Download zip from <https://github.com/scpd-proed/XCS224N-A5>

Unzip content into

C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master

* Open Anaconda Prompt

(base) C:\Users\ADMIN>python --version

Python 3.7.3

* navigate to your project

(base) C:\Users\ADMIN>cd C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master

# Environment Setup Start

* Create a virtual environment using conda

(base) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>conda env create --file local\_env.yml

|  |
| --- |
| CondaValueError: prefix already exists: C:\ProgramData\Anaconda3\envs\local\_nmt  Note: The name “local\_nmt” has been used by Assignment4. 🡺 change the name in the local\_env.yml file to “local\_nmt5”  (base) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>conda env create --file local\_env.yml  Collecting package metadata (repodata.json): done  Solving environment: failed  ResolvePackageNotFound:  - torchvision=0.5.0  - pytorch=1.4.0 |

Note:

* The soumith channel (<https://anaconda.org/soumith>) shows that pytorch is 1.0.1 and torchvision is 0.2.1, which are behind the specified pytorch=1.4.0 and torchvision=0.5.0.
* On the other hand, the local\_nmt env created for the Assignment4 shown below are identical to those specified for the Assignment5, except that the tqdm is 4.42.0 in Assignment4, slightly higher than the 4.41.1 specified for Assignment5.
* Conclusion: Using the same local\_nmt env for Assignment5.

The output from running local\_env.yml in Assignment4

|  |
| --- |
| Collecting package metadata (repodata.json): done  Solving environment: done  Downloading and Extracting Packages  torchvision-0.5.0 | 6.4 MB | #################################### | 100%  wheel-0.34.1 | 65 KB | #################################### | 100%  six-1.14.0 | 27 KB | #################################### | 100%  docopt-0.6.2 | 23 KB | #################################### | 100%  scipy-1.3.2 | 11.2 MB | #################################### | 100%  numpy-base-1.18.1 | 3.8 MB | #################################### | 100%  pip-20.0.2 | 1.7 MB | #################################### | 100%  tqdm-4.42.0 | 55 KB | #################################### | 100%  pytorch-1.4.0 | 472.8 MB | #################################### | 100%  nltk-3.4.5 | 1.7 MB | #################################### | 100%  cudatoolkit-10.1.243 | 300.3 MB | #################################### | 100%  numpy-1.18.1 | 6 KB | #################################### | 100%  setuptools-45.1.0 | 536 KB | #################################### | 100%  Preparing transaction: done  Verifying transaction: done  Executing transaction: done  #  # To activate this environment, use  #  # $ conda activate local\_nmt  #  # To deactivate an active environment, use  #  # $ conda deactivate |

Note: The conda env is located in C:\ProgramData\Anaconda3\envs\local\_nmt.

* Activate the virtual environment using conda

(base) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>conda activate local\_nmt

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master> |

* Check all the python packages in this virtual environment

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>conda list  # packages in environment at C:\ProgramData\Anaconda3\envs\local\_nmt:  #  # Name Version Build Channel  blas 1.0 mkl  certifi 2019.11.28 py36\_0  cffi 1.13.2 py36h7a1dbc1\_0  cudatoolkit 10.1.243 h74a9793\_0  docopt 0.6.2 py36\_0  freetype 2.9.1 ha9979f8\_1  icc\_rt 2019.0.0 h0cc432a\_1  intel-openmp 2019.4 245  jpeg 9b hb83a4c4\_2  libpng 1.6.37 h2a8f88b\_0  libtiff 4.1.0 h56a325e\_0  mkl 2019.4 245  mkl-service 2.3.0 py36hb782905\_0  mkl\_fft 1.0.15 py36h14836fe\_0  mkl\_random 1.1.0 py36h675688f\_0  ninja 1.9.0 py36h74a9793\_0  nltk 3.4.5 py36\_0  numpy 1.18.1 py36h93ca92e\_0  numpy-base 1.18.1 py36hc3f5095\_1  olefile 0.46 py36\_0  pillow 7.0.0 py36hcc1f983\_0  pip 20.0.2 py36\_1  pycparser 2.19 py36\_0  python 3.6.10 h9f7ef89\_0  pytorch 1.4.0 py3.6\_cuda101\_cudnn7\_0 pytorch  scipy 1.3.2 py36h29ff71c\_0  setuptools 45.1.0 py36\_0  six 1.14.0 py36\_0  sqlite 3.30.1 he774522\_0  tk 8.6.8 hfa6e2cd\_0  torchvision 0.5.0 py36\_cu101 pytorch  tqdm 4.42.0 py\_0  vc 14.1 h0510ff6\_4  vs2015\_runtime 14.16.27012 hf0eaf9b\_1  wheel 0.34.1 py36\_0  wincertstore 0.2 py36h7fe50ca\_0  xz 5.2.4 h2fa13f4\_4  zlib 1.2.11 h62dcd97\_3  zstd 1.3.7 h508b16e\_0 |

* Specified in local\_env.yml

|  |
| --- |
| name: local\_nmt  channels:  - soumith  - defaults  dependencies:  - python=3.6  - numpy=1.18.1  - scipy=1.3.2  - tqdm=4.41.1  - docopt=0.6.2  - pytorch=1.4.0  - nltk=3.4.5  - torchvision=0.5.0 |

* create the correct vocab files

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python vocab.py --train-src=./en\_es\_data/train\_tiny.es --train-tgt=./en\_es\_data/train\_tiny.en --size=200 --freq-cutoff=1 vocab\_tiny\_q1.json  read in source sentences: ./en\_es\_data/train\_tiny.es  read in target sentences: ./en\_es\_data/train\_tiny.en  initialize source vocabulary ..  number of word types: 128, number of word types w/ frequency >= 1: 128  initialize target vocabulary ..  number of word types: 130, number of word types w/ frequency >= 1: 130  generated vocabulary, source 132 words, target 132 words  vocabulary saved to vocab\_tiny\_q1.json  (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python vocab.py --train-src=./en\_es\_data/train\_tiny.es --train-tgt=./en\_es\_data/train\_tiny.en vocab\_tiny\_q2.json  read in source sentences: ./en\_es\_data/train\_tiny.es  read in target sentences: ./en\_es\_data/train\_tiny.en  initialize source vocabulary ..  number of word types: 128, number of word types w/ frequency >= 2: 22  initialize target vocabulary ..  number of word types: 130, number of word types w/ frequency >= 2: 30  generated vocabulary, source 26 words, target 32 words  vocabulary saved to vocab\_tiny\_q2.json  (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python vocab.py --train-src=./en\_es\_data/train.es --train-tgt=./en\_es\_data/train.en vocab.json  read in source sentences: ./en\_es\_data/train.es  read in target sentences: ./en\_es\_data/train.en  initialize source vocabulary ..  number of word types: 172418, number of word types w/ frequency >= 2: 80623  initialize target vocabulary ..  number of word types: 128873, number of word types w/ frequency >= 2: 64215  generated vocabulary, source 50004 words, target 50002 words  vocabulary saved to vocab.json |

# Environment Setup End

# Coding/Testing Start

## 1. Character-based convolutional encoder for NMT (27 points)

1. (1 point) (coding) In vocab.py, implement the method words2charindices() (hint: copy the structure of words2indices()) to convert each character to its corresponding index in the character-vocabulary. This corresponds to the first two steps of Figure 2 (splitting and vocab lookup). Run the following for a non-exhaustive sanity check:

python sanity\_check.py 1a

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1a  --------------------------------------------------------------------------------  Running Sanity Check for Question 1a: words2charindices()  --------------------------------------------------------------------------------  Running test on small list of sentences  Running test on large list of sentences  All Sanity Checks Passed for Question 1a: words2charindices()!  -------------------------------------------------------------------------------- |

1. (4 points) (coding) Implement pad\_sents\_char() in utils.py, similar to the version for words. This method should pad at the character and word level. All words should be padded/truncated to max word length , and all sentences should be padded to the length of the longest sentence in the batch. A padding word is represented by <PAD>-characters. Run the following for a non-exhaustive sanity check:

python sanity\_check.py 1b

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1b  --------------------------------------------------------------------------------  Running Sanity Check for Question 1b: Padding  --------------------------------------------------------------------------------  Running test on a list of sentences  Sanity Check Passed for Question 1b: Padding!  -------------------------------------------------------------------------------- |

1. (3 points) (coding) Implement to\_input\_tensor\_char() in vocab.py to connect the previous two parts: use both methods created in the previous steps and convert the resulting padded sentences to a torch tensor. Ensure you reshape the dimensions so that the output has shape: (max sentence length, batch size, max word length) so that we don't have any shape errors downstream!

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1c  --------------------------------------------------------------------------------  Running Sanity Check for Question 1c: to\_input\_tensor\_char  --------------------------------------------------------------------------------  Running test on a list of sentences  Sanity Check Passed for Question 1c: to\_input\_tensor\_char!  -------------------------------------------------------------------------------- |

1. (coding) In the empty file highway.py, implement the highway network as a nn.Module class called Highway.

* Your module will need a \_init\_() and a forward() function (whose inputs and outputs you decide for yourself).
* The forward() function will need to map from to .
* Note that although the model description above is not batched, your forward() function should operate on batches of words.
* Make sure that your module uses two nn.Linear layers.

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1d  --------------------------------------------------------------------------------  Running Sanity Check for Question 1d: Highway  --------------------------------------------------------------------------------  Running test on a random tensor  Sanity Check Passed for Question 1d: Highway!  -------------------------------------------------------------------------------- |

1. (coding) In the empty file cnn.py, implement the convolutional network as a nn.Module class called CNN.

* Your module will need a \_init\_() and a forward() function (whose inputs and outputs you decide for yourself).
* The forward() function will need to map from to .
* Note that although the model description above is not batched, your forward() function should operate on batches of words.
* Make sure that your module uses one nn.Conv1d layer (this is important for the autograder).
* Use a kernel size of k = 5.

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1e  --------------------------------------------------------------------------------  Running Sanity Check for Question 1e: CNN  --------------------------------------------------------------------------------  Running test on a random tensor  Sanity Check Passed for Question 1e: CNN!  -------------------------------------------------------------------------------- |

1. (12 points) (coding) Write the \_\_init\_\_() and forward() functions for the ModelEmbeddings class in model\_embeddings.py.:

* The forward() function must map from to - note this is for whole batches of sentences, rather than batches of words.
* You will need to use the CNN and Highway modules you created.
* Don't forget the dropout layer! Use 0.3 dropout probability.
* Your ModelEmbeddings class should contain one nn.Embedding object (this is important for the autograder).
* Remember that we are using = 50.
* Depending on your implementation of CNN and Highway, it's likely that you will need to reshape tensors to get them into the shape required by CNN and Highway, and then reshape again to get the final output of ModelEmbeddings.forward().

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 1f  --------------------------------------------------------------------------------  Running Sanity Check for Question 1f: Model Embedding  --------------------------------------------------------------------------------  Sanity Check Passed for Question 1f: Model Embedding!  -------------------------------------------------------------------------------- |

1. (2 points) (coding) In nmt\_model.py, complete the forward()method to use character-level padded encodings instead of word-level encodings. This ties together your ModelEmbeddings code with the preprocessing code you wrote. Check your code!
2. (5 points) (coding) On your local machine, confirm that you're in the proper conda environment then execute the following command. This will train your model on a very small subset of the training data, then test it. Your model should overfit the training data.
   1. Train
   2. Create an empty file in the outputs subfolder
   3. Test

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py train --train-src=./en\_es\_data/train\_tiny.es --train-tgt=./en\_es\_data/train\_tiny.en --dev-src=./en\_es\_data/dev\_tiny.es --dev-tgt=./en\_es\_data/dev\_tiny.en --vocab=vocab\_tiny\_q1.json --batch-size=2 --valid-niter=100 --max-epoch=101 --no-char-decoder  uniformly initialize parameters [-0.100000, +0.100000]  use device: cpu  begin Maximum Likelihood training  epoch 2, iter 10, avg. loss 96.08, avg. ppl 121.98 cum. examples 20, speed 162.07 words/sec, time elapsed 2.47 sec  epoch 4, iter 20, avg. loss 91.40, avg. ppl 96.54 cum. examples 40, speed 184.08 words/sec, time elapsed 4.64 sec  epoch 6, iter 30, avg. loss 87.19, avg. ppl 78.22 cum. examples 60, speed 178.09 words/sec, time elapsed 6.89 sec  epoch 8, iter 40, avg. loss 83.54, avg. ppl 65.15 cum. examples 80, speed 203.56 words/sec, time elapsed 8.85 sec  epoch 10, iter 50, avg. loss 80.05, avg. ppl 54.75 cum. examples 100, speed 202.12 words/sec, time elapsed 10.83 sec  epoch 12, iter 60, avg. loss 77.25, avg. ppl 47.57 cum. examples 120, speed 181.08 words/sec, time elapsed 13.04 sec  epoch 14, iter 70, avg. loss 73.44, avg. ppl 39.33 cum. examples 140, speed 178.17 words/sec, time elapsed 15.29 sec  epoch 16, iter 80, avg. loss 70.14, avg. ppl 33.35 cum. examples 160, speed 187.88 words/sec, time elapsed 17.41 sec  epoch 18, iter 90, avg. loss 66.65, avg. ppl 28.01 cum. examples 180, speed 191.39 words/sec, time elapsed 19.50 sec  epoch 20, iter 100, avg. loss 63.19, avg. ppl 23.55 cum. examples 200, speed 174.14 words/sec, time elapsed 21.80 sec  epoch 20, iter 100, cum. loss 78.89, cum. ppl 51.66 cum. examples 200  begin validation ...  validation: iter 100, dev. ppl 26.937010  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 22, iter 110, avg. loss 59.85, avg. ppl 19.94 cum. examples 20, speed 173.09 words/sec, time elapsed 24.11 sec  epoch 24, iter 120, avg. loss 55.90, avg. ppl 16.36 cum. examples 40, speed 180.26 words/sec, time elapsed 26.33 sec  epoch 26, iter 130, avg. loss 52.61, avg. ppl 13.88 cum. examples 60, speed 184.50 words/sec, time elapsed 28.50 sec  epoch 28, iter 140, avg. loss 49.57, avg. ppl 11.92 cum. examples 80, speed 195.03 words/sec, time elapsed 30.55 sec  epoch 30, iter 150, avg. loss 47.52, avg. ppl 10.76 cum. examples 100, speed 205.23 words/sec, time elapsed 32.50 sec  epoch 32, iter 160, avg. loss 41.00, avg. ppl 7.77 cum. examples 120, speed 185.79 words/sec, time elapsed 34.65 sec  epoch 34, iter 170, avg. loss 37.68, avg. ppl 6.58 cum. examples 140, speed 178.65 words/sec, time elapsed 36.89 sec  epoch 36, iter 180, avg. loss 34.46, avg. ppl 5.60 cum. examples 160, speed 187.44 words/sec, time elapsed 39.03 sec  epoch 38, iter 190, avg. loss 37.22, avg. ppl 6.43 cum. examples 180, speed 181.16 words/sec, time elapsed 41.23 sec  epoch 40, iter 200, avg. loss 32.21, avg. ppl 5.01 cum. examples 200, speed 179.29 words/sec, time elapsed 43.47 sec  epoch 40, iter 200, cum. loss 44.80, cum. ppl 9.39 cum. examples 200  begin validation ...  validation: iter 200, dev. ppl 3.841558  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 42, iter 210, avg. loss 27.03, avg. ppl 3.86 cum. examples 20, speed 182.65 words/sec, time elapsed 45.66 sec  epoch 44, iter 220, avg. loss 23.99, avg. ppl 3.32 cum. examples 40, speed 179.61 words/sec, time elapsed 47.88 sec  epoch 46, iter 230, avg. loss 19.28, avg. ppl 2.62 cum. examples 60, speed 183.65 words/sec, time elapsed 50.06 sec  epoch 48, iter 240, avg. loss 16.92, avg. ppl 2.33 cum. examples 80, speed 188.15 words/sec, time elapsed 52.19 sec  epoch 50, iter 250, avg. loss 14.93, avg. ppl 2.11 cum. examples 100, speed 187.71 words/sec, time elapsed 54.32 sec  epoch 52, iter 260, avg. loss 12.24, avg. ppl 1.84 cum. examples 120, speed 194.55 words/sec, time elapsed 56.37 sec  epoch 54, iter 270, avg. loss 10.36, avg. ppl 1.68 cum. examples 140, speed 173.99 words/sec, time elapsed 58.67 sec  epoch 56, iter 280, avg. loss 9.32, avg. ppl 1.59 cum. examples 160, speed 181.90 words/sec, time elapsed 60.87 sec  epoch 58, iter 290, avg. loss 7.22, avg. ppl 1.43 cum. examples 180, speed 191.30 words/sec, time elapsed 62.96 sec  epoch 60, iter 300, avg. loss 5.74, avg. ppl 1.33 cum. examples 200, speed 189.04 words/sec, time elapsed 65.08 sec  epoch 60, iter 300, cum. loss 14.70, cum. ppl 2.09 cum. examples 200  begin validation ...  validation: iter 300, dev. ppl 1.274550  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 62, iter 310, avg. loss 5.15, avg. ppl 1.29 cum. examples 20, speed 170.94 words/sec, time elapsed 67.42 sec  epoch 64, iter 320, avg. loss 5.42, avg. ppl 1.31 cum. examples 40, speed 173.39 words/sec, time elapsed 69.73 sec  epoch 66, iter 330, avg. loss 3.73, avg. ppl 1.21 cum. examples 60, speed 185.19 words/sec, time elapsed 71.89 sec  epoch 68, iter 340, avg. loss 3.24, avg. ppl 1.18 cum. examples 80, speed 176.13 words/sec, time elapsed 74.16 sec  epoch 70, iter 350, avg. loss 2.26, avg. ppl 1.12 cum. examples 100, speed 176.44 words/sec, time elapsed 76.43 sec  epoch 72, iter 360, avg. loss 2.17, avg. ppl 1.11 cum. examples 120, speed 189.13 words/sec, time elapsed 78.54 sec  epoch 74, iter 370, avg. loss 1.52, avg. ppl 1.08 cum. examples 140, speed 196.37 words/sec, time elapsed 80.58 sec  epoch 76, iter 380, avg. loss 1.02, avg. ppl 1.05 cum. examples 160, speed 177.70 words/sec, time elapsed 82.83 sec  epoch 78, iter 390, avg. loss 0.88, avg. ppl 1.04 cum. examples 180, speed 188.06 words/sec, time elapsed 84.96 sec  epoch 80, iter 400, avg. loss 0.68, avg. ppl 1.03 cum. examples 200, speed 195.60 words/sec, time elapsed 87.00 sec  epoch 80, iter 400, cum. loss 2.61, cum. ppl 1.14 cum. examples 200  begin validation ...  validation: iter 400, dev. ppl 1.019305  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 82, iter 410, avg. loss 0.52, avg. ppl 1.03 cum. examples 20, speed 190.11 words/sec, time elapsed 89.11 sec  epoch 84, iter 420, avg. loss 0.45, avg. ppl 1.02 cum. examples 40, speed 179.61 words/sec, time elapsed 91.33 sec  epoch 86, iter 430, avg. loss 0.70, avg. ppl 1.04 cum. examples 60, speed 196.37 words/sec, time elapsed 93.37 sec  epoch 88, iter 440, avg. loss 0.56, avg. ppl 1.03 cum. examples 80, speed 189.04 words/sec, time elapsed 95.49 sec  epoch 90, iter 450, avg. loss 0.33, avg. ppl 1.02 cum. examples 100, speed 196.85 words/sec, time elapsed 97.52 sec  epoch 92, iter 460, avg. loss 0.30, avg. ppl 1.02 cum. examples 120, speed 192.86 words/sec, time elapsed 99.59 sec  epoch 94, iter 470, avg. loss 0.23, avg. ppl 1.01 cum. examples 140, speed 182.65 words/sec, time elapsed 101.79 sec  epoch 96, iter 480, avg. loss 0.30, avg. ppl 1.01 cum. examples 160, speed 175.90 words/sec, time elapsed 104.06 sec  epoch 98, iter 490, avg. loss 0.19, avg. ppl 1.01 cum. examples 180, speed 207.90 words/sec, time elapsed 105.98 sec  epoch 100, iter 500, avg. loss 0.18, avg. ppl 1.01 cum. examples 200, speed 183.91 words/sec, time elapsed 108.16 sec  epoch 100, iter 500, cum. loss 0.38, cum. ppl 1.02 cum. examples 200  begin validation ...  validation: iter 500, dev. ppl 1.004388  save currently the best model to [model.bin]  save model parameters to [model.bin]  reached maximum number of epochs!  (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py decode model.bin ./en\_es\_data/test\_tiny.es ./en\_es\_data/test\_tiny.en outputs/test\_outputs\_local\_q1.txt --no-char-decoder  load test source sentences from [./en\_es\_data/test\_tiny.es]  load test target sentences from [./en\_es\_data/test\_tiny.en]  load model from model.bin  Decoding: 100%|████████████████████████████████████████████████████████| 4/4 [00:00<00:00, 15.33it/s]  Corpus BLEU: 99.29792465574434 |

