HW1: Classification

Jiafeng Chen Yufeng Ling Francisco Rivera

February 4, 2019

1 Introduction

This note lays out our expectation for a homework submission in *CS287: Statistical Natural Language Processing*. While you do not have to follow this template to the letter, we do expect that write-ups have a very clear structure and cover all the elements described in this note. With this in mind, the burden is on the presenter to demonstrate why deviations from the standard are necessary.

All write-ups should include a short introduction. In this section you should summarize the underlying problem in high-level language and describe the extensions that you have decided to propose in your implementation. When you describe these extensions you should carefully cite the papers of interest. For instance, it will often be useful to cite the work seen in class (Murphy, 2012). Alternatively, you can also cite papers inline, for instance the work of Berger et al. (1996).

2 Problem Description—Old

In general, homeworks will be specified using informal language. As part of the assignment, we expect you to write-out a definition of the problem and your model in formal language. For this class, we will use the following notation:

- $\boldsymbol{b}, \boldsymbol{m}$; bold letters for vectors.
- $\boldsymbol{B}, \boldsymbol{M}$; bold capital letters for matrices.
- $\mathcal{B}, \mathcal{M}$; script-case for sets.
- b_i, x_i; lower case for scalars or indexing into vectors.

For instance in natural language processing, it is common to use discrete sets like $\mathcal{V}$ for the vocabulary of the language, or $\mathcal{T}$ for a tag set of the language. We might also want

one-hot vectors representing words. These will be of the type $\boldsymbol{v} \in \{0,1\}^{|\mathcal{V}|}$. In a note, it is crucial to define the types of all variables that are introduced. The problem description is the right place to do this.

3 Problem Description

The focus of the problem set is sentiment classification. We are given *sentences* and aim to classify them as either positive or negative sentiment (a binary classification). These predictions can be made in a probabilistic fashion by writing y_i as the event that the ith sentence is positive sentiment, and calculating $p(y_i)$.

Sentences are themselves sequences of words $w \in \mathcal{V}$ in some vocabulary $\mathcal{V}$. In particular, there will be two special words, w_{unk}, $w_{\text{pad}} \in \mathcal{V}$ which represent an unknown word and a padding unit respectively; we address their significance later. A particular sequence of words $w_1, \ldots, w_n$ need not have a specific length n, which varies across sentences.

We represent words in two main ways. The first is as a one-hot encoded vector $\boldsymbol{v} \in \{0,1\}^{|\mathcal{V}|}$. This representation is useful for the Logistic Regression model. Alternatively, we can use a dense embedding. That is, each word gets assigned a vector $\boldsymbol{v} \in \mathbb{R}^d$ where d is the embedding dimension. We use Mikolov et al. (2013)'s pre-trained word embeddings ($d = 300$) for the Continuous Bag of Words and Convolutional Neural Network models.

4 Model and Algorithms

4.1 Naive Bayes

As we alluded to earlier, we can treat our prediction problem as probabilistic. We are interested in the probability of

$$p(y_i \mid \boldsymbol{w}) = \frac{p(\boldsymbol{w} \mid y_i)p(y_i)}{p(\boldsymbol{w})}$$

which is given by Bayes' rule. The Bayesian approach is formulated as follows:

- We first assign pseudo-counts to each word that act as priors in the Bayesian framework. This results in $\mathbf{p}, \mathbf{q} \in \mathbb{R}^{|\mathcal{V}|}$, representing the counts of words in positive/negative sentences. In our model, we initialized both to be vector of ones.

- We train the model by updating $\mathbf{p}$ and $\mathbf{q}$. We increment p_i (resp. q_i) by the occurences of word w_i in positive (resp. negative) sentences.

The predicted probabilities for a sentence with features $\boldsymbol{x}_i$ is given by softmaxing over

$$\begin{bmatrix} \log\left(\frac{\mathbf{p}}{\|\mathbf{p}\|_1}\right)^T \boldsymbol{x}_i + \log\left(\frac{N_+}{N}\right) \\ \log\left(\frac{\mathbf{q}}{\|\mathbf{q}\|_1}\right)^T \boldsymbol{x}_i + \log\left(\frac{N_-}{N}\right) \end{bmatrix}$$

where N_+ and N_- are respectively the number of positive sentences and negative sentences in the training set and $N = N_+ + N_-$. In addition, $\boldsymbol{x}_i$ is the max-over-time pooling of one-hot encoding matrix.

