1. Problem statement: Restate the initial project that you proposed in deliverable one in 2-3 sentences. Be sure to refer back to this problem statement in the following questions.

Answer

We are implementing a two-way Shakespearean to modern English translator. We are using a transformer encoder-decoder with attention model and plan to use Flask for deployment.

2. Data Preprocessing: Confirm the dataset you are working with. State any changes from the initial dataset you chose. Discuss the content of the dataset (number of samples, labels, etc). Describe and justify your data preprocessing methods.

Answer

Our dataset is scraped from Sparknotes' No Fear Shakespeare translations, as planned. Currently we have around 9[1] plays scraped, giving us approximately 22,000 lines of Shakespearean text and the corresponding 22,000 modern equivalents. This number could be higher without scraping more plays, but we limited ourselves to sentences no more than 100 characters in length for faster training during development.

During preprocessing, we break the lines into sentences and pair them as input-target language, we remove capitalization, digits, non-printable characters, and extra white space. We also add spaces around words and punctuation when needed as well as `<start>` and `<end>` tokens to every sentence. The scraper did an excellent job of recovering data from the website, which itself is clean, so there are no irregularities such as missing sentences. Though some translations are dubious, fixing them is not a priority as of now.

[1]Note that we can easily get more, as the scraper generalizes well to all other Shakespeare plays on the website.

3. Machine learning model: In the first deliverable, you proposed a model for your project. If you decided to change your model, explain why. Restate your chosen model and elaborate on the design decisions. Report the following:

   (a) Discuss the framework and tools that you used for your model. Explain your choice. Provide architecture graphs as appropriate. Justify any decision about training/validation/test splits, regularization techniques, optimization tricks, setting hyper-parameters, etc.

   (b) Description of validation methods: How did you test your model? Is your model overfitting or underfitting?

   (c) Did you face any challenges implementing the model? If so, how did you solve it?

Answers

(a) Through the Keras API, we train a tensorflow transformer encoder-decoder model with attention. Truth be told we do not understand enough to graph it yet, so pretend like there is something cool here.

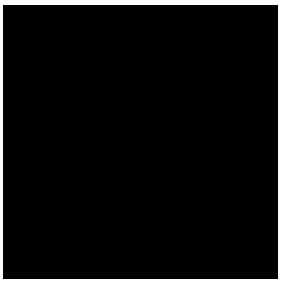


Figure 1: Pretend that this is an architecture graph.

The current model has 3 layers, and was trained for around 10 minutes on a gtx 1070 GPU, as we first wanted to see if it works at all. From initial tests, the final implementation should be trainable locally, but we already started a Google Cloud trial so we'll run the final version there.

Though transformers are, according to KC and various other sources, more expensive to train, the other models we have tried to implement did not work. At all. Therefore our choice of model was one of necessity.

(b) The model has only started to pass the python interpreter yesterday – prior to today, it was outputting only a single word. Thus with this little time, the only evaluation method we have used so far is trying out translating sentences. It is currently dreadfully underfit, but at least it is functional.

(c) We faced a lot of challenges. The first one came from the lack of a preexisting data set. Figuring out how to scrape data from the website and deal with unexpected errors was the main problem. For instance, when accessing the HTML tables, we were looking for a specific title pattern. However, one page (out of 150...) followed a different naming convention and so, when trying to access the rows latter on, we were calling a function

on a table that did not exist which caused the program to return an error. A fix for that has been to search for general tables, since each page only has one.

Moreover, the website is formatted such that the original text and translation are compared by line rather than by sentence. This caused us to have some unnecessarily long paragraphs in our original-translated pairs.We addressed this issue by checking if there is more then one of '.', '!' ,'?' in a line of each pre-processed pair. If it's the case, then instead of adding it to our training set directly, we make new pairs out of corresponding sentences while checking if the lengths are about the same. For instance, some translation have a dot while the original text has a coma or no punctuation. In these cases we discard the line altogether.

