SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

50.007 Machine Learning, Spring 2024

Design Project

Due 21 April 2024 (Week 14 Sunday), 11.59pm

This project will be graded by Chengshun Wang
Please submit your work to eDimension.

Please form groups for this project early, and start this project early.

Instructions

Please read the following instructions carefully before you start this project:

- This is a group project. You are allowed to form groups in any way you like, but each group must consist of either 3 or 4 people. Please send your group information to eDimension as soon as possible if you still haven't done so.
- You are strictly NOT allowed to use any external resources or machine learning packages. You will receive 0 for this project if you do so.
- Each group should submit code together with a report summarizing your work, and give clear instructions on how to run your code. Please also submit your system's outputs. Your output should be in the same column format as that of the training set.

Project Summary

Many start-up companies are interested in developing automated systems for analyzing sentiment information associated with social media data. Such sentiment information can be used for making important decisions such as making product recommendations, predicting social stance and forecasting financial market trends.

The idea behind sentiment analysis is to analyze the natural language texts typed, shared and read by users through services such as Twitter and Weibo and analyze such texts to infer the users' sentiment information towards certain targets. Such social texts can be different from standard texts that appear, for example, on news articles. They are often very informal, and can be very noisy. It is very essential to build machine learning systems that can automatically analyze and comprehend the underlying sentiment information associated with such informal texts.

In this design project, we would like to design our sequence labelling model for informal texts using the hidden Markov model (HMM) that we have learned in class. We hope that your sequence labelling system for informal texts can serve as the very first step towards building a more complex, intelligent sentiment analysis system for social media text.

The files for this project are in `EN.zip`. We provide a labelled training set `train`, an unlabelled development set `dev.in`, and a labelled development set `dev.out`. The labelled data has the format of one token per line with token and tag separated by tab and a single empty line that separates sentences.

The format can be something like the following:

```
Best O
Deal O
Chiang B-positive
mai I-positive
Tours I-positive
, O
The O
North O
of O
Thailand B-neutral
To O
Get O
special O
Promotion O
and O
free O
Transfer O
roundtrip O
. O
Contact O
: O
... O
http://t.co/sSn1OBTZ O

Independent B-neutral
Research I-neutral
Network I-neutral
News I-neutral
is O
out O
! O
http://t.co/KU5k7aHe O
! O
```

where labels such as `B-positive, I-positive` are used to indicate **B**eginning and the **I**nside of the entities which are associated with a positive sentiment. `O` is used to indicate the **O**utside of any entity. Similarly for `B-negative, I-negative` and `B-neutral, I-neutral`, which are used to indicate entities which are associated with negative and neutral sentiment, respectively.

Overall, our goal is to build a sequence labelling system from such training data and then use the system to predict tag sequences for new sentences. Specifically:

- We will be building two sentiment analysis systems for one different languages from scratch, using our own annotations.
- We will be building sentiment analysis system using annotations provided by others.

1 Part 1 *(25 points)*

Recall that the HMM discussed in class is defined as follows:

$$p(x_1, \ldots, x_n, y_1, \ldots, y_n) = \prod_{i=1}^{n+1} q(y_i|y_{i-1}) \cdot \prod_{i=1}^{n} e(x_i|y_i) \tag{1}$$

where $y_0 = \texttt{START}$ and $y_{n+1} = \texttt{STOP}$. Here q are transition probabilities, and e are emission parameters. In this project, x's are the natural language words, and y's are the tags (such as `O, B-positive`).

- Write a function that estimates the emission parameters from the training set using MLE (maximum likelihood estimation):

  $$e(x|y) = \frac{\texttt{Count}(y \to x)}{\texttt{Count}(y)}$$

  (5 points)

- One problem with estimating the emission parameters is that some words that appear in the test set do not appear in the training set. One simple idea to handle this issue is as follows. We introduce a special word token `#UNK#`, and make the following modifications to the computation of emission probabilities:

  $$e(x|y) = \begin{cases} \frac{\texttt{Count}(y \to x)}{\texttt{Count}(y)+k} & \text{If the word token } x \text{ appears in the training set} \\ \frac{k}{\texttt{Count}(y)+k} & \text{If word token } x \text{ is the special token } \texttt{\#UNK\#} \end{cases}$$

  (This basically says we assume from any label y there is a certain chance of generating `#UNK#` as a rare event, and empirically we assume we have observed that there are k occurrences of such an event.)

  During the testing phase, if the word does not appear in the training set, we replace that word with `#UNK#`.