## 2. Character-based LSTM decoder for NMT (26 points)

1. (2 points) (coding)Write the \_\_init\_\_ function for the CharDecoder module in char\_decoder.py. Run the following for a non-exhaustive sanity check:

python sanity\_check.py 2a

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 2a  --------------------------------------------------------------------------------  Running Sanity Check for Question 2a: CharDecoder.\_\_init\_\_()  --------------------------------------------------------------------------------  Sanity Check Passed for Question 2a: CharDecoder.\_\_init\_\_()!  -------------------------------------------------------------------------------- |

1. (3 points) (coding) Write the forward() function in char\_decoder.py. This function takes input x1, …, xn and (h0; c0) (as described in the **Forward computation of the character decoder** section) and returns s1, …, sn and (hn; cn). Run the following for a non-exhaustive sanity check:

python sanity\_check.py 2b

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 2b  --------------------------------------------------------------------------------  Running Sanity Check for Question 2b: CharDecoder.forward()  --------------------------------------------------------------------------------  Sanity Check Passed for Question 2b: CharDecoder.forward()!  -------------------------------------------------------------------------------- |

1. (5 points) (coding) Write the train\_forward() function in char\_decoder.py. This function computes losschar dec summed across the whole batch. Hint: Carefully read the documentation for nn.CrossEntropyLoss. Run the following for a non-exhaustive sanity check:

python sanity\_check.py 2c

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 2c  --------------------------------------------------------------------------------  Running Sanity Check for Question 2c: CharDecoder.train\_forward()  --------------------------------------------------------------------------------  Sanity Check Passed for Question 2c: CharDecoder.train\_forward()!  -------------------------------------------------------------------------------- |

1. (7 points) (coding) Write the decode\_greedy() function in char\_decoder.py to implement the algorithm DECODE\_GREEDY. Note that although Algorithm 1 is described for a single example, your implementation must work over a batch. Algorithm 1 also indicates that you should break when you reach the <END> token, but in the batched case you might find it more convenient to complete all max\_length steps of the for-loop, then truncate the output\_words afterwards. Run the following for a non-exhaustive sanity check:

python sanity\_check.py 2d

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python sanity\_check.py 2d  --------------------------------------------------------------------------------  Running Sanity Check for Question 2d: CharDecoder.decode\_greedy()  --------------------------------------------------------------------------------  Sanity Check Passed for Question 2d: CharDecoder.decode\_greedy()!  -------------------------------------------------------------------------------- |

1. (3 points) (coding) Once you have thoroughly checked your implementation of the CharDecoder functions (checking much more than just the sanity checks!), it's time to do the 'small training run' check. Confirm that you're in the proper conda environment and then execute the following command on your local machine to check if your model overfits to a small subset of the training data.

Running these should take around 10 minutes (but this depends on your local machine). You should observe the average loss go down to near 0 and average perplexity on train and dev set go to 1 during training. Once you run the test, you should observe BLEU score on the test set higher than 99.00. If you don't observe these things, you probably need to go back to debug!

*Important: Make sure not to modify the output file (outputs/test\_outputs\_local\_q2.txt) generated by the code - you need this to be included when you run the submission script.*

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py train --train-src=./en\_es\_data/train\_tiny.es --train-tgt=./en\_es\_data/train\_tiny.en --dev-src=./en\_es\_data/dev\_tiny.es --dev-tgt=./en\_es\_data/dev\_tiny.en --vocab=vocab\_tiny\_q2.json --batch-size=2 --max-epoch=201 --valid-niter=100  uniformly initialize parameters [-0.100000, +0.100000]  use device: cpu  begin Maximum Likelihood training  epoch 2, iter 10, avg. loss 454.31, avg. ppl 7331245264.01 cum. examples 20, speed 130.34 words/sec, time elapsed 3.07 sec  epoch 4, iter 20, avg. loss 353.59, avg. ppl 47656760.90 cum. examples 40, speed 144.14 words/sec, time elapsed 5.84 sec  epoch 6, iter 30, avg. loss 335.87, avg. ppl 19650100.03 cum. examples 60, speed 145.40 words/sec, time elapsed 8.59 sec  epoch 8, iter 40, avg. loss 329.99, avg. ppl 14643161.40 cum. examples 80, speed 164.95 words/sec, time elapsed 11.02 sec  epoch 10, iter 50, avg. loss 323.97, avg. ppl 10835662.22 cum. examples 100, speed 157.60 words/sec, time elapsed 13.56 sec  epoch 12, iter 60, avg. loss 320.90, avg. ppl 9294327.78 cum. examples 120, speed 146.74 words/sec, time elapsed 16.28 sec  epoch 14, iter 70, avg. loss 316.81, avg. ppl 7577105.49 cum. examples 140, speed 140.60 words/sec, time elapsed 19.13 sec  epoch 16, iter 80, avg. loss 313.37, avg. ppl 6377733.16 cum. examples 160, speed 150.49 words/sec, time elapsed 21.79 sec  epoch 18, iter 90, avg. loss 306.77, avg. ppl 4585909.78 cum. examples 180, speed 155.76 words/sec, time elapsed 24.36 sec  epoch 20, iter 100, avg. loss 302.16, avg. ppl 3642595.99 cum. examples 200, speed 141.64 words/sec, time elapsed 27.18 sec  epoch 20, iter 100, cum. loss 335.77, cum. ppl 19554114.40 cum. examples 200  begin validation ...  validation: iter 100, dev. ppl 4478406.813806  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 22, iter 110, avg. loss 294.55, avg. ppl 2489337.95 cum. examples 20, speed 138.79 words/sec, time elapsed 30.06 sec  epoch 24, iter 120, avg. loss 285.40, avg. ppl 1575547.41 cum. examples 40, speed 142.76 words/sec, time elapsed 32.86 sec  epoch 26, iter 130, avg. loss 274.74, avg. ppl 924468.71 cum. examples 60, speed 139.66 words/sec, time elapsed 35.73 sec  epoch 28, iter 140, avg. loss 265.02, avg. ppl 568564.66 cum. examples 80, speed 154.02 words/sec, time elapsed 38.32 sec  epoch 30, iter 150, avg. loss 257.04, avg. ppl 381583.86 cum. examples 100, speed 165.02 words/sec, time elapsed 40.75 sec  epoch 32, iter 160, avg. loss 244.98, avg. ppl 208824.42 cum. examples 120, speed 149.81 words/sec, time elapsed 43.42 sec  epoch 34, iter 170, avg. loss 235.06, avg. ppl 127109.92 cum. examples 140, speed 140.45 words/sec, time elapsed 46.27 sec  epoch 36, iter 180, avg. loss 225.16, avg. ppl 77514.86 cum. examples 160, speed 151.52 words/sec, time elapsed 48.91 sec  epoch 38, iter 190, avg. loss 215.00, avg. ppl 46620.27 cum. examples 180, speed 144.87 words/sec, time elapsed 51.67 sec  epoch 40, iter 200, avg. loss 204.75, avg. ppl 27932.40 cum. examples 200, speed 140.65 words/sec, time elapsed 54.51 sec  epoch 40, iter 200, cum. loss 250.17, cum. ppl 270632.91 cum. examples 200  begin validation ...  validation: iter 200, dev. ppl 28120.955500  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 42, iter 210, avg. loss 193.94, avg. ppl 16267.50 cum. examples 20, speed 151.23 words/sec, time elapsed 57.16 sec  epoch 44, iter 220, avg. loss 184.98, avg. ppl 10393.65 cum. examples 40, speed 145.61 words/sec, time elapsed 59.90 sec  epoch 46, iter 230, avg. loss 175.33, avg. ppl 6414.18 cum. examples 60, speed 150.49 words/sec, time elapsed 62.56 sec  epoch 48, iter 240, avg. loss 164.22, avg. ppl 3681.46 cum. examples 80, speed 152.26 words/sec, time elapsed 65.19 sec  epoch 50, iter 250, avg. loss 161.06, avg. ppl 3143.97 cum. examples 100, speed 153.49 words/sec, time elapsed 67.80 sec  epoch 52, iter 260, avg. loss 149.65, avg. ppl 1776.41 cum. examples 120, speed 157.17 words/sec, time elapsed 70.34 sec  epoch 54, iter 270, avg. loss 142.71, avg. ppl 1255.96 cum. examples 140, speed 139.96 words/sec, time elapsed 73.20 sec  epoch 56, iter 280, avg. loss 131.67, avg. ppl 722.95 cum. examples 160, speed 145.30 words/sec, time elapsed 75.95 sec  epoch 58, iter 290, avg. loss 123.11, avg. ppl 471.37 cum. examples 180, speed 153.02 words/sec, time elapsed 78.57 sec  epoch 60, iter 300, avg. loss 113.84, avg. ppl 296.51 cum. examples 200, speed 153.14 words/sec, time elapsed 81.18 sec  epoch 60, iter 300, cum. loss 154.05, cum. ppl 2213.99 cum. examples 200  begin validation ...  validation: iter 300, dev. ppl 368.119853  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 62, iter 310, avg. loss 107.71, avg. ppl 218.20 cum. examples 20, speed 138.79 words/sec, time elapsed 84.06 sec  epoch 64, iter 320, avg. loss 102.45, avg. ppl 167.71 cum. examples 40, speed 139.86 words/sec, time elapsed 86.92 sec  epoch 66, iter 330, avg. loss 95.61, avg. ppl 119.19 cum. examples 60, speed 150.32 words/sec, time elapsed 89.58 sec  epoch 68, iter 340, avg. loss 87.36, avg. ppl 78.89 cum. examples 80, speed 145.30 words/sec, time elapsed 92.34 sec  epoch 70, iter 350, avg. loss 80.91, avg. ppl 57.13 cum. examples 100, speed 140.75 words/sec, time elapsed 95.18 sec  epoch 72, iter 360, avg. loss 73.48, avg. ppl 39.42 cum. examples 120, speed 153.37 words/sec, time elapsed 97.79 sec  epoch 74, iter 370, avg. loss 67.18, avg. ppl 28.76 cum. examples 140, speed 156.86 words/sec, time elapsed 100.34 sec  epoch 76, iter 380, avg. loss 61.23, avg. ppl 21.36 cum. examples 160, speed 141.39 words/sec, time elapsed 103.17 sec  epoch 78, iter 390, avg. loss 57.62, avg. ppl 17.83 cum. examples 180, speed 151.34 words/sec, time elapsed 105.81 sec  epoch 80, iter 400, avg. loss 51.73, avg. ppl 13.29 cum. examples 200, speed 158.23 words/sec, time elapsed 108.34 sec  epoch 80, iter 400, cum. loss 78.53, cum. ppl 50.72 cum. examples 200  begin validation ...  validation: iter 400, dev. ppl 12.893379  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 82, iter 410, avg. loss 46.93, avg. ppl 10.45 cum. examples 20, speed 149.48 words/sec, time elapsed 111.01 sec  epoch 84, iter 420, avg. loss 43.40, avg. ppl 8.76 cum. examples 40, speed 138.36 words/sec, time elapsed 113.90 sec  epoch 86, iter 430, avg. loss 40.41, avg. ppl 7.54 cum. examples 60, speed 153.67 words/sec, time elapsed 116.51 sec  epoch 88, iter 440, avg. loss 34.23, avg. ppl 5.54 cum. examples 80, speed 154.20 words/sec, time elapsed 119.10 sec  epoch 90, iter 450, avg. loss 30.60, avg. ppl 4.62 cum. examples 100, speed 154.26 words/sec, time elapsed 121.69 sec  epoch 92, iter 460, avg. loss 27.80, avg. ppl 4.02 cum. examples 120, speed 154.86 words/sec, time elapsed 124.28 sec  epoch 94, iter 470, avg. loss 24.25, avg. ppl 3.36 cum. examples 140, speed 142.30 words/sec, time elapsed 127.09 sec  epoch 96, iter 480, avg. loss 21.88, avg. ppl 2.99 cum. examples 160, speed 137.84 words/sec, time elapsed 129.99 sec  epoch 98, iter 490, avg. loss 18.78, avg. ppl 2.56 cum. examples 180, speed 164.95 words/sec, time elapsed 132.42 sec  epoch 100, iter 500, avg. loss 16.30, avg. ppl 2.26 cum. examples 200, speed 151.06 words/sec, time elapsed 135.06 sec  epoch 100, iter 500, cum. loss 30.46, cum. ppl 4.59 cum. examples 200  begin validation ...  validation: iter 500, dev. ppl 2.010414  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 102, iter 510, avg. loss 15.46, avg. ppl 2.17 cum. examples 20, speed 140.65 words/sec, time elapsed 137.91 sec  epoch 104, iter 520, avg. loss 13.05, avg. ppl 1.92 cum. examples 40, speed 142.86 words/sec, time elapsed 140.71 sec  epoch 106, iter 530, avg. loss 11.52, avg. ppl 1.78 cum. examples 60, speed 143.88 words/sec, time elapsed 143.49 sec  epoch 108, iter 540, avg. loss 9.99, avg. ppl 1.65 cum. examples 80, speed 147.98 words/sec, time elapsed 146.19 sec  epoch 110, iter 550, avg. loss 9.09, avg. ppl 1.58 cum. examples 100, speed 140.94 words/sec, time elapsed 149.03 sec  epoch 112, iter 560, avg. loss 8.31, avg. ppl 1.52 cum. examples 120, speed 143.47 words/sec, time elapsed 151.82 sec  epoch 114, iter 570, avg. loss 6.78, avg. ppl 1.40 cum. examples 140, speed 157.42 words/sec, time elapsed 154.36 sec  epoch 116, iter 580, avg. loss 6.21, avg. ppl 1.36 cum. examples 160, speed 138.46 words/sec, time elapsed 157.25 sec  epoch 118, iter 590, avg. loss 5.59, avg. ppl 1.32 cum. examples 180, speed 147.44 words/sec, time elapsed 159.96 sec  epoch 120, iter 600, avg. loss 4.55, avg. ppl 1.26 cum. examples 200, speed 150.04 words/sec, time elapsed 162.63 sec  epoch 120, iter 600, cum. loss 9.06, cum. ppl 1.57 cum. examples 200  begin validation ...  validation: iter 600, dev. ppl 1.156141  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 122, iter 610, avg. loss 4.39, avg. ppl 1.25 cum. examples 20, speed 139.37 words/sec, time elapsed 165.50 sec  epoch 124, iter 620, avg. loss 3.71, avg. ppl 1.20 cum. examples 40, speed 150.49 words/sec, time elapsed 168.16 sec  epoch 126, iter 630, avg. loss 3.12, avg. ppl 1.17 cum. examples 60, speed 153.55 words/sec, time elapsed 170.76 sec  epoch 128, iter 640, avg. loss 2.68, avg. ppl 1.14 cum. examples 80, speed 144.30 words/sec, time elapsed 173.53 sec  epoch 130, iter 650, avg. loss 2.49, avg. ppl 1.13 cum. examples 100, speed 148.48 words/sec, time elapsed 176.23 sec  epoch 132, iter 660, avg. loss 2.22, avg. ppl 1.12 cum. examples 120, speed 164.61 words/sec, time elapsed 178.66 sec  epoch 134, iter 670, avg. loss 2.02, avg. ppl 1.11 cum. examples 140, speed 143.16 words/sec, time elapsed 181.45 sec  epoch 136, iter 680, avg. loss 2.17, avg. ppl 1.11 cum. examples 160, speed 148.81 words/sec, time elapsed 184.14 sec  epoch 138, iter 690, avg. loss 1.84, avg. ppl 1.10 cum. examples 180, speed 143.57 words/sec, time elapsed 186.93 sec  epoch 140, iter 700, avg. loss 1.52, avg. ppl 1.08 cum. examples 200, speed 141.54 words/sec, time elapsed 189.75 sec  epoch 140, iter 700, cum. loss 2.62, cum. ppl 1.14 cum. examples 200  begin validation ...  validation: iter 700, dev. ppl 1.038166  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 142, iter 710, avg. loss 1.21, avg. ppl 1.06 cum. examples 20, speed 144.56 words/sec, time elapsed 192.52 sec  epoch 144, iter 720, avg. loss 1.07, avg. ppl 1.05 cum. examples 40, speed 162.14 words/sec, time elapsed 194.99 sec  epoch 146, iter 730, avg. loss 0.98, avg. ppl 1.05 cum. examples 60, speed 142.10 words/sec, time elapsed 197.80 sec  epoch 148, iter 740, avg. loss 0.87, avg. ppl 1.04 cum. examples 80, speed 158.60 words/sec, time elapsed 200.32 sec  epoch 150, iter 750, avg. loss 0.82, avg. ppl 1.04 cum. examples 100, speed 147.66 words/sec, time elapsed 203.03 sec  epoch 152, iter 760, avg. loss 0.71, avg. ppl 1.04 cum. examples 120, speed 154.38 words/sec, time elapsed 205.62 sec  epoch 154, iter 770, avg. loss 0.61, avg. ppl 1.03 cum. examples 140, speed 149.81 words/sec, time elapsed 208.29 sec  epoch 156, iter 780, avg. loss 0.56, avg. ppl 1.03 cum. examples 160, speed 138.99 words/sec, time elapsed 211.17 sec  epoch 158, iter 790, avg. loss 0.51, avg. ppl 1.03 cum. examples 180, speed 145.30 words/sec, time elapsed 213.92 sec  epoch 160, iter 800, avg. loss 0.48, avg. ppl 1.02 cum. examples 200, speed 152.85 words/sec, time elapsed 216.54 sec  epoch 160, iter 800, cum. loss 0.78, cum. ppl 1.04 cum. examples 200  begin validation ...  validation: iter 800, dev. ppl 1.010369  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 162, iter 810, avg. loss 0.41, avg. ppl 1.02 cum. examples 20, speed 133.82 words/sec, time elapsed 219.53 sec  epoch 164, iter 820, avg. loss 0.37, avg. ppl 1.02 cum. examples 40, speed 145.45 words/sec, time elapsed 222.28 sec  epoch 166, iter 830, avg. loss 0.34, avg. ppl 1.02 cum. examples 60, speed 152.44 words/sec, time elapsed 224.91 sec  epoch 168, iter 840, avg. loss 0.32, avg. ppl 1.02 cum. examples 80, speed 151.72 words/sec, time elapsed 227.54 sec  epoch 170, iter 850, avg. loss 0.32, avg. ppl 1.02 cum. examples 100, speed 138.46 words/sec, time elapsed 230.43 sec  epoch 172, iter 860, avg. loss 0.28, avg. ppl 1.01 cum. examples 120, speed 152.79 words/sec, time elapsed 233.05 sec  epoch 174, iter 870, avg. loss 0.24, avg. ppl 1.01 cum. examples 140, speed 147.82 words/sec, time elapsed 235.75 sec  epoch 176, iter 880, avg. loss 0.24, avg. ppl 1.01 cum. examples 160, speed 143.68 words/sec, time elapsed 238.54 sec  epoch 178, iter 890, avg. loss 0.23, avg. ppl 1.01 cum. examples 180, speed 153.08 words/sec, time elapsed 241.15 sec  epoch 180, iter 900, avg. loss 0.22, avg. ppl 1.01 cum. examples 200, speed 148.04 words/sec, time elapsed 243.85 sec  epoch 180, iter 900, cum. loss 0.30, cum. ppl 1.01 cum. examples 200  begin validation ...  validation: iter 900, dev. ppl 1.004988  save currently the best model to [model.bin]  save model parameters to [model.bin]  epoch 182, iter 910, avg. loss 0.22, avg. ppl 1.01 cum. examples 20, speed 146.20 words/sec, time elapsed 246.59 sec  epoch 184, iter 920, avg. loss 0.20, avg. ppl 1.01 cum. examples 40, speed 157.67 words/sec, time elapsed 249.13 sec  epoch 186, iter 930, avg. loss 0.20, avg. ppl 1.01 cum. examples 60, speed 144.72 words/sec, time elapsed 251.89 sec  epoch 188, iter 940, avg. loss 0.17, avg. ppl 1.01 cum. examples 80, speed 148.98 words/sec, time elapsed 254.58 sec  epoch 190, iter 950, avg. loss 0.17, avg. ppl 1.01 cum. examples 100, speed 147.28 words/sec, time elapsed 257.29 sec  epoch 192, iter 960, avg. loss 0.17, avg. ppl 1.01 cum. examples 120, speed 147.11 words/sec, time elapsed 260.01 sec  epoch 194, iter 970, avg. loss 0.16, avg. ppl 1.01 cum. examples 140, speed 148.86 words/sec, time elapsed 262.70 sec  epoch 196, iter 980, avg. loss 0.16, avg. ppl 1.01 cum. examples 160, speed 146.90 words/sec, time elapsed 265.42 sec  epoch 198, iter 990, avg. loss 0.15, avg. ppl 1.01 cum. examples 180, speed 148.37 words/sec, time elapsed 268.12 sec  epoch 200, iter 1000, avg. loss 0.14, avg. ppl 1.01 cum. examples 200, speed 143.01 words/sec, time elapsed 270.92 sec  epoch 200, iter 1000, cum. loss 0.17, cum. ppl 1.01 cum. examples 200  begin validation ...  validation: iter 1000, dev. ppl 1.003154  save currently the best model to [model.bin]  save model parameters to [model.bin]  reached maximum number of epochs!  (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py decode model.bin ./en\_es\_data/test\_tiny.es ./en\_es\_data/test\_tiny.en outputs/test\_outputs\_local\_q2.txt  load test source sentences from [./en\_es\_data/test\_tiny.es]  load test target sentences from [./en\_es\_data/test\_tiny.en]  load model from model.bin  Decoding: 100%|██████████████████████████████████████████████████████████████████| 4/4 [00:00<00:00, 6.36it/s]  Corpus BLEU: 99.29792465574434 |

