



## PROJECT SPECIFICATION

## Generate TV Scripts

## Required Files and Tests

CRITERIA	MEETS SPECIFICATIONS
Have all project files been included with the submission?	The project submission contains the project notebook, called "dInd_tv_script_generation.ipynb".
Have all the unit tests in the project passed?	All the unit tests in project have passed.

## Preprocessing

CRITERIA	MEETS SPECIFICATIONS
Does the <code>create_lookup_tables</code> function correctly create lookup dictionaries?	<p>The function <code>create_lookup_tables</code> create two dictionaries:</p> <ul style="list-style-type: none"> <li>Dictionary to go from the words to an id, we'll call <code>vocab_to_int</code></li> <li>Dictionary to go from the id to word, we'll call <code>int_to_vocab</code></li> </ul> <p>The function <code>create_lookup_tables</code> return these dictionaries in the a tuple (<code>vocab_to_int</code>, <code>int_to_vocab</code>)</p>
Does the project correctly create a token dictionary?	The function <code>token_lookup</code> returns a dict that can correctly tokenizes the provided symbols.

## Build the Neural Network

CRITERIA	MEETS SPECIFICATIONS
Does the project create the correct input tensors?	<p>Implemented the <code>get_inputs</code> function to create TF Placeholders for the Neural Network with the following placeholders:</p> <ul style="list-style-type: none"> <li>Input text placeholder named "input" using the TF Placeholder name parameter.</li> <li>Targets placeholder</li> <li>Learning Rate placeholder</li> </ul> <p>The <code>get_inputs</code> function return the placeholders in the following the tuple (Input, Targets, LearningRate)</p>

CRITERIA	MEETS SPECIFICATIONS
Does the project correctly create a RNN Cell and initialize it?	<p>The <code>get_init_cell</code> function does the following:</p> <ul style="list-style-type: none"> <li>Stacks one or more BasicLSTMCells in a MultiRNNCell using the RNN size <code>rnn_size</code>.</li> <li>Initializes Cell State using the MultiRNNCell's <code>zero_state</code> function</li> <li>The name "initial_state" is applied to the initial state.</li> <li>The <code>get_init_cell</code> function return the cell and initial state in the following tuple (Cell, InitialState)</li> </ul>
Does the project correctly apply word embeddings?	The function <code>get_embed</code> applies embedding to <code>input_data</code> and returns embedded sequence.
Does the project correctly build a RNN using the RNN Cell?	<p>The function <code>build_rnn</code> does the following:</p> <ul style="list-style-type: none"> <li>Builds the RNN using the <code>tf.nn.dynamic_rnn</code>.</li> <li>Applies the name "final_state" to the final state.</li> <li>Returns the outputs and final_state state in the following tuple (Outputs, FinalState)</li> </ul>
Does the project correctly build the neural network?	<p>The <code>build_nn</code> function does the following in order:</p> <ul style="list-style-type: none"> <li>Apply embedding to <code>input_data</code> using <code>get_embed</code> function.</li> <li>Build RNN using cell using <code>build_rnn</code> function.</li> <li>Apply a fully connected layer with a linear activation and <code>vocab_size</code> as the number of outputs.</li> <li>Return the logits and final state in the following tuple (Logits, FinalState)</li> </ul>
Does the project correctly batch the data?	<p>The <code>get_batches</code> function create batches of input and targets using <code>int_text</code>. The batches should be a Numpy array of tuples. Each tuple is (batch of input, batch of target).</p> <ul style="list-style-type: none"> <li>The first element in the tuple is a single batch of input with the shape [batch size, sequence length]</li> <li>The second element in the tuple is a single batch of targets with the shape [batch size, sequence length]</li> </ul>

### Neural Network Training

CRITERIA	MEETS SPECIFICATIONS
Does the project use reasonable hyperparameters?	<ul style="list-style-type: none"> <li>Enough epochs to get near a minimum in the training loss, no real upper limit on this. Just need to make sure the training loss is low and not improving much with more training.</li> <li>Batch size is large enough to train efficiently, but small enough to fit the data in memory. No real "best" value here, depends on GPU memory usually.</li> <li>Size of the RNN cells (number of units in the hidden layers) is large enough to fit the data well. Again, no real "best" value.</li> <li>The sequence length (<code>seq_length</code>) here should be about the size of the length of sentences you want to generate. Should match the structure of the data.</li> </ul> <p>The learning rate shouldn't be too large because the training algorithm won't converge. But needs to be large enough that training doesn't take forever.</p> <p>Set <code>show_every_n_batches</code> to the number of batches the neural network should print progress.</p>

CRITERIA	MEETS SPECIFICATIONS
Does the project achieve a good loss?	The project gets a loss less than 1.0

### Generate TV Script

CRITERIA	MEETS SPECIFICATIONS
Does the project load the correct tensors?	"input:0", "initial_state:0", "final_state:0", and "probs:0" are all returned by <code>get_tensor_by_name</code> , in that order, and in a tuple
Does the project predict reasonable words?	The <code>pick_word</code> function predicts the next word correctly.
Does the project generate a TV script?	<p>The generated script looks similar to the TV script in the dataset.</p> <p>It doesn't have to be grammatically correct or make sense.</p>

---

[Student FAQ](#)