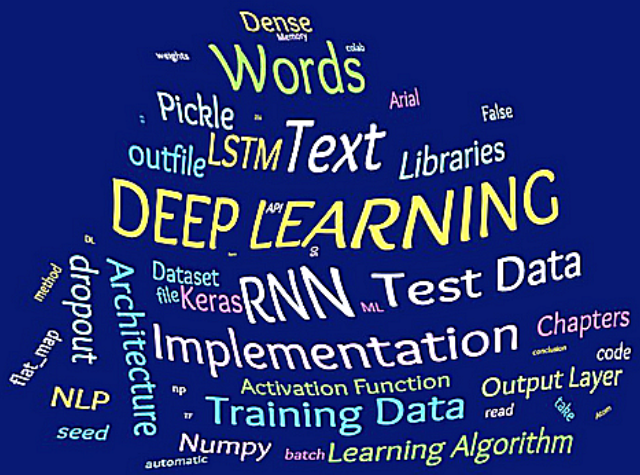
DeepSphere.AI
Enterprise AI and IIoT for Analytics

DEEP LEARNING

Implementing Recurrent Neural Network for Text Generation/Text Enrichment Using Tensorflow.



NLP

DeepSphere.AI
Enterprise AI and IIoT for Analytics

TABLE OF CONTENTS

DeepSphere.AI
Enterprise AI and IIoT for Analytics

DISCLAIMER

All software and hardware used or referenced in this guide belong to their respective vendor. We developed this guide based on our development infrastructure and this guide may or may not work on other systems and technical Infrastructure. We are not liable for any direct or indirect problems caused by users using this guide.

EXECUTIVE SUMMARY

The purpose of this document is to provide adequate information to users to implement a Recurrent Neural Network Model. To achieve this, we are using one of the most common and basic problem statement that occurs to any person involved in creating models having text data as Input in a Neural Network, an unsupervised Deep Learning model.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

BUSINESS PROBLEM

DeepSphere.AI
Enterprise AI and IIoT for Analytics

PROBLEM STATEMENT

Content Enrichment – make the given content simple, summarized, straightforward, and easy to follow without losing the content's Integrity.

BUSINESS CHALLENGES

- Analysis Paralysis
- Lack of time
- Data overload and complexity
- Misinterpretation of context
- Human intensive

BUSINESS CONTEXT

Example A) In a News Company, there is a content writer, who is writing the key content. So now we want to use Artificial Intelligence to rewrite the content based on a key paragraph into a well defined article.

Example B) In a Shakespeare Drama, the dialogues are written based on an event or a topic. Now we want to use Artificial Intelligence to rewrite this paragraph based on the target audience(based on their language,interest, etc.) into movies or dramas in a more enriched way.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

MODEL SELECTION

DeepSphere.AI
Enterprise AI and IIoT for Analytics

Model selection is the process of choosing between different machine learning approaches - e.g. Decision Tree, Logistic Regression, etc. - or choosing between different hyperparameters or sets of features for the same machine learning approach - e.g. deciding between the polynomial degrees/complexities for linear
regression.

The choice of the actual machine learning algorithm (e.g. SVM or logistic regression) is less important than you'd think - there may be a "best" algorithm for a particular problem, but often its performance is not much better than other well-performing approaches for that problem.

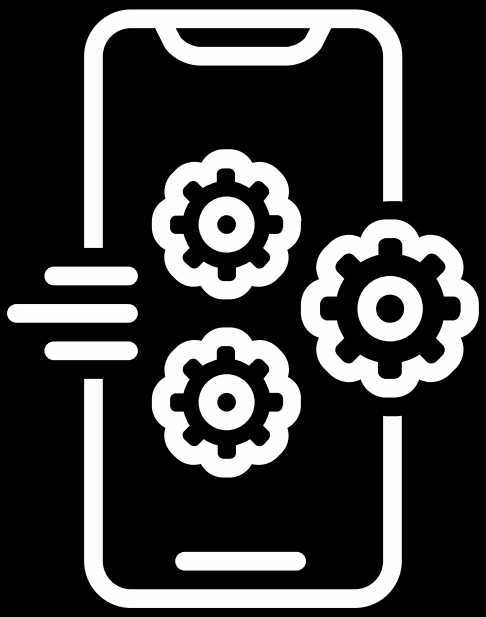
There may be certain qualities you look for in a model:
- Interpretable - can we see or understand why the model is making the decisions it makes?
- Simple - easy to explain and understand
- Accurate
- Fast (to train and test)
- Scalable (it can beapplied to a large dataset)