1. (6 points) (coding) Now that we've implemented both the character-based encoder and the character-based decoder, it's finally time to train the full system!

|  |
| --- |
| (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py train --train-src=./en\_es\_data/train.es --train-tgt=./en\_es\_data/train.en --dev-src=./en\_es\_data/dev.es --dev-tgt=./en\_es\_data/dev.en --vocab=vocab.json --cuda --batch-size=2  ......  epoch 4, iter 402560, avg. loss 101.70, avg. ppl 678.23 cum. examples 1120, speed 431.54 words/sec, time elapsed 30706.82 sec  epoch 4, iter 402570, avg. loss 149.18, avg. ppl 1048.01 cum. examples 1140, speed 512.54 words/sec, time elapsed 30707.66 sec  epoch 4, iter 402580, avg. loss 167.22, avg. ppl 1194.67 cum. examples 1160, speed 519.82 words/sec, time elapsed 30708.56 sec  epoch 4, iter 402590, avg. loss 135.71, avg. ppl 1265.02 cum. examples 1180, speed 477.99 words/sec, time elapsed 30709.36 sec  epoch 4, iter 402600, avg. loss 163.15, avg. ppl 1779.31 cum. examples 1200, speed 490.99 words/sec, time elapsed 30710.25 sec  epoch 4, iter 402610, avg. loss 117.06, avg. ppl 2326.29 cum. examples 1220, speed 421.79 words/sec, time elapsed 30710.96 sec  epoch 4, iter 402620, avg. loss 121.69, avg. ppl 2058.29 cum. examples 1240, speed 458.99 words/sec, time elapsed 30711.66 sec  epoch 4, iter 402630, avg. loss 131.68, avg. ppl 968.97 cum. examples 1260, speed 494.19 words/sec, time elapsed 30712.43 sec  epoch 4, iter 402640, avg. loss 151.12, avg. ppl 2231.14 cum. examples 1280, speed 495.58 words/sec, time elapsed 30713.23 sec  epoch 4, iter 402650, avg. loss 100.83, avg. ppl 1489.51 cum. examples 1300, speed 453.20 words/sec, time elapsed 30713.83 sec  epoch 4, iter 402660, avg. loss 99.10, avg. ppl 1501.15 cum. examples 1320, speed 417.57 words/sec, time elapsed 30714.48 sec  epoch 4, iter 402670, avg. loss 115.58, avg. ppl 896.73 cum. examples 1340, speed 423.41 words/sec, time elapsed 30715.29 sec  epoch 4, iter 402680, avg. loss 144.64, avg. ppl 2954.90 cum. examples 1360, speed 502.78 words/sec, time elapsed 30716.01 sec  epoch 4, iter 402690, avg. loss 120.93, avg. ppl 1366.13 cum. examples 1380, speed 417.71 words/sec, time elapsed 30716.81 sec  epoch 4, iter 402700, avg. loss 126.15, avg. ppl 967.50 cum. examples 1400, speed 458.75 words/sec, time elapsed 30717.61 sec  epoch 4, iter 402710, avg. loss 132.90, avg. ppl 1397.81 cum. examples 1420, speed 501.37 words/sec, time elapsed 30718.34 sec  epoch 4, iter 402720, avg. loss 160.88, avg. ppl 1576.16 cum. examples 1440, speed 534.88 words/sec, time elapsed 30719.16 sec  epoch 4, iter 402730, avg. loss 181.05, avg. ppl 1720.89 cum. examples 1460, speed 503.11 words/sec, time elapsed 30720.13 sec  epoch 4, iter 402740, avg. loss 116.32, avg. ppl 2964.79 cum. examples 1480, speed 453.98 words/sec, time elapsed 30720.77 sec  epoch 4, iter 402750, avg. loss 135.46, avg. ppl 400.99 cum. examples 1500, speed 541.97 words/sec, time elapsed 30721.60 sec  epoch 4, iter 402760, avg. loss 108.46, avg. ppl 979.12 cum. examples 1520, speed 437.50 words/sec, time elapsed 30722.32 sec  epoch 4, iter 402770, avg. loss 133.18, avg. ppl 1634.44 cum. examples 1540, speed 479.36 words/sec, time elapsed 30723.07 sec  epoch 4, iter 402780, avg. loss 134.42, avg. ppl 1117.88 cum. examples 1560, speed 481.16 words/sec, time elapsed 30723.87 sec  epoch 4, iter 402790, avg. loss 121.27, avg. ppl 1063.47 cum. examples 1580, speed 451.36 words/sec, time elapsed 30724.64 sec  epoch 4, iter 402800, avg. loss 140.38, avg. ppl 1441.47 cum. examples 1600, speed 491.09 words/sec, time elapsed 30725.43 sec  epoch 4, iter 402810, avg. loss 142.22, avg. ppl 5257.53 cum. examples 1620, speed 451.09 words/sec, time elapsed 30726.16 sec  epoch 4, iter 402820, avg. loss 102.69, avg. ppl 359.61 cum. examples 1640, speed 459.82 words/sec, time elapsed 30726.92 sec  epoch 4, iter 402830, avg. loss 85.10, avg. ppl 546.44 cum. examples 1660, speed 412.84 words/sec, time elapsed 30727.58 sec  epoch 4, iter 402840, avg. loss 107.58, avg. ppl 570.77 cum. examples 1680, speed 472.80 words/sec, time elapsed 30728.29 sec  epoch 4, iter 402850, avg. loss 121.91, avg. ppl 1653.72 cum. examples 1700, speed 468.00 words/sec, time elapsed 30729.00 sec  epoch 4, iter 402860, avg. loss 133.16, avg. ppl 1105.80 cum. examples 1720, speed 477.99 words/sec, time elapsed 30729.79 sec  epoch 4, iter 402870, avg. loss 112.20, avg. ppl 976.27 cum. examples 1740, speed 460.45 words/sec, time elapsed 30730.50 sec  epoch 4, iter 402880, avg. loss 125.16, avg. ppl 1226.14 cum. examples 1760, speed 471.85 words/sec, time elapsed 30731.25 sec  epoch 4, iter 402890, avg. loss 113.42, avg. ppl 1074.38 cum. examples 1780, speed 453.91 words/sec, time elapsed 30731.96 sec  epoch 4, iter 402900, avg. loss 126.74, avg. ppl 774.98 cum. examples 1800, speed 448.76 words/sec, time elapsed 30732.81 sec  epoch 4, iter 402910, avg. loss 141.13, avg. ppl 2838.70 cum. examples 1820, speed 493.06 words/sec, time elapsed 30733.53 sec  epoch 4, iter 402920, avg. loss 127.26, avg. ppl 1088.41 cum. examples 1840, speed 482.12 words/sec, time elapsed 30734.29 sec  epoch 4, iter 402930, avg. loss 174.17, avg. ppl 2844.30 cum. examples 1860, speed 485.59 words/sec, time elapsed 30735.19 sec  epoch 4, iter 402940, avg. loss 154.31, avg. ppl 3092.85 cum. examples 1880, speed 467.72 words/sec, time elapsed 30736.01 sec  epoch 4, iter 402950, avg. loss 129.20, avg. ppl 1954.32 cum. examples 1900, speed 436.06 words/sec, time elapsed 30736.79 sec  epoch 4, iter 402960, avg. loss 135.32, avg. ppl 1150.52 cum. examples 1920, speed 495.48 words/sec, time elapsed 30737.57 sec  epoch 4, iter 402970, avg. loss 146.78, avg. ppl 1754.11 cum. examples 1940, speed 486.99 words/sec, time elapsed 30738.38 sec  epoch 4, iter 402980, avg. loss 135.13, avg. ppl 1486.36 cum. examples 1960, speed 480.52 words/sec, time elapsed 30739.15 sec  epoch 4, iter 402990, avg. loss 155.42, avg. ppl 2280.93 cum. examples 1980, speed 511.45 words/sec, time elapsed 30739.93 sec  epoch 4, iter 403000, avg. loss 153.23, avg. ppl 2341.05 cum. examples 2000, speed 456.12 words/sec, time elapsed 30740.80 sec  epoch 4, iter 403010, avg. loss 97.96, avg. ppl 1179.70 cum. examples 2020, speed 395.71 words/sec, time elapsed 30741.50 sec  epoch 4, iter 403020, avg. loss 133.42, avg. ppl 1589.27 cum. examples 2040, speed 482.67 words/sec, time elapsed 30742.25 sec  epoch 4, iter 403030, avg. loss 105.84, avg. ppl 1771.67 cum. examples 2060, speed 416.18 words/sec, time elapsed 30742.93 sec  epoch 4, iter 403040, avg. loss 161.82, avg. ppl 5111.97 cum. examples 2080, speed 502.65 words/sec, time elapsed 30743.68 sec  epoch 4, iter 403050, avg. loss 130.91, avg. ppl 4294.62 cum. examples 2100, speed 477.13 words/sec, time elapsed 30744.34 sec  epoch 4, iter 403060, avg. loss 102.32, avg. ppl 857.09 cum. examples 2120, speed 454.96 words/sec, time elapsed 30745.00 sec  epoch 4, iter 403070, avg. loss 107.65, avg. ppl 3376.46 cum. examples 2140, speed 414.71 words/sec, time elapsed 30745.64 sec  epoch 4, iter 403080, avg. loss 152.96, avg. ppl 1250.33 cum. examples 2160, speed 528.98 words/sec, time elapsed 30746.45 sec  epoch 4, iter 403090, avg. loss 160.31, avg. ppl 2104.91 cum. examples 2180, speed 476.14 words/sec, time elapsed 30747.33 sec  epoch 4, iter 403100, avg. loss 131.59, avg. ppl 809.91 cum. examples 2200, speed 489.42 words/sec, time elapsed 30748.14 sec  epoch 4, iter 403110, avg. loss 139.32, avg. ppl 1178.64 cum. examples 2220, speed 465.17 words/sec, time elapsed 30748.98 sec  epoch 4, iter 403120, avg. loss 146.48, avg. ppl 1435.60 cum. examples 2240, speed 454.34 words/sec, time elapsed 30749.87 sec  epoch 4, iter 403130, avg. loss 199.69, avg. ppl 2517.83 cum. examples 2260, speed 510.51 words/sec, time elapsed 30750.87 sec  epoch 4, iter 403140, avg. loss 175.99, avg. ppl 1397.64 cum. examples 2280, speed 499.49 words/sec, time elapsed 30751.84 sec  epoch 4, iter 403150, avg. loss 150.12, avg. ppl 1229.81 cum. examples 2300, speed 473.62 words/sec, time elapsed 30752.73 sec  epoch 4, iter 403160, avg. loss 151.62, avg. ppl 3876.43 cum. examples 2320, speed 468.71 words/sec, time elapsed 30753.52 sec  epoch 4, iter 403170, avg. loss 130.87, avg. ppl 3566.52 cum. examples 2340, speed 467.15 words/sec, time elapsed 30754.20 sec  epoch 4, iter 403180, avg. loss 146.16, avg. ppl 2023.59 cum. examples 2360, speed 481.20 words/sec, time elapsed 30755.00 sec  epoch 4, iter 403190, avg. loss 136.33, avg. ppl 1866.82 cum. examples 2380, speed 471.35 words/sec, time elapsed 30755.77 sec  epoch 4, iter 403200, avg. loss 136.64, avg. ppl 649.33 cum. examples 2400, speed 496.47 words/sec, time elapsed 30756.62 sec  epoch 4, iter 403210, avg. loss 107.06, avg. ppl 1171.89 cum. examples 2420, speed 417.36 words/sec, time elapsed 30757.34 sec  epoch 4, iter 403220, avg. loss 100.87, avg. ppl 546.96 cum. examples 2440, speed 473.37 words/sec, time elapsed 30758.02 sec  epoch 4, iter 403230, avg. loss 94.65, avg. ppl 561.74 cum. examples 2460, speed 432.08 words/sec, time elapsed 30758.71 sec  epoch 4, iter 403240, avg. loss 145.89, avg. ppl 1809.02 cum. examples 2480, speed 466.43 words/sec, time elapsed 30759.55 sec  epoch 4, iter 403250, avg. loss 123.03, avg. ppl 1770.33 cum. examples 2500, speed 470.00 words/sec, time elapsed 30760.25 sec  epoch 4, iter 403260, avg. loss 152.23, avg. ppl 4007.45 cum. examples 2520, speed 461.05 words/sec, time elapsed 30761.04 sec  epoch 4, iter 403270, avg. loss 166.06, avg. ppl 4993.26 cum. examples 2540, speed 483.27 words/sec, time elapsed 30761.85 sec  epoch 4, iter 403280, avg. loss 165.70, avg. ppl 2435.01 cum. examples 2560, speed 454.06 words/sec, time elapsed 30762.79 sec  