The most important obstacle we faced, and indeed *are facing*, is probably the lack of good or appropriate tutorials and resources. Many of them are incomplete and/or outdated and/or beyond our knowledge, or so simplified as to be useless. Thus many of our implementations failed due to the removal or replacement of certain features from libraries, or lack of completeness from tutorials. Many times, we had to either discard whole blocks of code or modify them in order to mesh them with a newfound implementation.

Finally, an issue we had for a while is that the algorithm was outputing a single word when trying to translate whole sentences. That said, after trying out the tutorial code with the tutorial's dataset, it came to our attention that the same problem was encountered whenever the number of sentences used for training was low. Since there are only so much Shakespeare plays, we decided to yet again change our model. Though it still following a neural machine translation with attention model, we opted for a transformer model with scaled dot product and multi-head attention instead of Bahdanau – it now outputs something resembling full sentences.

4. Preliminary results: In this section, you will focus on the performance of your model. Confirm the metric discussed in Deliverable 1. Present a detailed analysis of your results, providing graphs as appropriate. In addition to an evaluation metric, discuss the overall performance of the model and the feasibility of the project with these results. Remember, graphs are beautiful and we love them!

Answers

We just got a working model on the 22nd of February, and have yet to evaluate it. The current model is Modern to Shakespearean, but I have included a Shakespearean input to see what happens. With that said, here are some sample outputs:

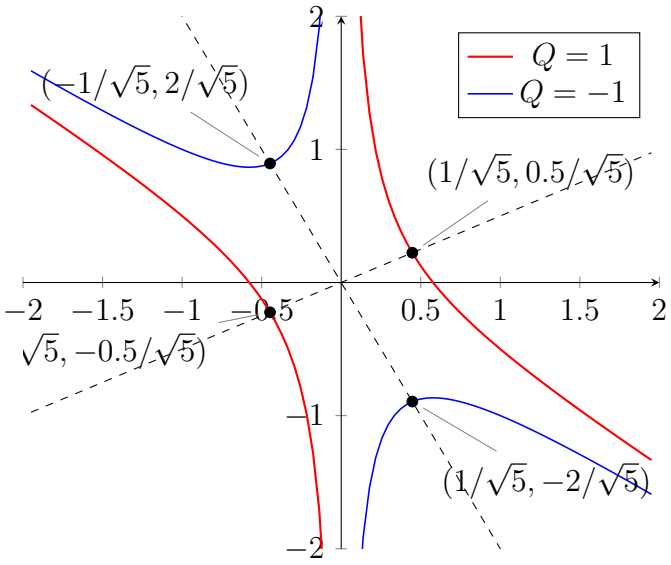
```
input: <start> this is a test . <end>
output: <start> this is a man . <end>

input: <start> <start> so frowned he once when ,  in an angry parle ,
      he smote the sledded polacks on the ice . <end> <end>
output: <start> if he have been a man of the king . <end>

input: <start> let me borrow your lantern so I can check on my horse in the stable . <end>
output: <start> let me see me to see the duke of my heart . <end>

input: <start> cringe translator is garbage . <end>
output: <start> i am gone . <end>
```

Figure 2: Not related, but you love graphs so here is one.



Still, nonsense aside, the model seems functional, outputting something that at least looks like text based off of $\sim$ 10 minutes of training on a gtx 1070. Feasibility-wise, this is very encouraging.

5. Next steps: Discuss your next steps. Describe the pros and cons of your approach and future work. will you be altering your model? For example, will you be fine-tuning it? At this point, if you think that your model is not performing well and/or does not work, please reach out to an exec to see what you can do to improve it.

Answers

Since we have very limited results at the moment it would be hard to tell what the pros and cons of our approach are just yet. It does output complete sentences rather than single words like our previous model did, so we can confidently say this is progress. As for the next steps, definitely more training will be needed. Also since we changed models many times, there remains some incompatibilities and redundancies in our code that need to be fixed. Furthermore, our model is only a one-way translator modern to Shakespearean English as of now. Since we want to make it two-ways we will be implementing that in the future. We have yet to make a GUI or web integration for it as well, but this will come much later.