Our Problem here is to generate text automatically. The Problem is to Identify the style in which an author writes and generate text based on that style.
This Type of Problem can be Solved by the following Model.
- Recurrent Neural Network

Recurrent Neural Networks are a technique of working with sequential data, because they can remember the last inputs via an internal memory. They achieve state of the art performance on pretty much every sequential problem and are used by most major companies. A RNN can be used to generate text in the style of a specific author.The steps of creating a text generation RNN are:
1. Creating or gathering a dataset
2. Building the RNN model
3. Creating new text by taking a random sentence as a starting point

DeepSphere.AI
Enterprise AI and IIoT for Analytics

FEATURE ENGINEERING



DeepSphere.AI
Enterprise AI and IIoT for Analytics

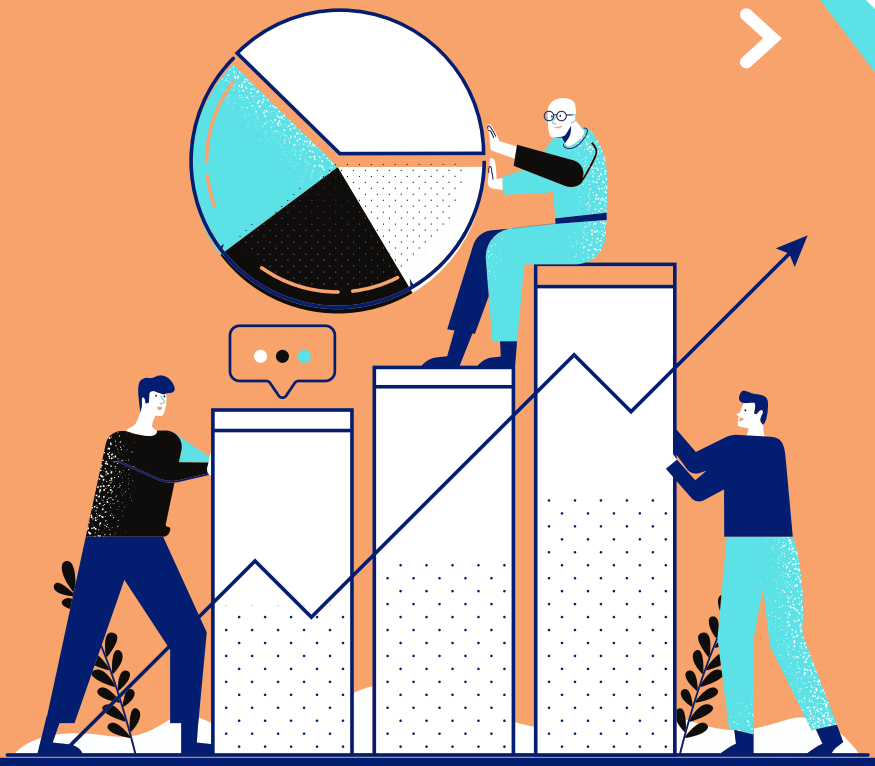
Feature engineering is the process of using domain knowledge of the data to create features that make machine learning algorithms work.

If feature engineering is done correctly, it increases the predictive power of machine learning algorithms by creating features from raw data that help facilitate the machine learning process. Feature engineering is an art.

Advantages of Feature Engineering:

- Good features provide you with the flexibility of choosing and algorithm; even if you choose a less complex model, you get good accuracy.

- If you choose good features, then even simple ML algorithms do well.
- Better features will lead you to better accuracy. You should spend more time on features engineering to generate the appropriate features for your dataset. If you derive the best and appropriate features, you have won most of the battle.



