Keaun Moughari

CAP5619 – Programming Project 2

Professor Liu

4 April 2022

1. Task I

The objective of the current assignment is to train a sequencing model on protein sequence data of varying length, of which each sample is comprised of at most 20 different amino acids. For operational reasons, the samples of the dataset will be one-hot encoded whereby the nonzero entry of each character's vector representation corresponds to an entry of the vocabulary. Due to limitations of computational power of my own machine, I've limited the dataset to only consist of sequences with a minimum length of 2 and maximum of 100 (including the end-of-sequence symbol). Language models build on conditional probabilities of each preceding token (in this case, character), so the last amino acid of each sequence is used as the label for that particular sample while its pace in the original sample is eclipsed by a padding of zeros, ending with the EOS symbol – that is, the model should learn the correct amino acid that completes the sequence. I've chosen a batch size of 128 because I believe that this amount of samples per step will be enough to be representative of the dataset at large. All of these considered, the resulting input shape that the model will accept will be [128, 100, 21].

As for the architecture, its constituent parts are explicitly as follows: a bidirectional long short-term memory (Bi-LSTM) layer of 64 units (activation: tanh, recurrent activation: sigmoid), a batch normalization layer, a dropout layer (probability: 20%), a fully-connected layer of 256 units (activation: ReLU), and a fully-connected output layer of 21 units (number of characters of the vocabulary) that uses softmax activation.

I've decided to use a Bi-LSTM to accomplish the task because I believe it will be effective in the task mentioned above. My reason for this is, each sequence can be comprised of multiple proteins, or different, reoccurring substrings of amino acids, making it necessary for the model to consider the characters that will follow as well as ones that precede a particular place in a sequence. That being the case, the defining features of a bidirectional model will address the former and an RNN model the latter. As for the LSTM's contribution to the problem, they make use of a system of gating units that control the flow of information, affording the model to capture long-term dependencies more effectively. For example, if previous consecutive characters of a substring follow the form of a known protein that could potentially be a constituent part of the superordinate protein sequence being processed, an LSTM cell can learn whether or not it should keep into consideration the previous characters or discard them . Reasoning for the output layer is trivial, but the 256-unit fully-connected layer is used to catch

any of the expressible functions that the sequence model fails to capture. As for batch normalization and dropout, the former has been said to cause a smoothing of the loss function which leads to faster convergence and better generalization performance, and the latter is meant decrease the chance of overfitting. (Use of ReLU activation functions in sequencing models can suffer from exploding gradients, so I settled on the default configuration that Keras implements).

