Implementation of Question Classifiers

Yuyang Zhou(10105592), Jinpei Han(9865011), Yilei Wu(9981601), Hongbo Zhu(10313053)
University of Manchester, UK

1 Problem Statement

This project aims to implement questions classifiers which receive a question put forward by users and then return a tag standing for the type of question. For example, if the question is "How many points make up a perfect fivepin bowling score?", the output returned by the classifier would be "NUM:count". To achieve the goal, the implementation is divided into three steps – word pre-processing stage, data training stage, and data testing stage. To deal with the text datatype, vector representation of words is also required since the classifier only processes data with numbers instead of characters.

2 Approaches

2.1 Bag Of Words

In BOW, sentences are firstly separated into words so that the vector of each word can be obtained according to the embedding which is processed previously. By averaging the sum of the vectors of words of a sentence, the sentence vector is calculated. Hence, as the same method is applied to the rest of the sentences, a sentence vectors list would be obtained. Then the sentence vector list, as well as the label set size and vector dimension of the training data, would be fed into the feed-forward neural network to train the desired model. Eventually, the trained model can classify a sentence which either input by users or used a testing data, and then output a scoring matrix. The prediction label can be calculated by finding the one with the greatest probability from the score matrix, which leads to the actual result – the type of the question.

2.2 BiLSTM

Bi-directional Long short-term memory(BiLSTM) network is a combination of Bidirectional recurrent neural network(RNN) and LSTM RNN. In this model, the structure of the feed-forward neural network can be described as follows: Embedding layer, which is a basic lookup table which converts a list of indexes of words in a sentence into a list of random/pre-trained word vectors. A BiLSTM network learns the knowledge of the embedded sentences and pass them to the next layer.An Attention network modifies the output of BiLSTM by making the network focus on a subset of every single sentence. A Linear Layer maps the hidden state to label space.Finally, a SoftMax Layer maps the temporary label space to a probability matrix by logarithm function. For better implementation, additional processing needs to be done on the training data set before the training starts. First, the length of sentences is normalized into a determined maximum length. If the sentence is too short, it will be filled with padding elements which will be ignored by LSTM in the training phase. If the sentence is too long, the overflow will be cut off. Besides, to improve the training efficiency, the training data set is divided into batches in a specified size.

2.3 Ensemble Classifier

The ensemble classifier can load two trained models. The models can be two BOW models or two BiLSTM models or one of each combined. The test data is first fed into each model after preprocessing and produces a prediction score matrix. The prediction score matrix from each model is then normalized to range 0-1. Finally, all prediction score matrix is averaged and the label with the highest score in each row is predicted.

3 Experiments

3.1 Data Pre-Processing

Data pre-processing starts with splitting the dataset 9 to 1 portion into training data and testing data respectively. The program splitdata.py takes the path of the dataset as an input argument and load the data into an array. The whole dataset is shuffled and splited into training and testing set by index. The main program question_classifier.py then loads the data depending

on training or testing and split the labels and the questions into two separate lists. The question list is then processed under several stages which can be specified in the configuration file, including lower case conversion, punctuation removal, non-alphabetical tokens removal, and stop-words removal. The processed questions are then saved into a two-dimensional list which contains questions in the first dimension and the words in each sentence in the second dimension. The labels and words are then indexed and composed two dictionaries word_to_ix and label_to_ix respectively. The information of vocabulary is then passed into BOW and BiLSTM depending on the model specified in the configuration file for training and testing.

3.2 Experiment setup

To investigate the performance of the implemented models, the performance of the models is tested with different combinations of parameters. Each experiment has been performed three times due to time limitations to get an average accuracy. The first experiment tested the relationship between training epochs and data loss. The second experiment tested the performance of BiLSTM and BOW models under different sizes of training data. The results are shown in the result section below.

3.3 Results

Table 1 in the appendix shows the overall performance of different models with different parameters. The parameters are displayed partially due to the limit of the table width. Most of the time, the overall performance of the ensemble classifiers is higher than the average performance of the combined two classifiers.

Figure 1 in the appendix shows the relationship between data loss and training iterations, which illustrates that the learning loss decreased as the number of training steps increased.

