot tightening in your **stomach** .	0
One long string of **cliches** .	0
Writer/director Joe Carnahan 's grimy crime drama is a manual of precinct cliches , but it moves fast enough to cover its clunky dialogue and lapses in logic .	1
A synthesis of cliches and absurdities that seems positively decadent in its cinematic flash and emptiness .	0

Words with low variance and low mean attention weights: "believable", "Barrie", "nagging", "animated", "cinematography". These words have low impact on the sentence sentiment.

Examples of low variance low mean attention weights	
Sentence	label
Far more imaginative and ambitious than the trivial , cash-in features Nickelodeon has made from its other animated TV series .	1
Confirms the nagging suspicion that Ethan Hawke would be even worse behind the camera than he is in front of it .	0
The acting , costumes , music , cinematography and sound are all astounding given the production 's austere locales .	1
Characters still need to function according to some set of believable and comprehensible impulses , no matter how many drugs they do or how much artistic license Avary employs .	0

Words with low variance and high mean attention weights: "stupid", "provocative", "creative". These words have strong impact on the sentence sentiment.

Examples of low variance high mean attention weights	
Sentence	label
A coarse and stupid gross-out .	0
Falls neatly into the category of Good Stupid Fun .	1
Smart , provocative and blisteringly funny .	1
Very psychoanalytical – provocatively so – and also refreshingly literary .	1
The characters are interesting and often very creatively constructed from figure to backstory .	1
Challenging , intermittently engrossing and unflaggingly creative .	1

2.4 Enriching the Attention Function

1. **Word position**: We add an extra dimension for each word embedding. The extra dimension is the relative position of the embedded word in the whole sentence. For example, if it is

the third word of the sentence and the sentence length is 10, then the extra dimension will be 0.3.

2. **Nearby words**: For each word embedding, instead of the original word embedding, we use a weighted sum of words in slide window size 3 centered at the current word. The weights for the three words are 0.1(previous word), 0.8(current word), 0.1(next word). The first word does not have a previous word and the last word does not have a next word.
3. **Sentence length**: We add an extra dimension after the weighted average word embedding of the full sentence, indicating the length of the sentence.

Accuracy with different attention models				
Attention Types	DEV Accuracy	TEST Accuracy	DEV Loss	TEST Loss
word average	0.754587	0.740253	1.098766	1.059788
weighted avg	0.727064	0.698517	1.142985	1.215174
Word position	0.756881	**0.754530**	**0.714970**	**0.710360**
Nearby words	**0.785550**	0.752334	0.869291	0.866840
Sentence ength	0.764908	0.750686	0.899356	0.859254

The accuracy of different models on DEV and TEST are listed in the above table. Word position with weighted attention gives the best TEST accuracy results, and Nearby words performs best on DEV.