epoch 4, iter 403290, avg. loss 140.56, avg. ppl 1632.19 cum. examples 2580, speed 505.99 words/sec, time elapsed 30763.54 sec  epoch 4, iter 403300, avg. loss 134.66, avg. ppl 2819.81 cum. examples 2600, speed 485.67 words/sec, time elapsed 30764.24 sec  epoch 4, iter 403310, avg. loss 107.72, avg. ppl 741.47 cum. examples 2620, speed 455.31 words/sec, time elapsed 30764.95 sec  epoch 4, iter 403320, avg. loss 143.16, avg. ppl 4219.94 cum. examples 2640, speed 466.67 words/sec, time elapsed 30765.69 sec  epoch 4, iter 403330, avg. loss 105.43, avg. ppl 684.21 cum. examples 2660, speed 466.76 words/sec, time elapsed 30766.38 sec  epoch 4, iter 403340, avg. loss 114.82, avg. ppl 988.53 cum. examples 2680, speed 441.06 words/sec, time elapsed 30767.13 sec  epoch 4, iter 403350, avg. loss 147.04, avg. ppl 1258.66 cum. examples 2700, speed 485.85 words/sec, time elapsed 30767.98 sec  epoch 4, iter 403360, avg. loss 111.89, avg. ppl 1396.82 cum. examples 2720, speed 443.33 words/sec, time elapsed 30768.68 sec  epoch 4, iter 403370, avg. loss 115.92, avg. ppl 1608.60 cum. examples 2740, speed 445.39 words/sec, time elapsed 30769.38 sec  epoch 4, iter 403380, avg. loss 137.23, avg. ppl 1508.96 cum. examples 2760, speed 471.70 words/sec, time elapsed 30770.18 sec  epoch 4, iter 403390, avg. loss 112.46, avg. ppl 1945.74 cum. examples 2780, speed 440.00 words/sec, time elapsed 30770.85 sec  epoch 4, iter 403400, avg. loss 115.69, avg. ppl 1548.94 cum. examples 2800, speed 476.55 words/sec, time elapsed 30771.52 sec  epoch 4, iter 403410, avg. loss 116.76, avg. ppl 980.93 cum. examples 2820, speed 469.53 words/sec, time elapsed 30772.24 sec  epoch 4, iter 403420, avg. loss 105.63, avg. ppl 872.52 cum. examples 2840, speed 417.11 words/sec, time elapsed 30772.99 sec  epoch 4, iter 403430, avg. loss 127.82, avg. ppl 1970.14 cum. examples 2860, speed 482.12 words/sec, time elapsed 30773.69 sec  epoch 4, iter 403440, avg. loss 120.29, avg. ppl 1841.66 cum. examples 2880, speed 467.15 words/sec, time elapsed 30774.37 sec  epoch 4, iter 403450, avg. loss 102.40, avg. ppl 965.08 cum. examples 2900, speed 441.48 words/sec, time elapsed 30775.05 sec  epoch 4, iter 403460, avg. loss 102.42, avg. ppl 602.46 cum. examples 2920, speed 424.97 words/sec, time elapsed 30775.80 sec  epoch 4, iter 403470, avg. loss 131.05, avg. ppl 1064.80 cum. examples 2940, speed 517.91 words/sec, time elapsed 30776.52 sec  epoch 4, iter 403480, avg. loss 134.38, avg. ppl 749.64 cum. examples 2960, speed 497.55 words/sec, time elapsed 30777.34 sec  epoch 4, iter 403490, avg. loss 154.30, avg. ppl 2000.56 cum. examples 2980, speed 513.28 words/sec, time elapsed 30778.13 sec  epoch 4, iter 403500, avg. loss 94.23, avg. ppl 534.68 cum. examples 3000, speed 432.90 words/sec, time elapsed 30778.82 sec  epoch 4, iter 403510, avg. loss 132.27, avg. ppl 1875.82 cum. examples 3020, speed 473.05 words/sec, time elapsed 30779.57 sec  epoch 4, iter 403520, avg. loss 91.81, avg. ppl 395.80 cum. examples 3040, speed 436.08 words/sec, time elapsed 30780.27 sec  epoch 4, iter 403530, avg. loss 144.81, avg. ppl 1321.31 cum. examples 3060, speed 450.28 words/sec, time elapsed 30781.17 sec  epoch 4, iter 403540, avg. loss 126.13, avg. ppl 1530.22 cum. examples 3080, speed 464.86 words/sec, time elapsed 30781.91 sec  epoch 4, iter 403550, avg. loss 130.20, avg. ppl 946.37 cum. examples 3100, speed 462.29 words/sec, time elapsed 30782.73 sec  epoch 4, iter 403560, avg. loss 136.18, avg. ppl 1971.75 cum. examples 3120, speed 481.88 words/sec, time elapsed 30783.47 sec  epoch 4, iter 403570, avg. loss 118.44, avg. ppl 2989.07 cum. examples 3140, speed 452.60 words/sec, time elapsed 30784.13 sec  epoch 4, iter 403580, avg. loss 142.87, avg. ppl 1918.66 cum. examples 3160, speed 439.53 words/sec, time elapsed 30784.99 sec  epoch 4, iter 403590, avg. loss 137.37, avg. ppl 5214.36 cum. examples 3180, speed 452.75 words/sec, time elapsed 30785.70 sec  epoch 4, iter 403600, avg. loss 142.00, avg. ppl 1869.25 cum. examples 3200, speed 479.64 words/sec, time elapsed 30786.48 sec  epoch 4, iter 403610, avg. loss 100.09, avg. ppl 826.68 cum. examples 3220, speed 429.39 words/sec, time elapsed 30787.18 sec  epoch 4, iter 403620, avg. loss 104.39, avg. ppl 641.33 cum. examples 3240, speed 424.44 words/sec, time elapsed 30787.94 sec  epoch 4, iter 403630, avg. loss 148.85, avg. ppl 3902.54 cum. examples 3260, speed 458.02 words/sec, time elapsed 30788.72 sec  epoch 4, iter 403640, avg. loss 129.37, avg. ppl 1349.43 cum. examples 3280, speed 497.92 words/sec, time elapsed 30789.44 sec  epoch 4, iter 403650, avg. loss 102.36, avg. ppl 2077.41 cum. examples 3300, speed 411.04 words/sec, time elapsed 30790.10 sec  epoch 4, iter 403660, avg. loss 135.08, avg. ppl 3775.77 cum. examples 3320, speed 445.05 words/sec, time elapsed 30790.83 sec  epoch 4, iter 403670, avg. loss 137.31, avg. ppl 1777.62 cum. examples 3340, speed 454.77 words/sec, time elapsed 30791.64 sec  epoch 4, iter 403680, avg. loss 151.69, avg. ppl 1605.98 cum. examples 3360, speed 491.63 words/sec, time elapsed 30792.48 sec  epoch 4, iter 403690, avg. loss 110.31, avg. ppl 887.34 cum. examples 3380, speed 427.07 words/sec, time elapsed 30793.24 sec  epoch 4, iter 403700, avg. loss 115.11, avg. ppl 1896.75 cum. examples 3400, speed 420.11 words/sec, time elapsed 30793.96 sec  epoch 4, iter 403710, avg. loss 143.12, avg. ppl 4764.43 cum. examples 3420, speed 443.57 words/sec, time elapsed 30794.73 sec  epoch 4, iter 403720, avg. loss 125.87, avg. ppl 4409.82 cum. examples 3440, speed 458.02 words/sec, time elapsed 30795.38 sec  epoch 4, iter 403730, avg. loss 108.24, avg. ppl 2157.49 cum. examples 3460, speed 392.76 words/sec, time elapsed 30796.10 sec  epoch 4, iter 403740, avg. loss 104.93, avg. ppl 1199.40 cum. examples 3480, speed 445.78 words/sec, time elapsed 30796.76 sec  epoch 4, iter 403750, avg. loss 124.55, avg. ppl 767.19 cum. examples 3500, speed 492.13 words/sec, time elapsed 30797.52 sec  epoch 4, iter 403760, avg. loss 99.70, avg. ppl 2020.06 cum. examples 3520, speed 398.18 words/sec, time elapsed 30798.18 sec  epoch 4, iter 403770, avg. loss 123.41, avg. ppl 1584.47 cum. examples 3540, speed 457.65 words/sec, time elapsed 30798.92 sec  epoch 4, iter 403780, avg. loss 81.48, avg. ppl 596.34 cum. examples 3560, speed 435.90 words/sec, time elapsed 30799.50 sec  epoch 4, iter 403790, avg. loss 122.44, avg. ppl 1788.14 cum. examples 3580, speed 418.69 words/sec, time elapsed 30800.28 sec  epoch 4, iter 403800, avg. loss 142.97, avg. ppl 2754.17 cum. examples 3600, speed 485.87 words/sec, time elapsed 30801.02 sec  epoch 4, iter 403810, avg. loss 142.53, avg. ppl 3213.88 cum. examples 3620, speed 480.27 words/sec, time elapsed 30801.76 sec  epoch 4, iter 403820, avg. loss 155.03, avg. ppl 7797.01 cum. examples 3640, speed 443.02 words/sec, time elapsed 30802.54 sec  epoch 4, iter 403830, avg. loss 127.22, avg. ppl 1630.15 cum. examples 3660, speed 435.44 words/sec, time elapsed 30803.33 sec  epoch 4, iter 403840, avg. loss 130.85, avg. ppl 1157.26 cum. examples 3680, speed 472.01 words/sec, time elapsed 30804.12 sec  epoch 4, iter 403850, avg. loss 150.57, avg. ppl 2086.63 cum. examples 3700, speed 502.55 words/sec, time elapsed 30804.90 sec  epoch 4, iter 403860, avg. loss 139.41, avg. ppl 728.70 cum. examples 3720, speed 485.65 words/sec, time elapsed 30805.77 sec  epoch 4, iter 403870, avg. loss 107.16, avg. ppl 1236.65 cum. examples 3740, speed 458.84 words/sec, time elapsed 30806.43 sec  epoch 4, iter 403880, avg. loss 98.88, avg. ppl 2560.38 cum. examples 3760, speed 417.22 words/sec, time elapsed 30807.03 sec  epoch 4, iter 403890, avg. loss 145.55, avg. ppl 6098.62 cum. examples 3780, speed 484.76 words/sec, time elapsed 30807.72 sec  epoch 4, iter 403900, avg. loss 147.45, avg. ppl 1506.65 cum. examples 3800, speed 508.20 words/sec, time elapsed 30808.52 sec  epoch 4, iter 403910, avg. loss 143.47, avg. ppl 2831.81 cum. examples 3820, speed 495.88 words/sec, time elapsed 30809.24 sec  epoch 4, iter 403920, avg. loss 107.06, avg. ppl 1694.10 cum. examples 3840, speed 431.14 words/sec, time elapsed 30809.91 sec  epoch 4, iter 403930, avg. loss 154.27, avg. ppl 3152.60 cum. examples 3860, speed 479.35 words/sec, time elapsed 30810.71 sec  epoch 4, iter 403940, avg. loss 148.51, avg. ppl 1400.25 cum. examples 3880, speed 478.41 words/sec, time elapsed 30811.57 sec  epoch 4, iter 403950, avg. loss 133.75, avg. ppl 1434.80 cum. examples 3900, speed 453.20 words/sec, time elapsed 30812.38 sec  epoch 4, iter 403960, avg. loss 126.82, avg. ppl 2769.50 cum. examples 3920, speed 466.47 words/sec, time elapsed 30813.07 sec  epoch 4, iter 403970, avg. loss 152.45, avg. ppl 2867.09 cum. examples 3940, speed 484.20 words/sec, time elapsed 30813.86 sec  epoch 4, iter 403980, avg. loss 168.07, avg. ppl 4557.34 cum. examples 3960, speed 512.19 words/sec, time elapsed 30814.64 sec  epoch 4, iter 403990, avg. loss 115.86, avg. ppl 736.36 cum. examples 3980, speed 459.42 words/sec, time elapsed 30815.40 sec  epoch 4, iter 404000, avg. loss 108.10, avg. ppl 2023.04 cum. examples 4000, speed 443.75 words/sec, time elapsed 30816.04 sec (8 hr, 33 min, 36 sec)  epoch 4, iter 404000, cum. loss 130.26, cum. ppl 1565.87 cum. examples 4000  begin validation ...  validation: iter 404000, dev. ppl 1817.070715  hit patience 5  hit #5 trial  early stop!  (local\_nmt) C:\Users\ADMIN\Documents\Stanford\NLP\_DeepLearning\Assignment5\XCS224N-A5-master>python run.py decode model.bin ./en\_es\_data/test.es outputs/test\_outputs.txt --cuda  load test source sentences from [./en\_es\_data/test.es]  load model from model.bin  Decoding: 100%|████████████████████████████| 8064/8064 [20:01<00:00, 6.71it/s] |