Figure 2 in the appendix shows the effect of using just part of the training data set on each of the models. The performance of both classifiers increased as the number of training data increased. However, the BiLSTM model is more likely to be affected by the shortage of training data, thereby producing a poorer result.

4 Evaluation

4.1 Random and Pre-trained Embedding

If the word embedding is randomly initialized, the precision of word embedding would remain unknown. To make sure the word embedding suitable enough for training the model for classification, updating steps are necessary. An optimizer is constructed to alleviate the uncertainty. During the training stage, the accuracy of word embedding can be improved by increasing the number of the iteration of training, in which case, the optimizer can update the word embedding automatically for every training stage. The loss of data is also calculated by summing up the loss of every sentence so that we can know how well the model is trained. With more iterations, the model would be more accurate as the data loss decreases and the change of loss (as shown in Figure 1) decreases, too. Therefore, it is reasonable to assume that the data of loss would eventually reach a convergence stage if the number of the iteration is big enough.

According to table 1, pre-trained embedding does not provide better performance than randomly initialized embedding, as the pre-trained embedding involves a much larger range of vocabulary where the training data set does not. The pre-trained embedding, therefore, has less similarity for some specific words, which leads to a smaller probability when predicting the best-matched labels.

4.2 Freeze/Fine tune Pre-trained Embedding

According to table 1, the performance of using fine-tuned pre-trained Embedding is pretty low. Furthermore, the performance of using a freezed pre-trained weight is very close to random guessing. As discussed in the previous section, the pre-trained embedding contains too much vocabulary, and its relationship with words that are closely related in this particular dataset are relatively low. In order to achieve similar performance with randomly initialized embedding, the learning rate need to be set as the same as the learning rate when using randomly initialized embedding. However, using a high learning rate and large training iterations means the pre-trained embedding should be treated as a random one rather than 'fine-tune'.

4.3 Performance with parts of training set

The quality of the training data set directly affects the performance of the training model. If only a part of the training data set is used, it will make

the selected part of the training data set not representative enough to the information in this field, or the training set itself. A too small sample category will make the model's learning effect poor, and lead to an increase in the model's false positive rate for the testing data set.

Compared with the bag of words, the BiLSTM model is more dependent on the size of training data set. According to Figure 2, when the training data set is relatively small, the classification accuracy shown by the BiLSTM model is lower, about 25%. The increase of the size of the data set leads to a higher classification accuracy of BiLSTM, with the highest classification accuracy reaching to 75%. In contrast, the performance of Bag of Words is quite robust, and it can still achieve a classification accuracy of 50% even with fewer training data samples. But as the size of training data sets increases, the classification accuracy of BOW increases slowly, and finally maintain at 65%.

4.4 Some Difficult Classes Classification

For some special classes, the classifiers perform less ideal. For example, classes such as "ENTY:" or "DESC:" tend to be difficult to be classified for the classifiers because the questions provided would usually start with "What" but sometimes the second word like "currency" is the key information for correct classification. Thus, it is more difficult to classify these types of classes if the classifiers are not properly trained during the training phase.

4.5 Confusion matrix

Since neither data nor labels used are binary, it can be difficult to construct and observe the confusion matrix due to the fact that the size of the confusion matrix is based on the exact number of classes existed in the data set. However, since the number of classes involved can be variable and the testing label set can be different from the training label set, the use of the confusion matrix is abandoned. An example constructing confusion matrix of the BOW model is attached in the Appendix with 50 different labels.

4.6 Stop Word Removal

According to table 1, it is found that the result without removing stop word shows a higher classification accuracy. This is because the classification is made depending on keywords such as "what", "when" and "who", which are integrated into the provided stop word set. If the models are trained without those words mentioned, the model's recognition of sentences will be biased, leading to the weak performance of classification. The difference of the accuracy of the BiLSTM model is significantly greater than the BOW model, as it fulfills sentences with PAD when the stop words are removed, which would largely affect the precision of embedding, whilst the BOW model relies more on the embedding of each word.