DATA MANAGEMENT

There are three types of data sets Training, Test and Dev that are used at various stage of Implementation. Training dataset is the largest of three of them, while test data functions as seal of approval and you don't need to use till the end of the development.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

TRAINING DATASET

The training data set is the actual dataset used to train the model for performing various Machine Learning Operations (Regression, Classification, Clustering etc.). This is the actual data with which the models learn with various API and algorithms to train the machine to work automatically.

So she was considering in her own mind (as well as she could, for the hot day made her feel very sleepy and stupid), whether the pleasure of making a daisy-chain would be worth the trouble of getting up and picking the daisies, when suddenly a White Rabbit with pink eyes ran close by her.

There was nothing so VERY remarkable in that; nor did Alice think it so VERY much out of the way to hear the Rabbit say to itself, 'Oh dear! Oh dear! I shall be late!' (when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually TOOK A WATCH OUT OF ITS WAISTCOAT-POCKET, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.

In another moment down went Alice after it, never once considering how in the world she was to get out again.

The rabbit-hole went straight on like a tunnel for some way, and then dipped suddenly down, so suddenly that Alice had not a moment to think about stopping herself before she found herself falling down a very deep well.

Either the well was very deep, or she fell very slowly, for she had plenty of time as she went down to look about her and to wonder what was going to happen next. First, she tried to look down and make out what she was coming to, but it was too dark to see anything; then she looked at the sides of the well, and noticed that they were filled with cupboards and book-shelves; here and there she saw maps and pictures hung upon pegs. She took down a jar from one of the shelves as she passed; it was labelled 'ORANGE MARMALADE', but to her great disappointment it was empty: she did not like to drop the jar for fear of killing somebody, so managed to put it into one of the cupboards as she fell past it.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

TEST DATASET

Test data set helps you to validate that the training has happened efficiently in terms of either accuracy, or precision so on. Actually, such data is used for testing the model whether it is responding or working appropriately or not.

```
CHAPTER I. Down the Rabbit-Hole

Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, 'and what is the use of a book,' thought Alice 'without pictures or
conversations?'
```

LEARNING ALGORITHM

- A self-learning (not a human developed code) code, performs data analysis and extracts patterns (business characteristics) in data for business application development - A Modern approach to application/software development.
- Automatically understands and extracts data patterns when data changes (change in style of writing) and performs data analysis based on the new/changed data set. - No code change required to implement changes that took place in the data

DeepSphere.AI
Enterprise AI and IIoT for Analytics

MACHINE LEARNING LIBRARIES USED

1. Numpy 1.18.5
2. Pickle
3. String
4. Tensorflow 2.3.0

NEURAL NETWORK MODEL USED:

Recurrent Neural Network

DeepSphere.AI
Enterprise AI and IIoT for Analytics

MODEL BUILDING BLOCKS

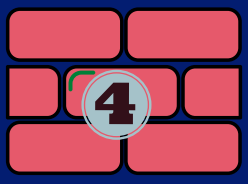
There are several technical and functional components involved in implementing this model. Here are the key building blocks to implement the model.

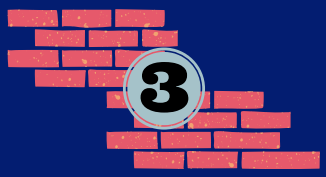
DeepSphere.AI
Enterprise AI and IIoT for Analytics