* Zip all .py files and include all files under ./en\_es\_data ./sanity\_check\_en\_es\_data ./outputs folders to produce assignment5.zip

|  |
| --- |
| **STUDENT**   * Shuo-Fu Chen  AUTOGRADER SCORE **53.0 / 53.0** PASSED TESTS  1. [1a -- test output of words2charindices (public) (1.0/1.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1a%20--%20test%20output%20of%20words2charindices%20(public)) 2. [1b -- Test output of pad\_sents\_char (public) (4.0/4.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1b%20--%20Test%20output%20of%20pad_sents_char%20(public)) 3. [1c -- Test shape of output of to\_input\_tensor\_char (hidden) (3.0/3.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1c%20--%20Test%20shape%20of%20output%20of%20to_input_tensor_char%20(hidden)) 4. [1f -- Test shape of Conv1D weights in CNN layer (hidden) (2.0/2.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1f%20--%20Test%20shape%20of%20Conv1D%20weights%20in%20CNN%20layer%20(hidden)) 5. [1f -- Test shape of weights in Character Embeddings in ModelEmbeddings (hidden) (2.0/2.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1f%20--%20Test%20shape%20of%20weights%20in%20Character%20Embeddings%20in%20ModelEmbeddings%20(hidden)) 6. [1f -- Test output of ModelEmbeddings.forward (hidden) (5.0/5.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1f%20--%20Test%20output%20of%20ModelEmbeddings.forward%20(hidden)) 7. [1f -- Test shape of weights in Linear Layers in Highway (hidden) (2.0/2.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1f%20--%20Test%20shape%20of%20weights%20in%20Linear%20Layers%20in%20Highway%20(hidden)) 8. [1f -- Test shape of output of ModelEmbeddings.forward (public) (1.0/1.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1f%20--%20Test%20shape%20of%20output%20of%20ModelEmbeddings.forward%20(public)) 9. [1g -- Test output of NMT.forward (hidden) (2.0/2.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1g%20--%20Test%20output%20of%20NMT.forward%20(hidden)) 10. [1h -- BLEU score on tiny test set is over 99 (public) (5.0/5.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#1h%20--%20BLEU%20score%20on%20tiny%20test%20set%20is%20over%2099%20(public)) 11. [2a -- Test correct initialization of self.charDecoder (hidden) (0.4/0.4)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2a%20--%20Test%20correct%20initialization%20of%20self.charDecoder%20(hidden)) 12. [2a -- Test correct initialization of self.char\_output\_projection (hidden) (0.4/0.4)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2a%20--%20Test%20correct%20initialization%20of%20self.char_output_projection%20(hidden)) 13. [2a -- Test correct initialization of self.decoderCharEmb (hidden) (0.4/0.4)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2a%20--%20Test%20correct%20initialization%20of%20self.decoderCharEmb%20(hidden)) 14. [2a -- Test shapes of initialized layers (public) (0.4/0.4)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2a%20--%20Test%20shapes%20of%20initialized%20layers%20(public)) 15. [2a -- Test correct initialization of self.target\_vocab (hidden) (0.4/0.4)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2a%20--%20Test%20correct%20initialization%20of%20self.target_vocab%20(hidden)) 16. [2b -- Test decoder cell state returned by forward (hidden) (0.75/0.75)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2b%20--%20Test%20decoder%20cell%20state%20returned%20by%20forward%20(hidden)) 17. [2b -- Test decoder hidden state returned by forward (hidden) (0.75/0.75)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2b%20--%20Test%20decoder%20hidden%20state%20returned%20by%20forward%20(hidden)) 18. [2b -- Test logits returned by forward (hidden) (1.25/1.25)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2b%20--%20Test%20logits%20returned%20by%20forward%20(hidden)) 19. [2b --- Test shapes of outputs returned by forward (public) (0.25/0.25)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2b%20---%20Test%20shapes%20of%20outputs%20returned%20by%20forward%20(public)) 20. [2c -- Test output of train\_forward (hidden) (4.0/4.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2c%20--%20Test%20output%20of%20train_forward%20(hidden)) 21. [2c -- Test shape of output of train\_forward (public) (1.0/1.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2c%20--%20Test%20shape%20of%20output%20of%20train_forward%20(public)) 22. [2d -- Test output of decode\_greedy for exact match (hidden) (3.0/3.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2d%20--%20Test%20output%20of%20decode_greedy%20for%20exact%20match%20(hidden)) 23. [2d -- Test output of decode\_greedy for partial match (hidden) (3.5/3.5)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2d%20--%20Test%20output%20of%20decode_greedy%20for%20partial%20match%20(hidden)) 24. [2d -- Test shape of output of decode\_greedy (public) (0.5/0.5)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2d%20--%20Test%20shape%20of%20output%20of%20decode_greedy%20(public)) 25. [2e -- BLEU score on tiny test set is over 99 (public) (3.0/3.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2e%20--%20BLEU%20score%20on%20tiny%20test%20set%20is%20over%2099%20(public)) 26. [2f -- BLEU score above 16 (public) (4.0/4.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2f%20--%20BLEU%20score%20above%2016%20(public)) 27. [2f -- BLEU score above 10 (public) (2.0/2.0)](https://www.gradescope.com/courses/71858/assignments/303160/submissions/29648087#2f%20--%20BLEU%20score%20above%2010%20(public)) |
| 1a -- test output of words2charindices (public) (1.0/1.0)  Test Input: [['<s>', "Let's", 'start', 'by', 'thinking', 'about', 'the', 'member', 'countries', 'of', 'the', 'OECD,', 'or', 'the', 'Organization', 'of', 'Economic', 'Cooperation', 'and', 'Development.', '</s>'], ['<s>', 'In', 'the', 'case', 'of', 'gun', 'control,', 'we', 'really', 'underestimated', 'our', 'opponents.', '</s>'], ['<s>', 'Let', 'me', 'share', 'with', 'those', 'of', 'you', 'here', 'in', 'the', 'first', 'row.', '</s>'], ['<s>', 'It', 'suggests', 'that', 'we', 'care', 'about', 'the', 'fight,', 'about', 'the', 'challenge.', '</s>'], ['<s>', 'A', 'lot', 'of', 'numbers', 'there.', 'A', 'lot', 'of', 'numbers.', '</s>']]  Student Output: [[[1, 90, 48, 91, 2], [1, 15, 34, 49, 72, 48, 2], [1, 48, 49, 30, 47, 49, 2], [1, 31, 54, 2], [1, 49, 37, 38, 43, 40, 38, 43, 36, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 42, 34, 42, 31, 34, 47, 2], [1, 32, 44, 50, 43, 49, 47, 38, 34, 48, 2], [1, 44, 35, 2], [1, 49, 37, 34, 2], [1, 18, 8, 6, 7, 66, 2], [1, 44, 47, 2], [1, 49, 37, 34, 2], [1, 18, 47, 36, 30, 43, 38, 55, 30, 49, 38, 44, 43, 2], [1, 44, 35, 2], [1, 8, 32, 44, 43, 44, 42, 38, 32, 2], [1, 6, 44, 44, 45, 34, 47, 30, 49, 38, 44, 43, 2], [1, 30, 43, 33, 2], [1, 7, 34, 51, 34, 41, 44, 45, 42, 34, 43, 49, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 12, 43, 2], [1, 49, 37, 34, 2], [1, 32, 30, 48, 34, 2], [1, 44, 35, 2], [1, 36, 50, 43, 2], [1, 32, 44, 43, 49, 47, 44, 41, 66, 2], [1, 52, 34, 2], [1, 47, 34, 30, 41, 41, 54, 2], [1, 50, 43, 33, 34, 47, 34, 48, 49, 38, 42, 30, 49, 34, 33, 2], [1, 44, 50, 47, 2], [1, 44, 45, 45, 44, 43, 34, 43, 49, 48, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 15, 34, 49, 2], [1, 42, 34, 2], [1, 48, 37, 30, 47, 34, 2], [1, 52, 38, 49, 37, 2], [1, 49, 37, 44, 48, 34, 2], [1, 44, 35, 2], [1, 54, 44, 50, 2], [1, 37, 34, 47, 34, 2], [1, 38, 43, 2], [1, 49, 37, 34, 2], [1, 35, 38, 47, 48, 49, 2], [1, 47, 44, 52, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 12, 49, 2], [1, 48, 50, 36, 36, 34, 48, 49, 48, 2], [1, 49, 37, 30, 49, 2], [1, 52, 34, 2], [1, 32, 30, 47, 34, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 35, 38, 36, 37, 49, 66, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 32, 37, 30, 41, 41, 34, 43, 36, 34, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 4, 2], [1, 41, 44, 49, 2], [1, 44, 35, 2], [1, 43, 50, 42, 31, 34, 47, 48, 2], [1, 49, 37, 34, 47, 34, 68, 2], [1, 4, 2], [1, 41, 44, 49, 2], [1, 44, 35, 2], [1, 43, 50, 42, 31, 34, 47, 48, 68, 2], [1, 90, 74, 48, 91, 2]]]  Solution Output: [[[1, 90, 48, 91, 2], [1, 15, 34, 49, 72, 48, 2], [1, 48, 49, 30, 47, 49, 2], [1, 31, 54, 2], [1, 49, 37, 38, 43, 40, 38, 43, 36, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 42, 34, 42, 31, 34, 47, 2], [1, 32, 44, 50, 43, 49, 47, 38, 34, 48, 2], [1, 44, 35, 2], [1, 49, 37, 34, 2], [1, 18, 8, 6, 7, 66, 2], [1, 44, 47, 2], [1, 49, 37, 34, 2], [1, 18, 47, 36, 30, 43, 38, 55, 30, 49, 38, 44, 43, 2], [1, 44, 35, 2], [1, 8, 32, 44, 43, 44, 42, 38, 32, 2], [1, 6, 44, 44, 45, 34, 47, 30, 49, 38, 44, 43, 2], [1, 30, 43, 33, 2], [1, 7, 34, 51, 34, 41, 44, 45, 42, 34, 43, 49, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 12, 43, 2], [1, 49, 37, 34, 2], [1, 32, 30, 48, 34, 2], [1, 44, 35, 2], [1, 36, 50, 43, 2], [1, 32, 44, 43, 49, 47, 44, 41, 66, 2], [1, 52, 34, 2], [1, 47, 34, 30, 41, 41, 54, 2], [1, 50, 43, 33, 34, 47, 34, 48, 49, 38, 42, 30, 49, 34, 33, 2], [1, 44, 50, 47, 2], [1, 44, 45, 45, 44, 43, 34, 43, 49, 48, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 15, 34, 49, 2], [1, 42, 34, 2], [1, 48, 37, 30, 47, 34, 2], [1, 52, 38, 49, 37, 2], [1, 49, 37, 44, 48, 34, 2], [1, 44, 35, 2], [1, 54, 44, 50, 2], [1, 37, 34, 47, 34, 2], [1, 38, 43, 2], [1, 49, 37, 34, 2], [1, 35, 38, 47, 48, 49, 2], [1, 47, 44, 52, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 12, 49, 2], [1, 48, 50, 36, 36, 34, 48, 49, 48, 2], [1, 49, 37, 30, 49, 2], [1, 52, 34, 2], [1, 32, 30, 47, 34, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 35, 38, 36, 37, 49, 66, 2], [1, 30, 31, 44, 50, 49, 2], [1, 49, 37, 34, 2], [1, 32, 37, 30, 41, 41, 34, 43, 36, 34, 68, 2], [1, 90, 74, 48, 91, 2]], [[1, 90, 48, 91, 2], [1, 4, 2], [1, 41, 44, 49, 2], [1, 44, 35, 2], [1, 43, 50, 42, 31, 34, 47, 48, 2], [1, 49, 37, 34, 47, 34, 68, 2], [1, 4, 2], [1, 41, 44, 49, 2], [1, 44, 35, 2], [1, 43, 50, 42, 31, 34, 47, 48, 68, 2], [1, 90, 74, 48, 91, 2]]]  passed!  1b -- Test output of pad\_sents\_char (public) (4.0/4.0)  Input: [['Human:', 'What', 'do', 'we', 'want?'], ['Computer:', 'Natural', 'language', 'processing!'], ['Human:', 'When', 'do', 'we', 'want', 'it?'], ['Computer:', 'When', 'do', 'we', 'want', 'what?']]  Student Output: [[[1, 11, 50, 42, 30, 43, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 30, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 70, 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]], [[1, 6, 44, 42, 45, 50, 49, 34, 47, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 17, 30, 49, 50, 47, 30, 41, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 41, 30, 43, 36, 50, 30, 36, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 45, 47, 44, 32, 34, 48, 48, 38, 43, 36, 69, 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, 11, 50, 42, 30, 43, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 34, 43, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 38, 49, 70, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], [[1, 6, 44, 42, 45, 50, 49, 34, 47, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 34, 43, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 37, 30, 49, 70, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]]  Solution Output: [[[1, 11, 50, 42, 30, 43, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 30, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 70, 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]], [[1, 6, 44, 42, 45, 50, 49, 34, 47, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 17, 30, 49, 50, 47, 30, 41, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 41, 30, 43, 36, 50, 30, 36, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 45, 47, 44, 32, 34, 48, 48, 38, 43, 36, 69, 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, 11, 50, 42, 30, 43, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 34, 43, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 38, 49, 70, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], [[1, 6, 44, 42, 45, 50, 49, 34, 47, 71, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 26, 37, 34, 43, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 33, 44, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 34, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 30, 43, 49, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [1, 52, 37, 30, 49, 70, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]]  passed!  1c -- Test shape of output of to\_input\_tensor\_char (hidden) (3.0/3.0)  Input: [['<s>', "Let's", 'start', 'by', 'thinking', 'about', 'the', 'member', 'countries', 'of', 'the', 'OECD,', 'or', 'the', 'Organization', 'of', 'Economic', 'Cooperation', 'and', 'Development.', '</s>'], ['<s>', 'In', 'the', 'case', 'of', 'gun', 'control,', 'we', 'really', 'underestimated', 'our', 'opponents.', '</s>'], ['<s>', 'Let', 'me', 'share', 'with', 'those', 'of', 'you', 'here', 'in', 'the', 'first', 'row.', '</s>'], ['<s>', 'It', 'suggests', 'that', 'we', 'care', 'about', 'the', 'fight,', 'about', 'the', 'challenge.', '</s>'], ['<s>', 'A', 'lot', 'of', 'numbers', 'there.', 'A', 'lot', 'of', 'numbers.', '</s>']]  Student Output Shape: torch.Size([21, 5, 21])  Solution Output Shape: torch.Size([21, 5, 21])  passed  1f -- Test shape of Conv1D weights in CNN layer (hidden) (2.0/2.0)  Setting batch size to: 5  Initializing Student & Solution ModelEmbeddings Models with following params:  Word Embedding Size: 7  Word-level source vocab has size: 77  ----------------------------------------  `in\_channels` should be: 50  `out\_channels` should Be: 20  `kernel\_size` should be: 5  `bias` size should be: 20  Conv1D `in\_channels`: 50  Conv1D `out\_channels`: 7  Conv1D `kernel\_size`: 5  Conv1D `bias` size: 7  pass!  1f -- Test shape of weights in Character Embeddings in ModelEmbeddings (hidden) (2.0/2.0)  Setting batch size to: 5  Initializing Student & Solution ModelEmbeddings Models with following params:  Word Embedding Size: 7  Word-level source vocab has size: 77  ----------------------------------------  Student ModelEmbeddings Vocab Size: 96  Student ModelEmbeddings Character Embedding Size Size: 50  pass!  1f -- Test output of ModelEmbeddings.forward (hidden) (5.0/5.0)  Setting batch size to: 5  Initializing Student & Solution ModelEmbeddings Models with following params:  Word Embedding Size: 7  Word-level source vocab has size: 77  ----------------------------------------  Using same weights in student code and solution code.  --------------------------------------------------------------------------------  True  Running solution's ModelEmbeddings.forward()...  Running student's ModelEmbeddings.forward()...  Solution Output:  tensor([[[0.1062, 0.2372, 0.2838, 0.0263, 0.3298, 0.4440, 0.1762],  [0.5955, 0.3084, 0.2114, 0.3237, 0.5224, 0.6647, 0.4649],  [0.6174, 0.3262, 0.2837, 0.2292, 0.5176, 0.7171, 0.5129],  [0.4027, 0.7489, 0.2308, 0.7100, 0.7768, 0.7685, 0.4380],  [0.5263, 0.2075, 1.0493, 0.5782, 0.6176, 0.5016, 0.3212]],  [[0.1936, 0.2281, 0.6400, 0.0817, 0.3496, 0.4466, 0.1887],  [0.3429, 0.1560, 0.3965, 0.0975, 0.1272, 0.3678, 0.1811],  [0.3184, 0.1735, 0.3160, 0.3081, 0.1380, 0.4726, 0.4243],  [0.6230, 1.1023, 0.2879, 0.9216, 0.4478, 0.8996, 0.6253],  [0.2050, 0.1463, 0.6431, 0.0393, 0.1152, 0.3004, 0.0809]],  [[0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.2186, 0.3359, 0.6681, 0.2197, 0.6107, 0.6808, 0.2897],  [0.2319, 0.4264, 0.8624, 0.2201, 0.4138, 0.5076, 0.1614],  [0.2923, 0.6150, 0.8595, 0.2677, 0.4889, 0.5100, 0.0615],  [0.3864, 0.3417, 0.7594, 0.5917, 0.6607, 0.6041, 0.4321]],  [[0.3601, 0.1969, 0.4105, 0.0615, 0.1972, 0.5128, 0.3509],  [0.1651, 0.4629, 0.4418, 0.3020, 0.4171, 0.7249, 0.5767],  [0.5772, 0.1242, 0.7899, 0.1737, 0.2813, 0.4445, 0.2684],  [0.3253, 0.4423, 0.5529, 0.2226, 0.3909, 0.6019, 0.2625],  [0.3108, 0.1792, 0.6540, 0.3307, 0.1432, 0.4119, 0.3157]],  [[0.6051, 0.3148, 0.8775, 0.3893, 0.5405, 0.6415, 0.4485],  [0.1716, 0.7535, 0.9412, 0.1155, 0.6035, 0.6877, 0.3394],  [0.2645, 0.4515, 0.6651, 0.0279, 0.2924, 0.5959, 0.2108],  [0.3108, 0.1792, 0.6540, 0.3307, 0.1432, 0.4119, 0.3157],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191]],  [[0.8649, 0.3893, 0.5351, 0.6818, 0.4740, 0.7529, 0.6202],  [0.1017, 0.1305, 0.6679, 0.0928, 0.2006, 0.2866, 0.0514],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.2285, 0.1623, 0.2402, 0.0853, 0.4769, 0.5037, 0.3843]],  [[0.3809, 0.2816, 0.6349, 0.3953, 0.5584, 0.5673, 0.3722],  [0.6488, 0.3028, 0.6386, 0.3964, 0.9907, 0.9760, 0.7508],  [0.2285, 0.1623, 0.2402, 0.0853, 0.4769, 0.5037, 0.3843],  [0.3458, 0.0851, 0.7007, 0.1098, 0.3255, 0.4005, 0.2438],  [0.4162, 0.2981, 0.3618, 0.5677, 0.3013, 0.7459, 0.5833]],  [[0.8724, 0.3144, 0.7550, 0.5377, 0.6042, 0.8450, 0.7343],  [0.3429, 0.1560, 0.3965, 0.0975, 0.1272, 0.3678, 0.1811],  [0.9006, 0.3520, 0.4181, 0.5980, 0.4166, 0.6359, 0.4678],  [0.6880, 0.5739, 0.4533, 0.9088, 0.9325, 0.8685, 0.6545],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.3458, 0.0851, 0.7007, 0.1098, 0.3255, 0.4005, 0.2438],  [0.4839, 0.3504, 0.2373, 0.1176, 0.2937, 0.6476, 0.3955],  [0.2956, 0.0709, 0.2016, 0.2782, 0.3382, 0.4717, 0.4305],  [0.4144, 0.4315, 0.5478, 0.4445, 0.5833, 0.7949, 0.6646],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.6236, 0.4200, 0.8852, 0.4626, 0.7301, 0.7499, 0.5180],  [0.3563, 0.2031, 0.1599, 0.2383, 0.2159, 0.4903, 0.3192],  [0.6577, 0.0884, 0.8979, 0.3809, 0.5014, 0.4473, 0.2121],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.5168, 0.8877, 0.9355, 0.5904, 0.8218, 0.8157, 0.4226],  [0.3428, 0.3995, 0.5616, 0.3277, 0.7143, 0.8096, 0.6168],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.1936, 0.2281, 0.6400, 0.0817, 0.3496, 0.4466, 0.1887],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1012, 0.2722, 0.6341, 0.0639, 0.1852, 0.4120, 0.2472],  [0.6371, 0.3949, 0.6346, 0.5102, 0.6423, 0.9360, 0.8121],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.7108, 0.5199, 0.5656, 0.2599, 0.5355, 0.7236, 0.2883],  [0.6244, 0.2369, 1.0610, 0.4537, 0.3954, 0.5739, 0.5118],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2058, 0.2211, 0.1927, 0.3580, 0.4137, 0.5007, 0.3177],  [0.4460, 0.3862, 0.8091, 0.4319, 0.6227, 0.8523, 0.6046],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1662, 0.2608, 0.3548, 0.0273, 0.3784, 0.5741, 0.1609],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.5053, 0.4820, 0.6654, 0.1964, 0.4014, 0.5615, 0.2082],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2923, 0.6150, 0.8595, 0.2677, 0.4889, 0.5100, 0.0615],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1017, 0.1305, 0.6679, 0.0928, 0.2006, 0.2866, 0.0514],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2750, 0.3294, 0.3957, 0.6306, 0.4110, 0.5000, 0.3632],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.0698, 0.1815, 0.2862, 0.4127, 0.2893, 0.3905, 0.2361],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1841, 0.1591, 0.7595, 0.4279, 0.5422, 0.5682, 0.5174],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]]])  Student Output:  tensor([[[0.1062, 0.2372, 0.2838, 0.0263, 0.3298, 0.4440, 0.1762],  [0.5955, 0.3084, 0.2114, 0.3237, 0.5224, 0.6647, 0.4649],  [0.6174, 0.3262, 0.2837, 0.2292, 0.5176, 0.7171, 0.5129],  [0.4027, 0.7489, 0.2308, 0.7100, 0.7768, 0.7685, 0.4380],  [0.5263, 0.2075, 1.0493, 0.5782, 0.6176, 0.5016, 0.3212]],  [[0.1936, 0.2281, 0.6400, 0.0817, 0.3496, 0.4466, 0.1887],  [0.3429, 0.1560, 0.3965, 0.0975, 0.1272, 0.3678, 0.1811],  [0.3184, 0.1735, 0.3160, 0.3081, 0.1380, 0.4726, 0.4243],  [0.6230, 1.1023, 0.2879, 0.9216, 0.4478, 0.8996, 0.6253],  [0.2050, 0.1463, 0.6431, 0.0393, 0.1152, 0.3004, 0.0809]],  [[0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.2186, 0.3359, 0.6681, 0.2197, 0.6107, 0.6808, 0.2897],  [0.2319, 0.4264, 0.8624, 0.2201, 0.4138, 0.5076, 0.1614],  [0.2923, 0.6150, 0.8595, 0.2677, 0.4889, 0.5100, 0.0615],  [0.3864, 0.3417, 0.7594, 0.5917, 0.6607, 0.6041, 0.4321]],  [[0.3601, 0.1969, 0.4105, 0.0615, 0.1972, 0.5128, 0.3509],  [0.1651, 0.4629, 0.4418, 0.3020, 0.4171, 0.7249, 0.5767],  [0.5772, 0.1242, 0.7899, 0.1737, 0.2813, 0.4445, 0.2684],  [0.3253, 0.4423, 0.5529, 0.2226, 0.3909, 0.6019, 0.2625],  [0.3108, 0.1792, 0.6540, 0.3307, 0.1432, 0.4119, 0.3157]],  [[0.6051, 0.3148, 0.8775, 0.3893, 0.5405, 0.6415, 0.4485],  [0.1716, 0.7535, 0.9412, 0.1155, 0.6035, 0.6877, 0.3394],  [0.2645, 0.4515, 0.6651, 0.0279, 0.2924, 0.5959, 0.2108],  [0.3108, 0.1792, 0.6540, 0.3307, 0.1432, 0.4119, 0.3157],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191]],  [[0.8649, 0.3893, 0.5351, 0.6818, 0.4740, 0.7529, 0.6202],  [0.1017, 0.1305, 0.6679, 0.0928, 0.2006, 0.2866, 0.0514],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.2285, 0.1623, 0.2402, 0.0853, 0.4769, 0.5037, 0.3843]],  [[0.3809, 0.2816, 0.6349, 0.3953, 0.5584, 0.5673, 0.3722],  [0.6488, 0.3028, 0.6386, 0.3964, 0.9907, 0.9760, 0.7508],  [0.2285, 0.1623, 0.2402, 0.0853, 0.4769, 0.5037, 0.3843],  [0.3458, 0.0851, 0.7007, 0.1098, 0.3255, 0.4005, 0.2438],  [0.4162, 0.2981, 0.3618, 0.5677, 0.3013, 0.7459, 0.5833]],  [[0.8724, 0.3144, 0.7550, 0.5377, 0.6042, 0.8450, 0.7343],  [0.3429, 0.1560, 0.3965, 0.0975, 0.1272, 0.3678, 0.1811],  [0.9006, 0.3520, 0.4181, 0.5980, 0.4166, 0.6359, 0.4678],  [0.6880, 0.5739, 0.4533, 0.9088, 0.9325, 0.8685, 0.6545],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.3458, 0.0851, 0.7007, 0.1098, 0.3255, 0.4005, 0.2438],  [0.4839, 0.3504, 0.2373, 0.1176, 0.2937, 0.6476, 0.3955],  [0.2956, 0.0709, 0.2016, 0.2782, 0.3382, 0.4717, 0.4305],  [0.4144, 0.4315, 0.5478, 0.4445, 0.5833, 0.7949, 0.6646],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.6236, 0.4200, 0.8852, 0.4626, 0.7301, 0.7499, 0.5180],  [0.3563, 0.2031, 0.1599, 0.2383, 0.2159, 0.4903, 0.3192],  [0.6577, 0.0884, 0.8979, 0.3809, 0.5014, 0.4473, 0.2121],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.5168, 0.8877, 0.9355, 0.5904, 0.8218, 0.8157, 0.4226],  [0.3428, 0.3995, 0.5616, 0.3277, 0.7143, 0.8096, 0.6168],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1999, 0.2567, 0.1976, 0.2561, 0.5589, 0.5576, 0.3191],  [0.1936, 0.2281, 0.6400, 0.0817, 0.3496, 0.4466, 0.1887],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1012, 0.2722, 0.6341, 0.0639, 0.1852, 0.4120, 0.2472],  [0.6371, 0.3949, 0.6346, 0.5102, 0.6423, 0.9360, 0.8121],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.7108, 0.5199, 0.5656, 0.2599, 0.5355, 0.7236, 0.2883],  [0.6244, 0.2369, 1.0610, 0.4537, 0.3954, 0.5739, 0.5118],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2058, 0.2211, 0.1927, 0.3580, 0.4137, 0.5007, 0.3177],  [0.4460, 0.3862, 0.8091, 0.4319, 0.6227, 0.8523, 0.6046],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1662, 0.2608, 0.3548, 0.0273, 0.3784, 0.5741, 0.1609],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.5053, 0.4820, 0.6654, 0.1964, 0.4014, 0.5615, 0.2082],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2923, 0.6150, 0.8595, 0.2677, 0.4889, 0.5100, 0.0615],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1017, 0.1305, 0.6679, 0.0928, 0.2006, 0.2866, 0.0514],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.2750, 0.3294, 0.3957, 0.6306, 0.4110, 0.5000, 0.3632],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.0698, 0.1815, 0.2862, 0.4127, 0.2893, 0.3905, 0.2361],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]],  [[0.1841, 0.1591, 0.7595, 0.4279, 0.5422, 0.5682, 0.5174],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896],  [0.0616, 0.2059, 0.1195, 0.0599, 0.0000, 0.1745, 0.1896]]])  Passed!  /usr/local/lib/python3.6/dist-packages/torch/nn/functional.py:1351: UserWarning: nn.functional.sigmoid is deprecated. Use torch.sigmoid instead.  warnings.warn("nn.functional.sigmoid is deprecated. Use torch.sigmoid instead.")  1f -- Test shape of weights in Linear Layers in Highway (hidden) (2.0/2.0)  Setting batch size to: 5  Initializing Student & Solution ModelEmbeddings Models with following params:  Word Embedding Size: 7  Word-level source vocab has size: 77  ----------------------------------------  Highway's Linear Layers' `in\_features` and `out\_features` should be: 20  Highway's Linear Layers' `bias` size should be: 20  Linear Layer's `in\_features: 7  Linear Layer's `out\_features: 7  Linear Layer's `bias` size: 7  Linear Layer's `in\_features: 7  Linear Layer's `out\_features: 7  Linear Layer's `bias` size: 7  pass!  1f -- Test shape of output of ModelEmbeddings.forward (public) (1.0/1.0)  Setting batch size to: 5  Initializing Student & Solution NMT Models with following hyperparams:  Word Embedding Size: 3  Hidden Size: 3  Dropout Rate: 0.3  Word-level source vocab has size: 77  Word-level target vocab has size: 85  ----------------------------------------  Sentence Length: 10  Max Word Length: 21  Input of Dimensions: (10, 5, 21)  ModelEmbeddings.forward() expected output size:[10, 5, 3]  ModelEmbeddings.forward() output size:[10, 5, 3]  1g -- Test output of NMT.forward (hidden) (2.0/2.0)  Hyperparameters:  Batch Size: 50  Word Embedding Size: 20  Hidden Size: 128  Source Vocab Length: 77  Target Vocab Length: 85  ----------------------------------------  Initialize solutions's NMT instance with output embedding size 20 and hidden size 128.  Initialize student's NMT instance with output embedding size 20 and hidden size 128.  Substitute student's ModelEmbeddings instance with solution's ModelEmbeddings  Copying other weights from solution to student  Running student's NMT.forward() function...  Student output from NMT.forward():  -3497.7893  Solution output from NMT.forward():  -3497.7893  Passed!  1h -- BLEU score on tiny test set is over 99 (public) (5.0/5.0)  BLEU 99.29792465574434  2a -- Test correct initialization of self.charDecoder (hidden) (0.4/0.4)  Initializing CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Using batch\_size=5  Initializing student's CharDecoder...  Testing student's initialization of charDecoder...  Checking charDecoder is a nn.LSTM object...  Checking charDecoder.input\_size...  Checking charDecoder.hidden\_size...  Checking charDecoder.bias...  Note: student's charDecoder set batch\_first=False. Standard solution assumes batch\_first=False (default setting).  All tests passed!  2a -- Test correct initialization of self.char\_output\_projection (hidden) (0.4/0.4)  Initializing CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Using batch\_size=5  Initializing student's CharDecoder...  Testing student's initialization of char\_output\_projection...  Checking char\_output\_projection is a nn.Linear object...  Checking char\_output\_projection.in\_features...  Checking char\_output\_projection.out\_features...  Checking char\_output\_projection.bias...  Checking char\_output\_projection.bias shape...  All tests passed!  2a -- Test correct initialization of self.decoderCharEmb (hidden) (0.4/0.4)  Initializing CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Using batch\_size=5  Initializing student's CharDecoder...  Testing student's initialization of self.decoderCharEmb...  Checking decoderCharEmb is a nn.Embedding object...  Checking decoderCharEmb.num\_embeddings...  Checking decoderCharEmb.embedding\_dim...  Checking decoderCharEmb.padding\_idx...  All tests passed!  