5 Conclusion

In this coursework, question classifiers with BOW and BiLSTM Model are implemented, whose performance depends on various aspects such as the size of data input, number of training, stop word removal and learning rate, etc. The improvement of the performance is realistic by achieving higher efficiency technologies, for example, ensemble. The future work in this field is to find out a more efficient method for classification or to obtain more classification information such as performance.

Appendices

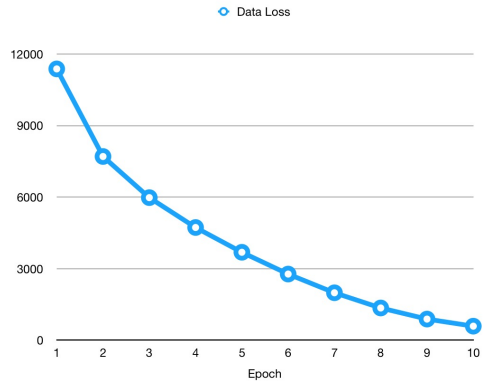

Figure 1: Data loss with different training epochs

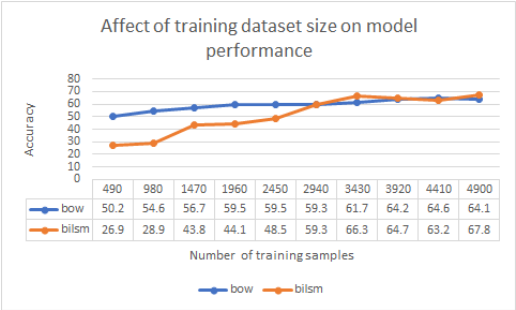

Figure 2: Accuracy with different sizes of training data

Table 1: Performance of models with different parameters

Model	hidden dim	stop words removal	word embeding	learning rate	accuracy
1. bow	128	FALSE	random	0.01	64.1
2. bilstm	128	FALSE	random	0.1	72.1
3. bilstm	32	TRUE	random	0.1	63.6
4. bow	128	TRUE	random	0.01	65.5
5. ensemble(1+2)	NA	NA	NA	NA	73.4
6. ensemble(3+4)	NA	NA	NA	NA	67.9
7. ensemble(1+4)	NA	NA	NA	NA	69.8
8. ensemble(2+3)	NA	NA	NA	NA	75.9
9. bow	32	FALSE	fine tuned pretrained	0.001	53.2
10. bilstm	32	FALSE	fine tuned pretrained	0.001	40.7
11. bow	32	FALSE	freezed pretrained	0	51.3
12. bilstm	32	FALSE	freezed pretrained	0	27.2
13. bow	128	TRUE	pretrained	0.01	59.5
14. bilstm	32	TRUE	pretrained	0.1	68.5

Appendix A Confusion Matrix (BOW 10 Epoch, Learning Rate = 0.1, Stop Word Removed)

```
[[15. 0. 0. 0. 6. 0. 0. 0. 0. 0. 0. 2. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 22. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 2. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 1. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 1. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 6. 0. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 6. 0. 0. 0. 76. 1. 0. 0. 0. 1. 0. 1. 0. 0. 0. 1. 0. 0. 3. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [ 1. 0. 0. 0. 4. 11. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 1. 1. 1. 1. 0. 22. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 1. 1. 0. 0. 12. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 1. 0. 2. 1. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 1. 2. 0. 3. 1. 0. 0. 11. 0. 0. 0. 3. 0. 0. 3. 0. 0. 0. 2. 0. 0. 0. 0. 1. 0. 0. 1. 0. 1. 0. 0. 0. 0. 1. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 35. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 2. 0. 0. 0. 4. 2. 0. 5. 1. 0. 0. 1. 0. 2. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 1. 7. 1. 0. 1. 1. 0. 0. 0. 0. 2. 0. 30. 0. 0. 2. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 21. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 2. 0. 0. 0. 1. 0. 0. 3. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 5. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 0. 4. 0. 0. 2. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 16. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 11. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 11. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

```
 [ 0. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 8. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 10. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 0. 2. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 3. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 2. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 1. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 4. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 3.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 2. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 2. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 1. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
 [ 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]]
```