# **Report On Product Description Generator for Clothing Items**

**Objective**: To build an automated description generator for various clothing items, namely – T-shirts, jeans and kurtas (for this project only) based on various attributes provided of the apparel item, namely – brand, color and style.

**Pre - requisites**:

1. Recurrent Neural Network for Text Generation. (Built using Tensorflow)
2. 10400 descriptions per apparel item of various brands. (Obtained from Jabong.com) for training the neural network.

**Data Collected:**

**Site used for collecting descriptions: jabong.com**

**Web-scrapper used: jabongscrap.py**

|  |  |
| --- | --- |
| **Type of Apparel Item** | **Number of descriptions collected** |
| T-Shirt | 20484 |
| Jeans | 10493 |
| Kurta | 10150 |
| Shoes | 8450 |
| Watches | 5364 |

**Input Data Collection:**

The above table provides the information about the number of descriptions of products per apparel item. These were collected from the fashion e-commerce website [**jabong.com**](jabong.com)using a web-scrapper we designed ourselves names as [jabongscrapper.py](Product%20Description%20Generation/Input%20Generation%20Code/jabongscrap.py). The web-scrapper takes a link of the catalog webpage of each apparel item and downloads the desired number of descriptions (also taken as input) from the website.

In our case, the number of descriptions collected belong to the entire catalog of each apparel item available on the website.

The web-scrapper outputs a number of xml files containing the webpage url, image url, title and description of each product of an apparel item.

The title and description from these xml files are then stored in a single text file belonging to a particular apparel item. The text files are then used for training the recurrent neural network to generate a description for each apparel item.

**Process of Generation:**

Descriptions of t-shirts, jeans and kurtas were extracted from jabong.com using a web scrapper. It provided us with xml files of 10,400 products of each apparel item mentioned above. Using the xml files, input text files for the recurrent neural network were generated. The recurrent neural network was first trained separately on the descriptions per apparel item so as to enable it to generate a proper legible and understandable description. It was trained for 3000 epochs. After the training was complete, a prime word to start the description was provided to the neural network. The input to the program is a feature string containing the various features of the apparel item. The prime word is selected out of the feature string by recognizing the brand in the feature string using a text file containing the information of all brands. The prime word hence selected is provided to the RNN to start generating descriptions. Numerous descriptions are generated with the first description about the brand in the feature string. The other descriptions generated are partially random according to the context learnt from the input text files. Hence, to convert the context of the random descriptions, templates of the features (brand, color and style) were created so as to recognize the features in the random descriptions. After the features were found in the descriptions, these features were replaced by the features provided in the feature string. This finally provided us with descriptions about the feature string.

**Steps:**

1. Provide text file as input for RNN.
2. Train the RNN for 3000 epochs.
3. Provide the feature string containing info (brand, color and style) of apparel item.
4. Templates generated by collecting brands, colors and styles from input file.
5. Prime word selected from feature string by searching brand.
6. Multiple descriptions are generated according to generation length provided.
7. The following steps (8-9)are applied to each of the descriptions to output a number of final descriptions.
8. Templates are then used to replace the features in each description by the features in the feature string.
9. If certain templates are not present in the description, append features from feature string.

**Shortcomings of the text generator:**

1. The RNN used for the generator is designed to generate a sentence based on a single starting word hence the descriptions generated have a more or less meaningful context about the feature string containing multiple words i.e. features. The descriptions generated have a context accuracy of about 60-70%.
2. The generator may create some garbage words in the last sentence of the description.
3. Brands with more than one-word cause problems such as repetition and loss of semantics in the descriptions since, only the first word is replaced by brand template and not the rest of the words.

**Improvement Suggestions:**

1. The RNN can be redesigned to generate a sentence using multiple keywords from the feature string.
2. Templates have only been generated from brands, colors and styles of the apparel item. An in-depth study of the input text files can help in generation of templates consisting of other features apart from the above mentioned features, such as button designed, print design, style of collar, etc.
3. The multiple descriptions generated about a product using RNN can be summarized to produce a final meaningful description of the product.