2a -- Test shapes of initialized layers (public) (0.4/0.4)  Initializing CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Using batch\_size=5  Initializing student's CharDecoder...  Expected charDecoder.input\_size=3, got 3  Expected charDecoder.hidden\_size=3, got 3  Expected char\_output\_projection.in\_features=3, got 3  Expected char\_output\_projection.out\_features=30, got 30  Expected decoderCharEmb.num\_embeddings=30, got 30  Expected decoderCharEmb.embedding\_dim=3, got 3  All tests passed!  2a -- Test correct initialization of self.target\_vocab (hidden) (0.4/0.4)  Initializing CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Using batch\_size=5  Initializing student's CharDecoder...  Testing student's initialization of target\_vocab...  Checking self.target\_vocab is equal to the target\_vocab passed in...  All tests passed!  2b -- Test decoder cell state returned by forward (hidden) (0.75/0.75)  Initializing solution's CharDecoder...  Initializing student's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Running student's forward function...  Running solution's forward function...  Hyperparameters:  Batch Size: 5  Initializing Student & Solution CharDecoder with following parameters:  Hidden Size: 7  Character Embedding Size: 20  Target Vocab of Length: 30  Running Model Forward with:  Target Chars: tensor([[ 1, 19, 23, 7, 19, 17, 16, 4, 17, 7, 11, 19, 23, 23, 26, 17, 17, 22,  2, 0, 0],  [ 1, 6, 4, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 19, 6, 19, 22, 4, 26, 6, 11, 9, 7, 7, 4, 12, 17, 11, 6, 9,  2, 0, 0],  [ 1, 11, 11, 11, 17, 26, 22, 7, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 23, 9, 9, 7, 12, 4, 4, 26, 12, 9, 6, 2, 0, 0, 0, 0, 0,  0, 0, 0]])  Decoder Initial Hidden State: (tensor([[[ 0.3035, -0.2470, -0.3217, 0.8342, 0.0149, -0.1532, 1.6928],  [-0.7161, 1.7399, -0.5616, 0.7198, 1.4556, 1.0760, 1.3860],  [-0.0228, -0.4944, -1.2735, 1.1543, 0.0233, -0.6302, 0.8885],  [ 0.1748, -2.0528, 0.0386, -0.9440, 0.5850, -0.5959, -1.8833],  [ 0.5549, -0.9462, -0.7394, 0.7492, 0.6955, 1.5209, 0.3852],  [-0.3152, 0.0390, -0.1716, -0.8425, 1.2268, 2.1754, -0.4421],  [-1.4505, 0.8095, -0.7144, 0.7041, -1.2723, -0.7134, -0.2687],  [-0.7011, -0.9031, -1.2791, -0.4035, -0.6025, -0.4698, 0.5733],  [-0.2162, -0.2958, -0.6787, -0.2703, -0.1524, 0.1105, -0.9913],  [-0.8879, -0.7560, 0.5877, 1.5159, 0.5368, 1.7698, 1.3942],  [-0.4475, -0.8029, -0.0714, 0.4539, 0.4942, -0.0994, 0.5940],  [ 0.8260, 1.2241, -0.1431, 0.9761, 0.6290, -0.2798, 0.2504],  [ 1.7242, 2.1783, -0.6140, 0.9940, -0.9766, 0.4351, -1.0596],  [ 0.9594, -1.4276, 0.8310, -1.4381, 0.6213, -1.5297, 1.0560],  [ 2.0524, -0.3510, -1.1979, 0.5597, 0.4875, -1.5288, 0.8938],  [ 0.8216, 1.0384, 0.3729, -1.4194, -0.2249, -0.5342, -0.0788],  [-0.4675, -0.3125, -2.4208, -0.6981, -0.8373, -0.4473, -0.1180],  [ 0.8932, 0.4594, -0.1176, 0.0582, -0.1127, 0.6009, 0.0546],  [-0.5670, -1.0099, 1.3880, -0.4120, 0.1785, 0.2434, 2.0407],  [-0.7569, -1.8586, 1.3976, -0.8236, -0.3667, 0.5761, 0.3662],  [-1.2378, 1.4525, -0.1425, 0.2661, -0.9247, 0.6715, 0.4190]]]), tensor([[[ 1.0370, 0.1266, 0.9138, 0.1524, -1.7770, -0.2701, 1.1325],  [ 1.2827, 0.7584, -0.2537, -0.6027, 0.4112, -0.4786, -0.3798],  [-0.6784, -1.0526, 0.3302, 0.3409, 0.0966, 0.8048, -0.9309],  [-0.7496, -1.1346, 0.4310, 0.2429, 0.2790, 0.3316, -0.0520],  [-0.3900, 0.4596, 1.5001, 0.3899, 0.8540, -1.2495, -1.8244],  [ 1.9354, 0.3297, -0.5037, 0.5825, -2.6750, 0.2698, -1.1496],  [ 0.5576, -0.4424, -1.1939, 0.5121, 0.3451, -1.1712, -0.5270],  [ 0.3645, -1.3919, -0.1360, 0.5022, 0.3100, 0.0807, 0.1372],  [-0.2191, 1.0261, -0.1593, 1.6456, 0.8096, -0.5084, -0.3842],  [-0.1862, 0.3175, 0.5438, 0.5824, 0.2559, -0.7650, 1.6420],  [-1.3696, -0.1333, 2.7724, -0.0430, 0.4399, 0.4453, -1.7399],  [ 0.6852, 1.6848, -0.5546, 0.0731, -0.5317, -0.0354, -0.4692],  [ 3.5407, -0.3801, -0.2620, -0.5226, 2.1914, -0.9427, -0.3424],  [-1.6840, 0.6537, 0.3624, -1.1096, 0.1569, 0.1476, -0.6777],  [-0.1429, 0.3918, 0.2703, -1.9761, -0.6592, -0.6763, 0.9087],  [ 0.3067, 2.6992, -0.2381, -0.3891, -0.3006, -0.0498, 1.6682],  [-0.2175, 0.8391, -1.0962, -0.4409, -1.6116, 0.7782, -0.8651],  [-0.4189, -0.6947, 0.2014, -0.9889, -0.9279, -1.0297, -1.0448],  [-0.8124, 0.4942, -0.2674, 0.2319, 1.0128, 0.7808, -1.6109],  [-1.8808, 1.4752, 1.6363, -0.9645, 1.1408, 0.4157, -0.3488],  [ 0.1302, 0.3486, 0.5818, -0.3935, 0.4536, -1.1786, 0.7898]]]))  ----------------------------------------  Checking decoder hidden cell state...  Student decoder hidden cell state: tensor([[[ 1.6884, 0.0766, -0.3285, -0.7625, -0.2821, 0.2282, -0.0302],  [ 0.2769, 0.8490, 0.0743, 0.1813, 0.6697, 0.1730, 0.4753],  [ 0.1493, 0.3216, -0.2070, 0.1048, 0.0390, 0.0944, -0.5217],  [ 0.0423, 0.0538, -0.1941, -0.1190, 0.0652, -0.0338, -0.7918],  [-0.5782, 0.1250, 0.0094, 0.7261, -0.3543, -0.1709, -0.6508],  [ 0.1368, -0.1815, -1.0691, -0.2737, -0.0151, 0.0184, -0.9072],  [-0.4807, 0.4744, -0.7372, 0.0451, -0.1502, -0.1408, -0.0312],  [-0.5023, 0.5424, -0.7293, 0.5284, -0.4406, -0.0911, -0.1710],  [-0.2236, 0.1240, -0.5729, 0.4613, 0.3534, -0.0114, -0.7460],  [ 0.3368, -0.0876, -0.7326, -0.2422, -0.1281, 0.1696, -1.1630],  [ 0.0612, 0.2838, -0.1040, 0.0449, -0.0267, -0.1779, -0.7342],  [-0.1083, 0.3782, -0.1811, 0.5675, -0.5149, -0.0749, -0.1457],  [-0.2262, 0.2909, -0.2478, 0.7591, 0.5502, 0.4602, -0.8165],  [-0.0349, 0.3142, -0.2656, -0.1727, -0.1512, -0.0128, -0.4757],  [ 0.0637, 0.3994, -0.1167, -0.0305, -0.2806, -0.0115, -0.3994],  [ 0.1769, 0.4404, -0.1379, 0.0753, -0.2365, 0.1623, -0.3279],  [ 0.0571, 0.4693, -0.1787, 0.0820, -0.2817, 0.0807, -0.2407],  [-0.0665, 0.2835, -0.2140, -0.2813, -0.1777, -0.0764, -0.3993],  [-0.1445, 0.3172, -0.0896, 0.2068, -0.1306, 0.0948, -0.7873],  [-0.1032, 0.3885, -0.1785, -0.0883, -0.2023, -0.0100, -0.4289],  [ 0.0132, 0.4124, -0.1478, -0.0143, -0.2389, 0.0596, -0.3497]]])  Solution decoder hidden cell state: tensor([[[ 1.6884, 0.0766, -0.3285, -0.7625, -0.2821, 0.2282, -0.0302],  [ 0.2769, 0.8490, 0.0743, 0.1813, 0.6697, 0.1730, 0.4753],  [ 0.1493, 0.3216, -0.2070, 0.1048, 0.0390, 0.0944, -0.5217],  [ 0.0423, 0.0538, -0.1941, -0.1190, 0.0652, -0.0338, -0.7918],  [-0.5782, 0.1250, 0.0094, 0.7261, -0.3543, -0.1709, -0.6508],  [ 0.1368, -0.1815, -1.0691, -0.2737, -0.0151, 0.0184, -0.9072],  [-0.4807, 0.4744, -0.7372, 0.0451, -0.1502, -0.1408, -0.0312],  [-0.5023, 0.5424, -0.7293, 0.5284, -0.4406, -0.0911, -0.1710],  [-0.2236, 0.1240, -0.5729, 0.4613, 0.3534, -0.0114, -0.7460],  [ 0.3368, -0.0876, -0.7326, -0.2422, -0.1281, 0.1696, -1.1630],  [ 0.0612, 0.2838, -0.1040, 0.0449, -0.0267, -0.1779, -0.7342],  [-0.1083, 0.3782, -0.1811, 0.5675, -0.5149, -0.0749, -0.1457],  [-0.2262, 0.2909, -0.2478, 0.7591, 0.5502, 0.4602, -0.8165],  [-0.0349, 0.3142, -0.2656, -0.1727, -0.1512, -0.0128, -0.4757],  [ 0.0637, 0.3994, -0.1167, -0.0305, -0.2806, -0.0115, -0.3994],  [ 0.1769, 0.4404, -0.1379, 0.0753, -0.2365, 0.1623, -0.3279],  [ 0.0571, 0.4693, -0.1787, 0.0820, -0.2817, 0.0807, -0.2407],  [-0.0665, 0.2835, -0.2140, -0.2813, -0.1777, -0.0764, -0.3993],  [-0.1445, 0.3172, -0.0896, 0.2068, -0.1306, 0.0948, -0.7873],  [-0.1032, 0.3885, -0.1785, -0.0883, -0.2023, -0.0100, -0.4289],  [ 0.0132, 0.4124, -0.1478, -0.0143, -0.2389, 0.0596, -0.3497]]])  Passed decoder cell state test  2b -- Test decoder hidden state returned by forward (hidden) (0.75/0.75)  Initializing solution's CharDecoder...  Initializing student's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Running student's forward function...  Running solution's forward function...  Hyperparameters:  Batch Size: 5  Initializing Student & Solution CharDecoder with following parameters:  Hidden Size: 7  Character Embedding Size: 20  Target Vocab of Length: 30  Running Model Forward with:  Target Chars: tensor([[ 1, 19, 23, 7, 19, 17, 16, 4, 17, 7, 11, 19, 23, 23, 26, 17, 17, 22,  2, 0, 0],  [ 1, 6, 4, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 19, 6, 19, 22, 4, 26, 6, 11, 9, 7, 7, 4, 12, 17, 11, 6, 9,  2, 0, 0],  [ 1, 11, 11, 11, 17, 26, 22, 7, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 23, 9, 9, 7, 12, 4, 4, 26, 12, 9, 6, 2, 0, 0, 0, 0, 0,  0, 0, 0]])  Decoder Initial Hidden State: (tensor([[[ 0.3035, -0.2470, -0.3217, 0.8342, 0.0149, -0.1532, 1.6928],  [-0.7161, 1.7399, -0.5616, 0.7198, 1.4556, 1.0760, 1.3860],  [-0.0228, -0.4944, -1.2735, 1.1543, 0.0233, -0.6302, 0.8885],  [ 0.1748, -2.0528, 0.0386, -0.9440, 0.5850, -0.5959, -1.8833],  [ 0.5549, -0.9462, -0.7394, 0.7492, 0.6955, 1.5209, 0.3852],  [-0.3152, 0.0390, -0.1716, -0.8425, 1.2268, 2.1754, -0.4421],  [-1.4505, 0.8095, -0.7144, 0.7041, -1.2723, -0.7134, -0.2687],  [-0.7011, -0.9031, -1.2791, -0.4035, -0.6025, -0.4698, 0.5733],  [-0.2162, -0.2958, -0.6787, -0.2703, -0.1524, 0.1105, -0.9913],  [-0.8879, -0.7560, 0.5877, 1.5159, 0.5368, 1.7698, 1.3942],  [-0.4475, -0.8029, -0.0714, 0.4539, 0.4942, -0.0994, 0.5940],  [ 0.8260, 1.2241, -0.1431, 0.9761, 0.6290, -0.2798, 0.2504],  [ 1.7242, 2.1783, -0.6140, 0.9940, -0.9766, 0.4351, -1.0596],  [ 0.9594, -1.4276, 0.8310, -1.4381, 0.6213, -1.5297, 1.0560],  [ 2.0524, -0.3510, -1.1979, 0.5597, 0.4875, -1.5288, 0.8938],  [ 0.8216, 1.0384, 0.3729, -1.4194, -0.2249, -0.5342, -0.0788],  [-0.4675, -0.3125, -2.4208, -0.6981, -0.8373, -0.4473, -0.1180],  [ 0.8932, 0.4594, -0.1176, 0.0582, -0.1127, 0.6009, 0.0546],  [-0.5670, -1.0099, 1.3880, -0.4120, 0.1785, 0.2434, 2.0407],  [-0.7569, -1.8586, 1.3976, -0.8236, -0.3667, 0.5761, 0.3662],  [-1.2378, 1.4525, -0.1425, 0.2661, -0.9247, 0.6715, 0.4190]]]), tensor([[[ 1.0370, 0.1266, 0.9138, 0.1524, -1.7770, -0.2701, 1.1325],  [ 1.2827, 0.7584, -0.2537, -0.6027, 0.4112, -0.4786, -0.3798],  [-0.6784, -1.0526, 0.3302, 0.3409, 0.0966, 0.8048, -0.9309],  [-0.7496, -1.1346, 0.4310, 0.2429, 0.2790, 0.3316, -0.0520],  [-0.3900, 0.4596, 1.5001, 0.3899, 0.8540, -1.2495, -1.8244],  [ 1.9354, 0.3297, -0.5037, 0.5825, -2.6750, 0.2698, -1.1496],  [ 0.5576, -0.4424, -1.1939, 0.5121, 0.3451, -1.1712, -0.5270],  [ 0.3645, -1.3919, -0.1360, 0.5022, 0.3100, 0.0807, 0.1372],  [-0.2191, 1.0261, -0.1593, 1.6456, 0.8096, -0.5084, -0.3842],  [-0.1862, 0.3175, 0.5438, 0.5824, 0.2559, -0.7650, 1.6420],  [-1.3696, -0.1333, 2.7724, -0.0430, 0.4399, 0.4453, -1.7399],  [ 0.6852, 1.6848, -0.5546, 0.0731, -0.5317, -0.0354, -0.4692],  [ 3.5407, -0.3801, -0.2620, -0.5226, 2.1914, -0.9427, -0.3424],  [-1.6840, 0.6537, 0.3624, -1.1096, 0.1569, 0.1476, -0.6777],  [-0.1429, 0.3918, 0.2703, -1.9761, -0.6592, -0.6763, 0.9087],  [ 0.3067, 2.6992, -0.2381, -0.3891, -0.3006, -0.0498, 1.6682],  [-0.2175, 0.8391, -1.0962, -0.4409, -1.6116, 0.7782, -0.8651],  [-0.4189, -0.6947, 0.2014, -0.9889, -0.9279, -1.0297, -1.0448],  [-0.8124, 0.4942, -0.2674, 0.2319, 1.0128, 0.7808, -1.6109],  [-1.8808, 1.4752, 1.6363, -0.9645, 1.1408, 0.4157, -0.3488],  [ 0.1302, 0.3486, 0.5818, -0.3935, 0.4536, -1.1786, 0.7898]]]))  ----------------------------------------  Checking decoder hidden state...  Student decoder hidden state: tensor([[[ 0.2162, 0.0483, -0.1860, -0.5142, -0.1512, 0.1821, -0.0270],  [ 0.1477, 0.4475, 0.0353, 0.0966, 0.0832, 0.0694, 0.0673],  [ 0.1041, 0.2005, -0.1318, 0.0802, 0.0114, 0.0408, -0.2881],  [ 0.0310, 0.0347, -0.1276, -0.0911, 0.0186, -0.0153, -0.4042],  [-0.3359, 0.0229, 0.0086, 0.2837, -0.1154, -0.1117, -0.4367],  [ 0.0591, -0.1357, -0.5556, -0.2028, -0.0063, 0.0049, -0.3424],  [-0.2285, 0.1978, -0.0708, 0.0333, -0.0295, -0.0921, -0.0272],  [-0.2337, 0.2070, -0.0706, 0.3359, -0.0778, -0.0606, -0.1476],  [-0.1581, 0.0894, -0.3275, 0.2443, 0.1135, -0.0082, -0.3436],  [ 0.1543, -0.0672, -0.4653, -0.1826, -0.0526, 0.0406, -0.4293],  [ 0.0440, 0.1839, -0.0706, 0.0349, -0.0066, -0.0763, -0.3843],  [-0.0394, 0.0784, -0.1356, 0.2564, -0.1135, -0.0352, -0.0372],  [-0.1717, 0.0291, -0.1925, 0.2993, 0.3070, 0.1145, -0.4951],  [-0.0156, 0.1653, -0.1424, -0.0984, -0.0598, -0.0066, -0.2536],  [ 0.0274, 0.2103, -0.0639, -0.0173, -0.1053, -0.0059, -0.2152],  [ 0.0761, 0.2246, -0.0754, 0.0423, -0.0951, 0.0800, -0.1841],  [ 0.0237, 0.2356, -0.0943, 0.0470, -0.1077, 0.0399, -0.1325],  [-0.0294, 0.1540, -0.1163, -0.1624, -0.0672, -0.0397, -0.2160],  [-0.0671, 0.1552, -0.0498, 0.1133, -0.0530, 0.0467, -0.3758],  [-0.0446, 0.2008, -0.0966, -0.0514, -0.0762, -0.0051, -0.2289],  [ 0.0057, 0.2123, -0.0802, -0.0083, -0.0914, 0.0299, -0.1917]]])  Solution decoder hidden state: tensor([[[ 0.2162, 0.0483, -0.1860, -0.5142, -0.1512, 0.1821, -0.0270],  [ 0.1477, 0.4475, 0.0353, 0.0966, 0.0832, 0.0694, 0.0673],  [ 0.1041, 0.2005, -0.1318, 0.0802, 0.0114, 0.0408, -0.2881],  [ 0.0310, 0.0347, -0.1276, -0.0911, 0.0186, -0.0153, -0.4042],  [-0.3359, 0.0229, 0.0086, 0.2837, -0.1154, -0.1117, -0.4367],  [ 0.0591, -0.1357, -0.5556, -0.2028, -0.0063, 0.0049, -0.3424],  [-0.2285, 0.1978, -0.0708, 0.0333, -0.0295, -0.0921, -0.0272],  [-0.2337, 0.2070, -0.0706, 0.3359, -0.0778, -0.0606, -0.1476],  [-0.1581, 0.0894, -0.3275, 0.2443, 0.1135, -0.0082, -0.3436],  [ 0.1543, -0.0672, -0.4653, -0.1826, -0.0526, 0.0406, -0.4293],  [ 0.0440, 0.1839, -0.0706, 0.0349, -0.0066, -0.0763, -0.3843],  [-0.0394, 0.0784, -0.1356, 0.2564, -0.1135, -0.0352, -0.0372],  [-0.1717, 0.0291, -0.1925, 0.2993, 0.3070, 0.1145, -0.4951],  [-0.0156, 0.1653, -0.1424, -0.0984, -0.0598, -0.0066, -0.2536],  [ 0.0274, 0.2103, -0.0639, -0.0173, -0.1053, -0.0059, -0.2152],  [ 0.0761, 0.2246, -0.0754, 0.0423, -0.0951, 0.0800, -0.1841],  [ 0.0237, 0.2356, -0.0943, 0.0470, -0.1077, 0.0399, -0.1325],  [-0.0294, 0.1540, -0.1163, -0.1624, -0.0672, -0.0397, -0.2160],  [-0.0671, 0.1552, -0.0498, 0.1133, -0.0530, 0.0467, -0.3758],  [-0.0446, 0.2008, -0.0966, -0.0514, -0.0762, -0.0051, -0.2289],  [ 0.0057, 0.2123, -0.0802, -0.0083, -0.0914, 0.0299, -0.1917]]])  Passed decoder hidden state test  2b -- Test logits returned by forward (hidden) (1.25/1.25)  Initializing solution's CharDecoder...  