DEEP LEARNING BUILDING BLOCKS

TENSORFLOW 

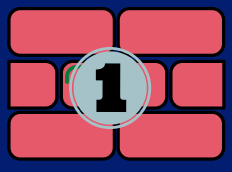
TRAINING DATA

NUMPY & STRING 



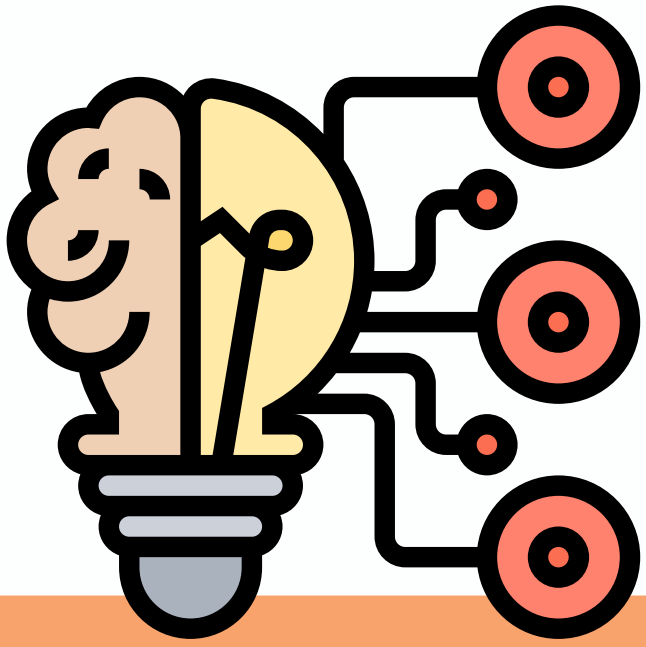
JUPYTER NOTEBOOK

GOOGLE CLOUD 



PYTHON

DeepSphere.AI
Enterprise AI and IIoT for Analytics

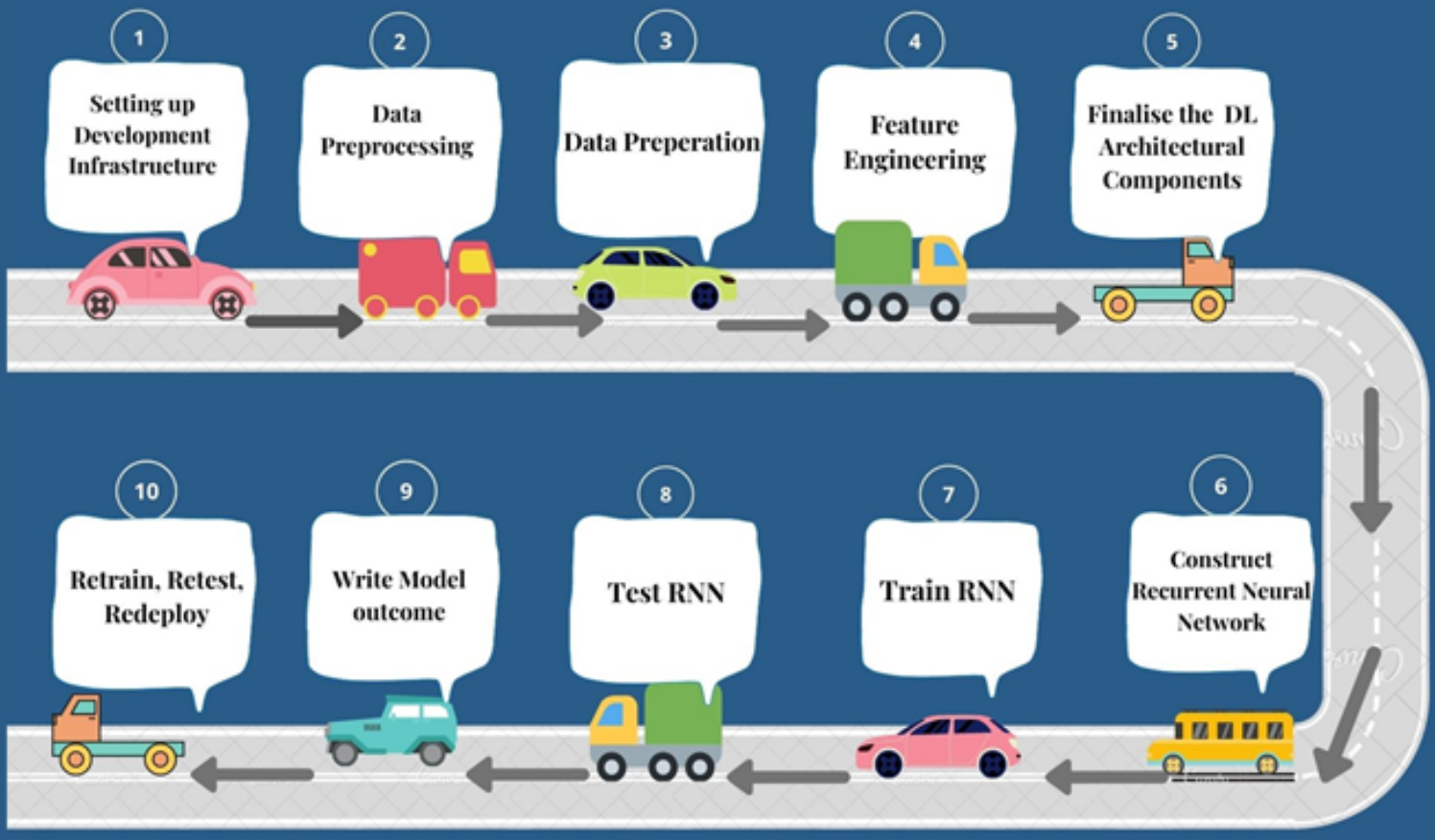


DEEP LEARNING IMPLEMENTATION STEPS

Here are the key steps that are involved to implement a deep learning model. You can customize these steps as needed and we have developed these steps for learning-purpose only.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

DEEP LEARNING IMPLEMENTATION STEPS



DeepSphere.AI
Enterprise AI and IIoT for Analytics

DL MODEL IMPLEMENTATION STEPS

STEP 1-SETTING UP DEVELOPMENT INFRASTRUCTURE:

For our Model Implementation we need the Following Libraries:

Numpy: NumPy is a python library used for working with arrays. It also has functions for working in the domain of linear algebra, fourier transform, and matrices. For Our Implementation we are using it for storing boolean arrays.

Pickle: The pickle module implements a fundamental, but powerful algorithm for serializing and de-serializing a Python object structure. "Pickling" is the process whereby a Python object hierarchy is converted into a byte stream, and "unpickling" is the inverse operation, whereby a byte stream is converted back into an object hierarchy. Pickling (and unpickling) is alternatively known as "serialization", "marshalling," or "flattening", however, to avoid confusion, the terms used here are "pickling" and "unpickling".

String : The string module contains a number of useful constants and classes, as well as some deprecated legacy functions that are also available as methods on strings. In addition, Python's built-in string classes support the sequence type methods described in the Sequence Types — str, unicode, list, tuple, bytearray, buffer, xrange section, and also the string-specific methods described in the String Methods section.

TensorFlow : TensorFlow is a Python library for fast numerical computing created and released by Google. It is a foundation library that can be used to create Deep Learning models directly or by using wrapper libraries that simplify the process built on top of TensorFlow.