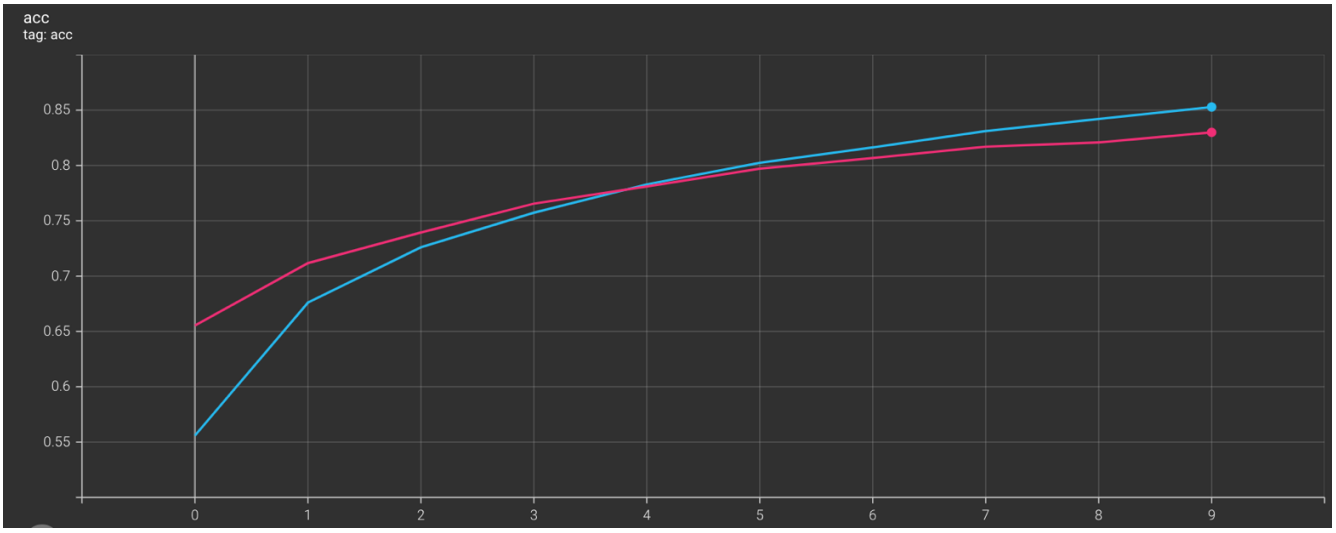
2. Task II

The general weights of the Bi-LSTM layer uses a Glorot Uniform initialization scheme, the recurrent weights are initialized using an orthogonal scheme to help with stability and prevent vanishing/exploding gradients, and the bias is set to all zeros. The weights and bias of the fully-connected layers also follow the initialization schemes of the Bi-LSTM layer, not including the initialization of recurrent weights for obvious reasons.

The optimization algorithm chosen for the model is stochastic gradient descent (SGD) with a learning rate of 3e-2, as 1e-2 was too slow and 5e-2 was too high.

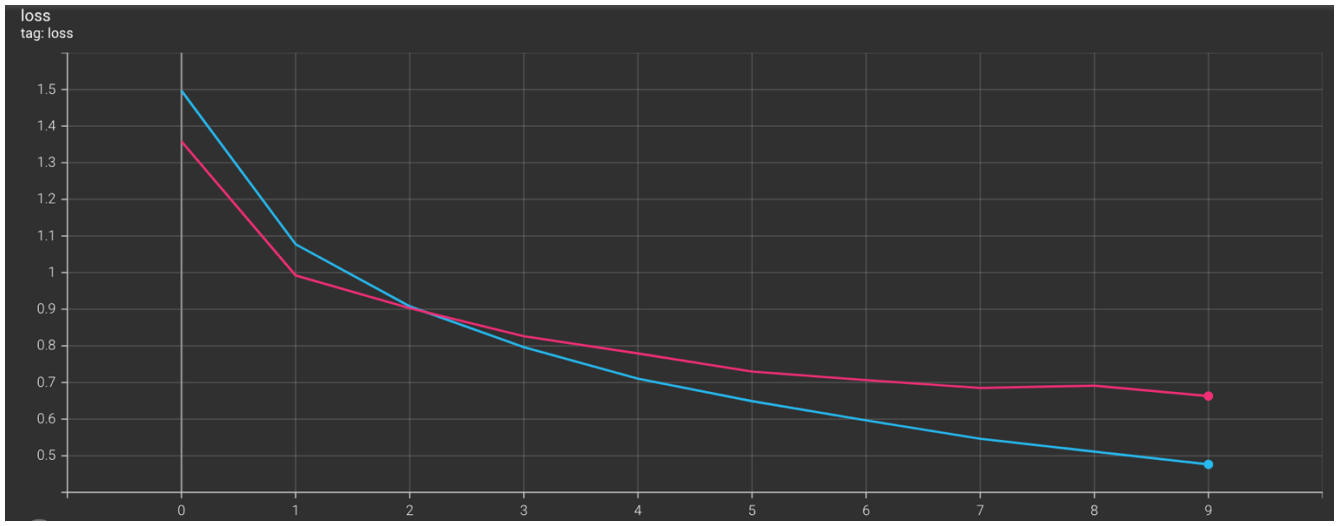
Accuracy

Name	Smoothed	Value	Step	Time	Relative
test	0.8299	0.8299	9	Sat Apr 9, 14:39:33	0s
train	0.8529	0.8529	9	Sat Apr 9, 14:39:33	0s