4.2 Logistic Regression

4.3 Continuous Bag of Words

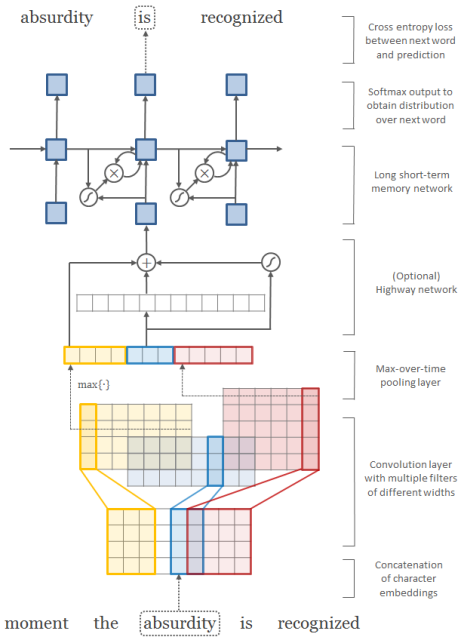
4.4 Convolutional Neural Network

5 Model and Algorithms—Old

Here you specify the model itself. This section should formally describe the model used to solve the task proposed in the previous section. This section should try to avoid introducing new vocabulary or notation, when possible use the notation from the previous section. Feel free to use the notation from class, but try to make the note understandable as a standalone piece of text.

This section is also a great place to include other material that describes the underlying structure and choices of your model, for instance here are some example tables and algorithms from full research papers:

- diagrams of your model,



- feature tables,

Mention Features	
Feature	Value Set
Mention Head	$\mathcal{V}$
Mention First Word	$\mathcal{V}$
Mention Last Word	$\mathcal{V}$
Word Preceding Mention	$\mathcal{V}$
Word Following Mention	$\mathcal{V}$
# Words in Mention	$\{1, 2, \ldots\}$
Mention Type	$\mathcal{T}$

- pseudo-code,

procedure LINEARIZE($x_1 \ldots x_N, K, g$)
  $B_0 \leftarrow \langle(\langle\rangle, \{1, \ldots, N\}, 0, \boldsymbol{h}_0, \mathbf{0})\rangle$
  for $m = 0, \ldots, M-1$ **do**
    for $k = 1, \ldots, |B_m|$ **do**
      for $i \in \mathcal{R}$ **do**
        $(y, \mathcal{R}, s, \boldsymbol{h}) \leftarrow \text{copy}(B_m^{(k)})$
        for word w in phrase x_i **do**
          $y \leftarrow y$ append w

$s \leftarrow s + \log q(w, \boldsymbol{h})$

$\boldsymbol{h} \leftarrow \delta(w, \boldsymbol{h})$

$B_{m+|w_i|} \leftarrow B_{m+|w_i|} + (y, \mathcal{R} - i, s, \boldsymbol{h})$

keep top-K of $B_{m+|w_i|}$ by $f(x, y) + g(\mathcal{R})$

return $B_M^{(k)}$

6 Experiments

Finally we end with the experimental section. Each assignment will make clear the main experiments and baselines that you should run. For these experiments you should present a main results table. Here we give a sample Table 1. In addition to these results you should describe in words what the table shows and the relative performance of the models.

Besides the main results we will also ask you to present other results comparing particular aspects of the models. For instance, for word embedding experiments, we may ask you to show a chart of the projected word vectors. This experiment will lead to something like Figure 1. This should also be described within the body of the text itself.

Model	Acc.
BASELINE 1	0.45
BASELINE 2	2.59
MODEL 1	10.59
MODEL 2	13.42
MODEL 3	7.49

Table 1: Table with the main results.

7 Conclusion

End the write-up with a very short recap of the main experiments and the main results. Describe any challenges you may have faced, and what could have been improved in the model.

References

Berger, A. L., Pietra, V. J. D., and Pietra, S. A. D. (1996). A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71.

Figure 1: Sample qualitative chart.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.