Assignment 3 Report

Amirhossein Sohrabbeig - STD: 1744420

2022-03-18

In this assignment, I used DeepStellar [1] paper for implementation.

1 Implement state abstraction

1.1 Build state abstraction and transition model based on training data

Code completion.

1.2 Based on your implementation, which state is most frequently visited?

```
Most frequently visited state is [5, 4, 5] which is visited 380580 times
```

Listing 1: Most frequently visited state

2 Implement a function to obtain state transition

Code completion.

```
trace is:
    [[5 4 5]
    [5 5 5]]
prediction is: 0(World)
```

Listing 2: Test result of get_trace function using a minimal example

3 Implement the metrics for measuring state-based trace similarity and transition-based trace similarity

Code completion.

4 Use DeepStellar to analyze adversarial attack

4.1 Output traces of original data and attacked data.

```
trace of text1 is:
    [[5 4 5]
    [5 5 5]
    [5 5 5]
    [4 5 5]
    [4 5 5]
    [4 5 6]
    [4 5 6]
    [4 5 6]
    [5 5 6]
    [5 5 6]
    [5 5 6]
    [5 5 6]
    [5 5 6]
    [5 6 6]
    [5 6 6]
    [5 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]
    [4 6 6]]
prediction is: 2(Business)
trace of text2 is:
    [[5 4 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 5 5]
    [5 6 5]
    [5 6 6]
    [5 6 6]
    [5 6 5]
    [5 6 5]
    [5 6 5]
    [5 6 5]
    [5 6 5]
    [5 6 6]
    [5 6 6]
    [5 7 6]
    [5 7 6]
    [5 7 6]
    [5 7 6]
    [5 7 6]
    [5 7 6]
    [4 7 6]]
prediction is: 0(World)
```

Listing 3: traces of original data and attacked data

4.2 Draw a figure to visualize each trace.

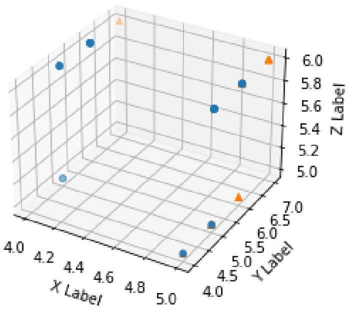

Figure 1: Visualization of abstract states.

4.3 Calculate their state-based trace similarity and transition-based trace similarity based on the defined functions in 3

```
State-based trace similarity: 0.30
Transition-based trace similarity: 0.16
```

Listing 4: state_based and transition_based similarity

4.4 Analyze the difference between original data and attacked data: give a brief explanation on why the model's prediction result is incorrect on the attacked data.

The attacked data is different from the original data in just the second word; it uses the word ”kill” instead of ”defeat.” That minor alteration does not change the meaning for humans, so the second sentence is expected to be classified as business class. However, we can see in Listing 3 that slight modification deviates the trace of states from the fourth state to the end. The ending abstract state is [4 6 6] for the first sentence and [4 7 6] for the other one, which makes the model's prediction different for these two sentences.

I thought maybe the words ”kills” and ”defeats” have different distributions regarding the predicted class of the sentences they occurred in. I plotted their distributions, and that confirms my guess.

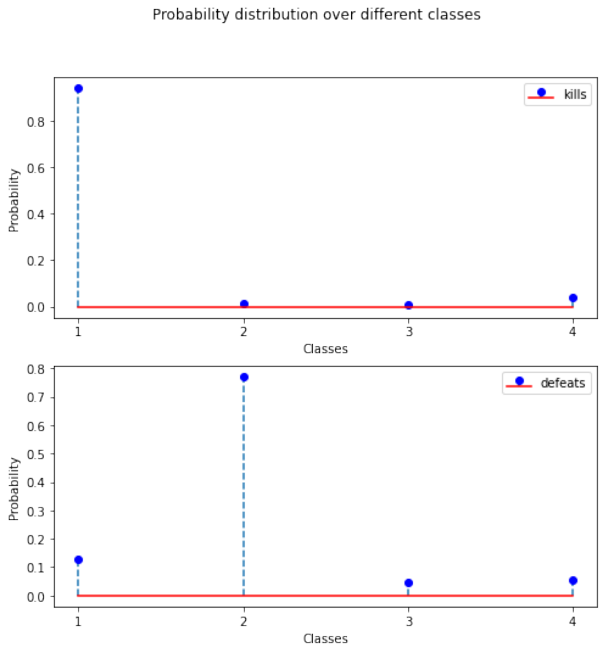

Figure 2: Probability distribution of ”kills” and ”defeats” with respect to classes

Most of the sentences that contains the word ”kills” and ”defeats” are classified as World and Sport, respectively. I believe that is the reason why we see a deviation in state trace of those two sentences, and as a result their different predicted classes.

5 Brief discussion on the open question: how to further improve the state abstraction method?

I think we can improve the results of Deepstellar model by doing a hyperparameter optimization on the number of dimensions for the dimensionality reduction. Moreover, we can use auto-encoder models instead of PCA to find a better low-dimensional representation of concrete states. We can also use better embedding representation of words by training models or existing pre-trained models.

References

[1] Xiaoning Du, Xiaofei Xie, Yi Li, Lei Ma, Yang Liu, and Jianjun Zhao. Deepstellar: Model-based quantitative analysis of stateful deep learning systems. *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019.