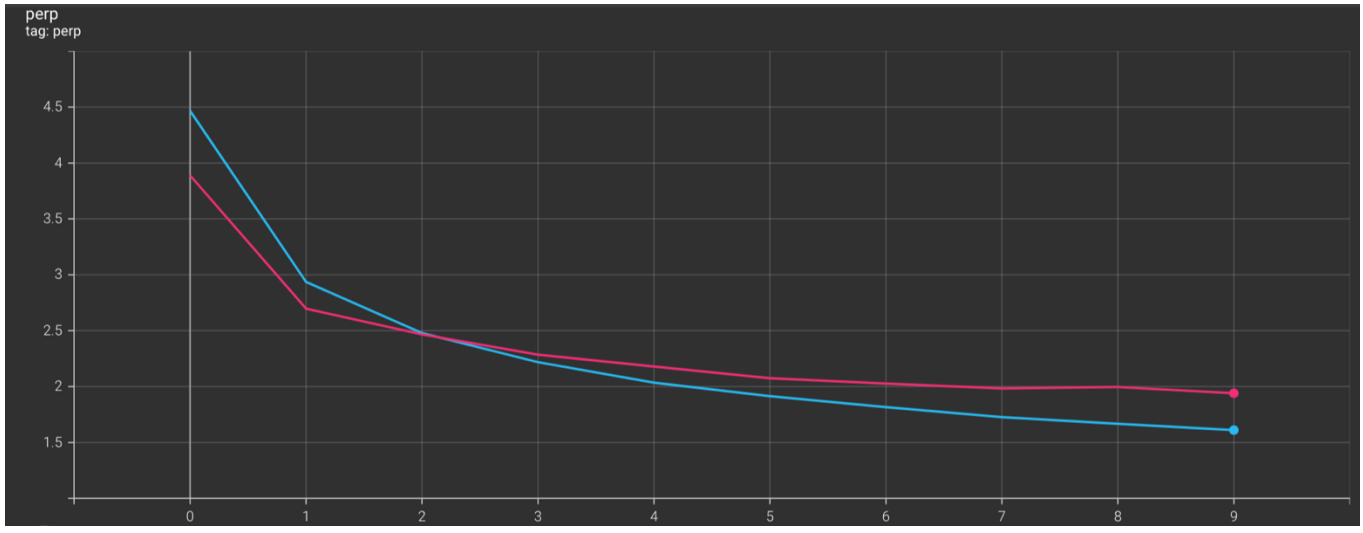
Loss

Name	Smoothed	Value	Step	Time	Relative
test	0.6628	0.6628	9	Sat Apr 9, 14:39:33	0s
train	0.4765	0.4765	9	Sat Apr 9, 14:39:33	0s



Perplexity

Name	Smoothed	Value	Step	Time	Relative
test	1.94	1.94	9	Sat Apr 9, 14:39:33	0s
train	1.61	1.61	9	Sat Apr 9, 14:39:33	0s