Requests: The requests module allows you to send HTTP requests using Python.The HTTP request returns a Response Object with all the response data (content, encoding, status, etc).

DeepSphere.AI
Enterprise AI and IIoT for Analytics

textgeneration.py

```
/************************************************************************

FILE NAME      : textGeneration.py
Purpose        : To generate a paragraph of text automatically
Author         : DeepSphere.AI, Inc.
Date and Time  : 11/23/2020 10:00 hrs
Version        : 1.0

/************************************************************************

##########################################
#       Step-1 Importing the libraries     #
##########################################

import tensorflow as tf
import numpy as np
import os
import pickle
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, LSTM, Dropout
from string import punctuation
import requests

```

DeepSphere.AI
Enterprise AI and IIoT for Analytics

STEP 2 - DATA PREPROCESSING

Next immediate step after importing all libraries is loading the Training Data.. We are importing the Text data from the website with the use of requests module. We prepare the dataset by cleaning the dataset. We remove Punctuations, Convert all characters to lower.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

```
24  ######################################################################
25  #       Step-2 Loading the dataset and preparing the dataset     #
26  ######################################################################
27
28  vAR_content = requests.get("http://www.gutenberg.org/cache/epub/11/pg11.txt").text
29  open("wonderland.txt", "w", encoding="utf-8").write(vAR_content)
30  vAR_sequence_length = 100
31  vAR_BATCH_SIZE = 128
32  vAR_EPOCHS = 10
33  vAR_FILE_PATH = "wonderland.txt"
34  vAR_BASENAME = os.path.basename(vAR_FILE_PATH)
35  # read the data
36  vAR_text = open(vAR_FILE_PATH, encoding="utf-8").read()
37  # remove caps
38  vAR_text = vAR_text.lower()
39  # remove punctuation
40  vAR_text = vAR_text.translate(str.maketrans("", "", punctuation))
41  # print some stats
42  vAR_n_chars = len(vAR_text)
43  vAR_vocab = ''.join(sorted(set(vAR_text)))
44  print("unique_chars:", vAR_vocab)
45  vAR_n_unique_chars = len(vAR_vocab)
46  print("Number of characters:", vAR_n_chars)
47  print("Number of unique characters:", vAR_n_unique_chars)
48
```