Initializing student's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Running student's forward function...  Running solution's forward function...  Hyperparameters:  Batch Size: 5  Initializing Student & Solution CharDecoder with following parameters:  Hidden Size: 7  Character Embedding Size: 20  Target Vocab of Length: 30  Running Model Forward with:  Target Chars: tensor([[ 1, 19, 23, 7, 19, 17, 16, 4, 17, 7, 11, 19, 23, 23, 26, 17, 17, 22,  2, 0, 0],  [ 1, 6, 4, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 19, 6, 19, 22, 4, 26, 6, 11, 9, 7, 7, 4, 12, 17, 11, 6, 9,  2, 0, 0],  [ 1, 11, 11, 11, 17, 26, 22, 7, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0,  0, 0, 0],  [ 1, 23, 9, 9, 7, 12, 4, 4, 26, 12, 9, 6, 2, 0, 0, 0, 0, 0,  0, 0, 0]])  Decoder Initial Hidden State: (tensor([[[ 0.3035, -0.2470, -0.3217, 0.8342, 0.0149, -0.1532, 1.6928],  [-0.7161, 1.7399, -0.5616, 0.7198, 1.4556, 1.0760, 1.3860],  [-0.0228, -0.4944, -1.2735, 1.1543, 0.0233, -0.6302, 0.8885],  [ 0.1748, -2.0528, 0.0386, -0.9440, 0.5850, -0.5959, -1.8833],  [ 0.5549, -0.9462, -0.7394, 0.7492, 0.6955, 1.5209, 0.3852],  [-0.3152, 0.0390, -0.1716, -0.8425, 1.2268, 2.1754, -0.4421],  [-1.4505, 0.8095, -0.7144, 0.7041, -1.2723, -0.7134, -0.2687],  [-0.7011, -0.9031, -1.2791, -0.4035, -0.6025, -0.4698, 0.5733],  [-0.2162, -0.2958, -0.6787, -0.2703, -0.1524, 0.1105, -0.9913],  [-0.8879, -0.7560, 0.5877, 1.5159, 0.5368, 1.7698, 1.3942],  [-0.4475, -0.8029, -0.0714, 0.4539, 0.4942, -0.0994, 0.5940],  [ 0.8260, 1.2241, -0.1431, 0.9761, 0.6290, -0.2798, 0.2504],  [ 1.7242, 2.1783, -0.6140, 0.9940, -0.9766, 0.4351, -1.0596],  [ 0.9594, -1.4276, 0.8310, -1.4381, 0.6213, -1.5297, 1.0560],  [ 2.0524, -0.3510, -1.1979, 0.5597, 0.4875, -1.5288, 0.8938],  [ 0.8216, 1.0384, 0.3729, -1.4194, -0.2249, -0.5342, -0.0788],  [-0.4675, -0.3125, -2.4208, -0.6981, -0.8373, -0.4473, -0.1180],  [ 0.8932, 0.4594, -0.1176, 0.0582, -0.1127, 0.6009, 0.0546],  [-0.5670, -1.0099, 1.3880, -0.4120, 0.1785, 0.2434, 2.0407],  [-0.7569, -1.8586, 1.3976, -0.8236, -0.3667, 0.5761, 0.3662],  [-1.2378, 1.4525, -0.1425, 0.2661, -0.9247, 0.6715, 0.4190]]]), tensor([[[ 1.0370, 0.1266, 0.9138, 0.1524, -1.7770, -0.2701, 1.1325],  [ 1.2827, 0.7584, -0.2537, -0.6027, 0.4112, -0.4786, -0.3798],  [-0.6784, -1.0526, 0.3302, 0.3409, 0.0966, 0.8048, -0.9309],  [-0.7496, -1.1346, 0.4310, 0.2429, 0.2790, 0.3316, -0.0520],  [-0.3900, 0.4596, 1.5001, 0.3899, 0.8540, -1.2495, -1.8244],  [ 1.9354, 0.3297, -0.5037, 0.5825, -2.6750, 0.2698, -1.1496],  [ 0.5576, -0.4424, -1.1939, 0.5121, 0.3451, -1.1712, -0.5270],  [ 0.3645, -1.3919, -0.1360, 0.5022, 0.3100, 0.0807, 0.1372],  [-0.2191, 1.0261, -0.1593, 1.6456, 0.8096, -0.5084, -0.3842],  [-0.1862, 0.3175, 0.5438, 0.5824, 0.2559, -0.7650, 1.6420],  [-1.3696, -0.1333, 2.7724, -0.0430, 0.4399, 0.4453, -1.7399],  [ 0.6852, 1.6848, -0.5546, 0.0731, -0.5317, -0.0354, -0.4692],  [ 3.5407, -0.3801, -0.2620, -0.5226, 2.1914, -0.9427, -0.3424],  [-1.6840, 0.6537, 0.3624, -1.1096, 0.1569, 0.1476, -0.6777],  [-0.1429, 0.3918, 0.2703, -1.9761, -0.6592, -0.6763, 0.9087],  [ 0.3067, 2.6992, -0.2381, -0.3891, -0.3006, -0.0498, 1.6682],  [-0.2175, 0.8391, -1.0962, -0.4409, -1.6116, 0.7782, -0.8651],  [-0.4189, -0.6947, 0.2014, -0.9889, -0.9279, -1.0297, -1.0448],  [-0.8124, 0.4942, -0.2674, 0.2319, 1.0128, 0.7808, -1.6109],  [-1.8808, 1.4752, 1.6363, -0.9645, 1.1408, 0.4157, -0.3488],  [ 0.1302, 0.3486, 0.5818, -0.3935, 0.4536, -1.1786, 0.7898]]]))  ----------------------------------------  Checking logits...  Student logits: tensor([[[-0.3309, -0.0488, 0.0656, ..., -0.1646, 0.5454, -0.2907],  [-0.1601, -0.3306, 0.3753, ..., -0.3194, 0.1667, -0.0139],  [-0.2061, -0.2577, 0.2101, ..., -0.2567, 0.3174, -0.1110],  ...,  [-0.2889, -0.6223, 0.3046, ..., -0.3284, 0.0381, 0.1397],  [-0.1812, -0.1638, 0.2784, ..., -0.0553, 0.5319, -0.2509],  [-0.3480, -0.1330, 0.1977, ..., -0.2137, 0.3245, 0.0151]],  [[-0.2812, -0.0894, 0.1632, ..., -0.1656, 0.5245, -0.2904],  [-0.3547, -0.3413, 0.1476, ..., -0.2879, 0.2543, -0.0743],  [-0.2690, -0.3458, 0.1139, ..., -0.2516, 0.3058, -0.1690],  ...,  [-0.2275, -0.3771, 0.2614, ..., -0.2388, 0.2495, -0.0894],  [-0.2772, -0.2266, 0.2152, ..., -0.1293, 0.4157, -0.1667],  [-0.3235, -0.1822, 0.1678, ..., -0.2204, 0.3320, -0.0814]],  [[-0.2758, -0.1136, 0.2074, ..., -0.1547, 0.5098, -0.2736],  [-0.2852, -0.3288, 0.1740, ..., -0.2479, 0.3101, -0.1720],  [-0.3822, -0.3706, 0.0709, ..., -0.2690, 0.2822, -0.1079],  ...,  [-0.2136, -0.4962, 0.3825, ..., -0.2294, 0.1591, 0.0783],  [-0.3082, -0.2381, 0.2032, ..., -0.1477, 0.3854, -0.1370],  [-0.3076, -0.2077, 0.1708, ..., -0.2114, 0.3409, -0.1207]],  [[-0.2718, -0.1177, 0.2361, ..., -0.1446, 0.5036, -0.2588],  [-0.3730, -0.4360, 0.1733, ..., -0.2581, 0.2465, -0.0754],  [-0.3820, -0.4070, 0.1054, ..., -0.2711, 0.2598, -0.0906],  ...,  [-0.2685, -0.3447, 0.2628, ..., -0.2012, 0.2628, -0.0374],  [-0.3128, -0.2321, 0.2025, ..., -0.1592, 0.3717, -0.1262],  [-0.3028, -0.2133, 0.1838, ..., -0.2000, 0.3460, -0.1256]],  [[-0.2686, -0.1159, 0.2548, ..., -0.1371, 0.5011, -0.2482],  [-0.2155, -0.1476, 0.1366, ..., -0.3440, 0.2568, -0.1353],  [-0.2771, -0.2474, 0.2523, ..., -0.2333, 0.2751, -0.0562],  ...,  [-0.2903, -0.2719, 0.2485, ..., -0.1713, 0.3148, -0.0659],  [-0.3108, -0.2249, 0.2042, ..., -0.1660, 0.3651, -0.1221],  [-0.3013, -0.2133, 0.1940, ..., -0.1910, 0.3501, -0.1243]]])  Solution logits: tensor([[[-0.3309, -0.0488, 0.0656, ..., -0.1646, 0.5454, -0.2907],  [-0.1601, -0.3306, 0.3753, ..., -0.3194, 0.1667, -0.0139],  [-0.2061, -0.2577, 0.2101, ..., -0.2567, 0.3174, -0.1110],  ...,  [-0.2889, -0.6223, 0.3046, ..., -0.3284, 0.0381, 0.1397],  [-0.1812, -0.1638, 0.2784, ..., -0.0553, 0.5319, -0.2509],  [-0.3480, -0.1330, 0.1977, ..., -0.2137, 0.3245, 0.0151]],  [[-0.2812, -0.0894, 0.1632, ..., -0.1656, 0.5245, -0.2904],  [-0.3547, -0.3413, 0.1476, ..., -0.2879, 0.2543, -0.0743],  [-0.2690, -0.3458, 0.1139, ..., -0.2516, 0.3058, -0.1690],  ...,  [-0.2275, -0.3771, 0.2614, ..., -0.2388, 0.2495, -0.0894],  [-0.2772, -0.2266, 0.2152, ..., -0.1293, 0.4157, -0.1667],  [-0.3235, -0.1822, 0.1678, ..., -0.2204, 0.3320, -0.0814]],  [[-0.2758, -0.1136, 0.2074, ..., -0.1547, 0.5098, -0.2736],  [-0.2852, -0.3288, 0.1740, ..., -0.2479, 0.3101, -0.1720],  [-0.3822, -0.3706, 0.0709, ..., -0.2690, 0.2822, -0.1079],  ...,  [-0.2136, -0.4962, 0.3825, ..., -0.2294, 0.1591, 0.0783],  [-0.3082, -0.2381, 0.2032, ..., -0.1477, 0.3854, -0.1370],  [-0.3076, -0.2077, 0.1708, ..., -0.2114, 0.3409, -0.1207]],  [[-0.2718, -0.1177, 0.2361, ..., -0.1446, 0.5036, -0.2588],  [-0.3730, -0.4360, 0.1733, ..., -0.2581, 0.2465, -0.0754],  [-0.3820, -0.4070, 0.1054, ..., -0.2711, 0.2598, -0.0906],  ...,  [-0.2685, -0.3447, 0.2628, ..., -0.2012, 0.2628, -0.0374],  [-0.3128, -0.2321, 0.2025, ..., -0.1592, 0.3717, -0.1262],  [-0.3028, -0.2133, 0.1838, ..., -0.2000, 0.3460, -0.1256]],  [[-0.2686, -0.1159, 0.2548, ..., -0.1371, 0.5011, -0.2482],  [-0.2155, -0.1476, 0.1366, ..., -0.3440, 0.2568, -0.1353],  [-0.2771, -0.2474, 0.2523, ..., -0.2333, 0.2751, -0.0562],  ...,  [-0.2903, -0.2719, 0.2485, ..., -0.1713, 0.3148, -0.0659],  [-0.3108, -0.2249, 0.2042, ..., -0.1660, 0.3651, -0.1221],  [-0.3013, -0.2133, 0.1940, ..., -0.1910, 0.3501, -0.1243]]])  Passed logits test  2b --- Test shapes of outputs returned by forward (public) (0.25/0.25)  Hyperparameters:  Batch Size: 5  Initializing Student & Solution CharDecoder with following parameters:  Hidden Size: 3  Character Embedding Size: 3  Target Vocab of Length: 30  ----------------------------------------  Initializing student's CharDecoder...  Running student's forward function...  Passed!  2c -- Test output of train\_forward (hidden) (4.0/4.0)  Batch Size: 5  Character Embedding Size: 20  Hidden Size: 128  Character-level target vocab with size: 30  Input to train\_forward is:  Sequence of Length: 21  Sequence: tensor([[ 1, 1, 1, 1, 1],  [19, 6, 19, 11, 23],  [23, 4, 6, 11, 9],  [ 7, 2, 19, 11, 9],  [19, 0, 22, 17, 7],  [17, 0, 4, 26, 12],  [16, 0, 26, 22, 4],  [ 4, 0, 6, 7, 4],  [17, 0, 11, 2, 26],  [ 7, 0, 9, 0, 12],  [11, 0, 7, 0, 9],  [19, 0, 7, 0, 6],  [23, 0, 4, 0, 2],  [23, 0, 12, 0, 0],  [26, 0, 17, 0, 0],  [17, 0, 11, 0, 0],  [17, 0, 6, 0, 0],  [22, 0, 9, 0, 0],  [ 2, 0, 2, 0, 0],  [ 0, 0, 0, 0, 0],  [ 0, 0, 0, 0, 0]])  ----------------------------------------  Initializing solution's CharDecoder...  Initializing student's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Using model solution's forward() function in student's CharDecoder  Running solution's train\_forward()...  Running student's train\_forward()...  Solution loss value: 200.39015197753906  Student loss value: 200.39015197753906  Passed!  2c -- Test shape of output of train\_forward (public) (1.0/1.0)  Batch Size: 5  Character Embedding Size: 3  Hidden Size: 3  Character-level target vocab with size: 30  Input to train\_forward is:  Sequence of Length: 4  Sequence: tensor([[0, 0, 0, 0, 0],  [0, 0, 0, 0, 0],  [0, 0, 0, 0, 0],  [0, 0, 0, 0, 0]])  ----------------------------------------  Initializing student's CharDecoder...  Running student's train\_forward()...  Expected loss with shape: 0  Student's loss has shape: 0  2d -- Test output of decode\_greedy for exact match (hidden) (3.0/3.0)  Setting batch size to 5  Initializing student's CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Initializing student's CharDecoder...  Initializing solution's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Using model solution's forward() and train\_forward() functions in student's CharDecoder  Passing decode\_greedy these initialStates:  (tensor([[[ 37.0292, 25.6323, -31.6395],  [-40.0942, -34.1682, -49.3439],  [-38.5819, -12.3649, 33.7439],  [ 8.3691, -38.0303, -40.1112],  [ 24.8738, -37.1921, -6.1564]]]), tensor([[[ 23.9853, -23.1406, -5.4520],  [ -4.3522, -11.8292, -25.3516],  [-44.5719, -40.4179, -26.7731],  [ 48.2919, -24.1507, -33.5764],  [ 12.1197, 13.7805, 27.3955]]]))  Running solution's decode\_greedy function...  Running student's decode\_greedy function...  Solution decode\_greedy output: ['uudddzzzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'ddddddzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz']  Student decode\_greedy output: ['uudddzzzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'ddddddzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz']  Passed - receiving credit for correct exact match (all characters of each string in decodedWords match solution)  2d -- Test output of decode\_greedy for partial match (hidden) (3.5/3.5)  Setting batch size to 5  Initializing student's CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Initializing student's CharDecoder...  Initializing solution's CharDecoder...  Copying model solution's target\_vocab, charDecoder, char\_output\_projection and decoderCharEmb to student's CharDecoder  Using model solution's forward() and train\_forward() functions in student's CharDecoder  Passing decode\_greedy these initialStates:  (tensor([[[ 37.0292, 25.6323, -31.6395],  [-40.0942, -34.1682, -49.3439],  [-38.5819, -12.3649, 33.7439],  [ 8.3691, -38.0303, -40.1112],  [ 24.8738, -37.1921, -6.1564]]]), tensor([[[ 23.9853, -23.1406, -5.4520],  [ -4.3522, -11.8292, -25.3516],  [-44.5719, -40.4179, -26.7731],  [ 48.2919, -24.1507, -33.5764],  [ 12.1197, 13.7805, 27.3955]]]))  Running solution's decode\_greedy function...  Running student's decode\_greedy function...  Solution decode\_greedy output: ['uudddzzzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'ddddddzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz']  Student decode\_greedy output: ['uudddzzzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzzzzz', 'ddddddzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz']  Just looking at first 18 characters...  First 18 chars of solution decode\_greedy output: ['uudddzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzz', 'dddddddzzzzzzzzzzz', 'ddddddzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzz']  First 18 chars of student's decode\_greedy output: ['uudddzzzzzzzzzzzzz', 'dddddddzzzzzzzzzzz', 'dddddddzzzzzzzzzzz', 'ddddddzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzz']  Passed - receiving credit for correct partial match (first 18 characters of each string in decodedWords match solution)  2d -- Test shape of output of decode\_greedy (public) (0.5/0.5)  Setting batch size to 5  Initializing student's CharDecoder with hidden\_size=3, char\_embedding\_size=3, target\_vocab from sanity\_check\_en\_es\_data/char\_vocab\_sanity\_check.json  Initializing student's CharDecoder...  Passing decode\_greedy this initialStates:  (tensor([[[0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.]]]), tensor([[[0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.],  [0., 0., 0.]]]))  Running student's decode\_greedy function...  Student's output from decode\_greedy: ['zzzzzzzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz', 'zzzzzzzzzzzzzzzzzzzzz']  Expect decode\_greedy output to be a list length 5  Student's decode\_greedy output is a list length 5  Passed!  2e -- BLEU score on tiny test set is over 99 (public) (3.0/3.0)  BLEU 99.29792465574434  2f -- BLEU score above 16 (public) (4.0/4.0)  Your BLEU score: 16.368168764760128  2f -- BLEU score above 10 (public) (2.0/2.0)  Your BLEU score: 16.368168764760128 |