For long-term dependencies testing, I made a new test set with samples that the model predicted correctly on and, using 3 of those samples, I tested how the probabilities of the predictions would change as each index of the sample would change to a random character of the vocabulary. I found that for 2/3 samples, the longest dependency was the first character of the sequence, while the outlier's longest dependency was the second character.

```
idx:  0
Original:  GLFGAIAGFIEGGWTGMIDGWYGSGKKKK
Altered:  >I<LFGAIAGFIEGGWTGMIDGWYGSGKKKK

True Top 3: ['0.8596604466', '0.0540446080', '0.0285916626']
New Top 3: ['0.5620129108', '0.1936844438', '0.0835439265']
Difference of Top 3:  ['0.2976', '0.1396', '0.0550']



idx:  1
Original:  GLFGAIAGFIEGGWTGMIDGWYGSGKKKK
Altered:  G>I<FGAIAGFIEGGWTGMIDGWYGSGKKKK

True Top 3: ['0.8596604466', '0.0540446080', '0.0285916626']
New Top 3: ['0.7689346671', '0.0898156837', '0.0509236492']
Difference of Top 3:  ['0.0907', '0.0358', '0.0223']



idx:  2
Original:  GLFGAIAGFIEGGWTGMIDGWYGSGKKKK
Altered:  GL>I<GAIAGFIEGGWTGMIDGWYGSGKKKK

True Top 3: ['0.8596604466', '0.0540446080', '0.0285916626']
New Top 3: ['0.8551705480', '0.0582753345', '0.0397687815']
Difference of Top 3:  ['0.0045', '0.0042', '0.0112']
```

```
idx:  0
Original:  GSHMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE
Altered:  >Q<SHMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE

True Top 3: ['0.8232359290', '0.0842136741', '0.0357957780']
New Top 3: ['0.8564868569', '0.0832778811', '0.0261487272']
Difference of Top 3:  ['0.0333', '0.0009', '0.0096']



idx:  1
Original:  GSHMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE
Altered:  G>Q<HMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE

True Top 3: ['0.8232359290', '0.0842136741', '0.0357957780']
New Top 3: ['0.7655990720', '0.1358013451', '0.0286748856']
Difference of Top 3:  ['0.0576', '0.0516', '0.0071']



idx:  2
Original:  GSHMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE
Altered:  GS>Q<MEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE

True Top 3: ['0.8232359290', '0.0842136741', '0.0357957780']
New Top 3: ['0.8453207612', '0.0682670325', '0.0465100035']
Difference of Top 3:  ['0.0221', '0.0159', '0.0107']



idx:  3
Original:  GSHMEEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE
Altered:  GSH>Q<EEEEEEEEEEDEDEDEDEDEAGSELGEGEEEVGLSYLMKEEIQDE

True Top 3: ['0.8232359290', '0.0842136741', '0.0357957780']
New Top 3: ['0.8115833402', '0.1000346392', '0.0226170458']
Difference of Top 3:  ['0.0117', '0.0158', '0.0132']
```

```
idx:  0
Original:  MELPAPVKAIEKQGITIIKTFDAPGGMKG
Altered:  >M<ELPAPVKAIEKQGITIIKTFDAPGGMKG

True Top 3: ['0.8080651164', '0.0887834653', '0.0401112549']
New Top 3: ['0.8080651164', '0.0887834653', '0.0401112549']
Difference of Top 3:  ['0.0000', '0.0000', '0.0000']



idx:  1
Original:  MELPAPVKAIEKQGITIIKTFDAPGGMKG
Altered:  M>M<LPAPVKAIEKQGITIIKTFDAPGGMKG

True Top 3: ['0.8080651164', '0.0887834653', '0.0401112549']
New Top 3: ['0.8769792914', '0.0362282880', '0.0217318013']
Difference of Top 3:  ['0.0689', '0.0526', '0.0184']



idx:  2
Original:  MELPAPVKAIEKQGITIIKTFDAPGGMKG
Altered:  ME>M<PAPVKAIEKQGITIIKTFDAPGGMKG

True Top 3: ['0.8080651164', '0.0887834653', '0.0401112549']
New Top 3: ['0.4859776497', '0.3217590451', '0.0660591573']
Difference of Top 3:  ['0.3221', '0.2330', '0.0259']



idx:  3
Original:  MELPAPVKAIEKQGITIIKTFDAPGGMKG
Altered:  MEL>M<APVKAIEKQGITIIKTFDAPGGMKG

True Top 3: ['0.8080651164', '0.0887834653', '0.0401112549']
New Top 3: ['0.8244720101', '0.1003016829', '0.0206247382']
Difference of Top 3:  ['0.0164', '0.0115', '0.0195']
```

3. Task III

```
generated protein: TGGTTTGGGGGGTGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG
generated protein: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
generated protein: LFFFKKKKNNNEEEEDEEDDEEEEEIIIIIIIIFIIIFFIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
generated protein: YFNNNNNNNNNNNNNNNNNNNNNNNN
generated protein: WWWWWWCCSCC
```

For generating protein sequences, it is obvious the model has some difficult. I believe this might have something to do with the fact that the first amino acid given to the model is randomly chosen and might not be realistic as a choice for the first amino acid.