DeepSphere.AI
Enterprise AI and IIoT for Analytics

STEP 3 - FEATURE ENGINEERING

Step 3 of the Implementation is Mapping From Character to Integer and Integer to Character. The sequences of characters must be encoded as integers.This means that each unique character will be assigned a specific integer value and each sequence of characters will be encoded as a sequence of integers. We can create the mapping given a sorted set of unique characters in the raw input data. The mapping is a dictionary of character values to integer values.Since we want to scale our code for larger datasets, we need to use tf.data API for efficient dataset handling, as a result, let's create a tf.data.Dataset object on this vAR_encoded_text array. And now this vAR_char_dataset object has all the characters of this dataset, let's try to print the first characters.This will take the very first 8 characters and print them out along with their integer representation. Next we need to construct our sequences, we want each input sample to be a sequence of characters of the length vAR_sequence_length and the output of a single character that is the next one. Luckily for us, we have to use tf.data.Dataset's batch() method to gather characters together:

DeepSphere.AI
Enterprise AI and IIoT for Analytics

```
49  ######################################################################
50  #       Step-3 Mapping characters to integers and vice versa     #
51  ######################################################################
52
53  # dictionary that converts characters to integers
54  vAR_char2int = {c: i for i, c in enumerate(vAR_vocab)}
55  # dictionary that converts integers to characters
56  vAR_int2char = {i: c for i, c in enumerate(vAR_vocab)}
57  # save these dictionaries for later generation
58  pickle.dump(vAR_char2int, open(f"{vAR_BASENAME}-char2int.pickle", "wb"))
59  pickle.dump(vAR_int2char, open(f"{vAR_BASENAME}-int2char.pickle", "wb"))
60  # convert all text into integers
61  vAR_encoded_text = np.array([vAR_char2int[c] for c in vAR_text])
62  # construct tf.data.Dataset object
63  vAR_char_dataset = tf.data.Dataset.from_tensor_slices(vAR_encoded_text)
64  # print first 5 characters
65  for char in vAR_char_dataset.take(8):
66      print(char.numpy(), vAR_int2char[char.numpy()])
67
68  # building sequences by batching
69  vAR_sequences = vAR_char_dataset.batch(2*vAR_sequence_length + 1, drop_remainder=True)
70
71  # printing sequences
72  for sequence in vAR_sequences.take(2):
73      print(''.join([vAR_int2char[i] for i in sequence.numpy()]))
```

STEP 4 - DATA PREPARATION (TRAINING & TEST DATA)

Step 4 is converting a single sample (sequence of characters) into multiple (input, target) samples. flat_map() method is exactly used for this purpose, it takes a callback function which loops over all our data samples.

Now the first data sample we are going to generate would be the following tuple of inputs and targets ('python is ', 'a'), the second is ('ython is a', ' '), the third is ('thon is a ', 'g')
and so on. We do that on all samples and in the end, we'll see that we have increased the number of training samples dramatically. We've used the ds.concatenate() method to add these samples together.

DeepSphere.AI
Enterprise AI and IIoT for Analytics

```
75  ##########################################################################
76  #       Step-4 Splitting the training dataset into input and Target      #
77  ##########################################################################
78
79  def split_sample(sample):
80      ds = tf.data.Dataset.from_tensors((sample[:vAR_sequence_length], sample[vAR_sequence_length]))
81      for i in range(1, (len(sample)-1) // 2):
82          input_ = sample[i: i+vAR_sequence_length]
83          target = sample[i+vAR_sequence_length]
84          # extend the dataset with these samples by concatenate() method
85          other_ds = tf.data.Dataset.from_tensors((input_, target))
86          ds = ds.concatenate(other_ds)
87      return ds
88
89  # prepare inputs and targets
90  dataset = vAR_sequences.flat_map(split_sample)
91
```