  Set k to 0.1, implement this fix into your function for computing the emission parameters.

  (10 points)

- Implement a simple sentiment analysis system that produces the tag

  $$y^* = \arg\max_y e(x|y)$$

  for each word x in the sequence.

  Learn these parameters with `train`, and evaluate your system on the development set `dev.in` for each of the dataset. Write your output to `dev.p1.out`. Compare your outputs and the gold-standard

outputs in `dev.out` and report the precision, recall and F scores of such a baseline system for each dataset.

The precision score is defined as follows:

$$\texttt{Precision} = \frac{\text{Total number of correctly predicted entities}}{\text{Total number of predicted entities}}$$

The recall score is defined as follows:

$$\texttt{Recall} = \frac{\text{Total number of correctly predicted entities}}{\text{Total number of gold entities}}$$

where a gold entity is a true entity that is annotated in the reference output file, and a predicted entity is regarded as correct if and only if it matches exactly the gold entity (*i.e.*, both their *boundaries* and *sentiment* are exactly the same).

Finally the F score is defined as follows:

$$\texttt{F} = \frac{2}{1/\texttt{Precision} + 1/\texttt{Recall}}$$

Note: in some cases, you might have an output sequence that consists of a transition from `O` *to* `I-negative` *(rather than* `B-negative`*). For example, "*`O I-negative I-negative O`*". In this case, the second and third words should be regarded as one entity with negative sentiment.*

You can use the evaluation script shared with you to calculate such scores. However it is strongly encouraged that you understand how the scores are calculated.

(10 points)

2 Part 2 *(25 points)*

- Write a function that estimates the transition parameters from the training set using MLE (maximum likelihood estimation):

$$q(y_i|y_{i-1}) = \frac{\texttt{Count}(y_{i-1}, y_i)}{\texttt{Count}(y_{i-1})}$$

  Please make sure the following special cases are also considered: $q(\texttt{STOP}|y_n)$ and $q(y_1|\texttt{START})$.

  (10 points)

- Use the estimated transition and emission parameters, implement the Viterbi algorithm to compute the following (for a sentence with n words):

$$y_1^*, \ldots, y_n^* = \arg\max_{y_1,\ldots,y_n} p(x_1, \ldots, x_n, y_1, \ldots, y_n)$$

  Learn the model parameters with `train`. Run the Viterbi algorithm on the development set `dev.in` using the learned models, write your output to `dev.p2.out`. Report the precision, recall and F scores of all systems.

  Note: in case you encounter potential numerical underflow issue, think of a way to address such an issue in your implementation.

  (15 points)

3 Part 3 *(25 points)*

- Use the estimated transition and emission parameters, implement an algorithm to find the *second best* output sequences. Clearly describe the steps of your algorithm in your report.

  Run the algorithm on the development sets `EN/dev.in`. Write the outputs to `EN/dev.p3.out`. Report the precision, recall and F scores for the output.

  Hint: find the top-2 best sequences using dynamic programming by modifying the original Viterbi algorithm.

  (25 points)

4 Part 4 – Design Challenge *(25 points)*

- Now, based on the training and development set, think of a better design for developing an improved sentiment analysis system for tweets using any model you like. Please explain clearly the model/method that you used for designing the new system. We will check your code and may call you for an interview if we have questions about your code. Please run your system on the development set `EN/dev.in`. Write your output to `EN/dev.p4.out`. Report the precision, recall and F scores of your new system.

  (15 points)

- We will evaluate your system's performance on the held out test set `EN/test.in`. The test set will only be released on 19 April 2024 at 11.59pm (48 hours before the deadline). Use your new system to generate the output. Write your outputs to `EN/test.p4.out`.

  The system that achieves the overall highest F score on the test sets will be announced as the winner.

  (10 points)

Hints: Can we handle the new words in a better way? Are there better ways to model the transition and emission probabilities? Or can we use a discriminative approach instead of the generative approach? Perhaps using Perceptron?[1]*. Any other creative ideas? Note that you are allowed to look into the scientific literature for ideas.*

Items To Be Submitted

Upload to eDimension a single ZIP file containing the following: (Please make sure you have only one submission from each team only.)

- A report detailing the approaches and results
- Source code (.py files) with README (instructions on how to run the code)
- Output files
  - `EN/`

[1] http://www.aclweb.org/anthology/W02-1001

1. `dev.p1.out`
2. `dev.p2.out`
3. `dev.p3.out`
4. `dev.p4.out`
5. `test.p4.out`