

Automated Website Development

Tushar Gangurde¹, Shivaraya Patil¹, Vaibhav Patil¹, Akshya Mahale¹

¹Computer Science, Savitribai Phule Pune University / Dr DY Patil School of Engineering, Pune, Maharashtra, India.

ABSTRACT

A website helps a business to grow by using different marketing strategies. This paper describes a novel approach to develop a website by just providing the text (description of the website) or an image as input. Using Text Input it will suggest template (screenshots) after identifying the theme of the site inferred from the input. Those templates are converted into code for further customizations for their personal use. Current problem was that a web developer will take more than 15 days only to just make the basic structure of a website. This issue is resolved by our work which will generate the complete code of the webpage/ website in less amount of time. In this paper, it will tokenize each word to find their synonyms and then mapped it with root words for the theme identification and uses deep learning model to convert templates into code.

Keywords: Root words, Theme, Synonyms, code

I. INTRODUCTION

To survive in the digital world, website becomes the basic requirement of each business to represent itself in digitalized world, big or small. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored.

If a business does not have a website, they are losing the number of great opportunities for their business. A website helps a business to grow by using different marketing strategies. So, deploying a high end and user interactive websites are important and should be done in minimal time as possible.

What makes a website?

□ A website contains HTML code which acts as its Skeleton for website

□ Then comes CSS (Cascading Style Sheet) and JS (JavaScript) which is as the name says used for design the website

The user input can be in any structure of the sentence from which tokens will be extracted and their synonymies will be mapped to identify the theme from root word. After identifying the theme, Suggestions will be shown and that suggestion will be feed into Convolution neural network (CNN) after that features will be extracted and feed into Long Short Time Memory (LSTM) which will generate DSL Tokens. Those tokens will be compiled to generate code.

Paragraph Segmentation:

first step is to break the word in segments of words which is called tokens in NLP. Using Word tokenize package from NLTK. The tokens contain English words, every token is extracted from the sentence of words.

Word Analysis is now after the sentence is converted into tokens. Then part of speech tagging is done for each word. Words containing only Noun and adjective are extracted from the tokenized.

Only Noun and adjective are used to detect theme because:

□ Noun contains words like objects, actions qualities and state of existence

□ The adjective is useful in identifying parts of speech with the noun which is mainly the motive of word analysis. Word Analysis Now after the sentence is converted into tokens. Then part of speech tagging is done for each word. Words containing only Noun and adjective are extracted from the tokenized.

Only Noun and adjective are used to detect theme because:

□ Noun contains words like objects, actions qualities and state of existence

□ The adjective is useful in identifying parts of speech with the noun which is mainly the motive of word analysis.

Word Mapping and Template suggestion

Next Step is to map the words with their original word which is already present in the list to map each word we need to find synonyms for each word by using WORDNET which is a lexical database for English language, by using Wordnet.synset() function

to find the synonyms of the word.

After finding synonyms the words will be mapped with root words for theme after mapping. Templates are already classified which template comes in which theme on

the base of the identified theme. Templates will be suggested to the user.

CNN

In deep learning, to analyze visual images the mostw common class of neural networks used is convolutional neural network (CNN, or ConvNet). CNN require minimal preprocessing because it is designed by using a variation of multilayer perceptron.[1] Due to shared-weights architecture and characteristics of translation invariance they are known as shift invariant or space invariant artificial neural networks (SIANN). Inspiration from biological processes [4], CNN develops a network of neurons by proving the connectivity pattern in the way that it looks like the pattern used in animal visual cortex. In receptive field, each individual cortical neuron will respond only in restricted region to the stimuli of the visual field. Different neurons show partially overlapping in the receptive fields in such a way that they will cover the entirevisual field.

LSTM

Recurrent neural networks (RNN) have the units as long short-term memory (LSTM). LSTM network consist of various LSTM units to build a recurrent neural network. Various components of LSTM unit are cell, a forget date, an input and output gate. All the 3 gates control the transmission of information in and out of the cell and then the cell stores the values on arbitrary intervals of time. To classify the data, process it and perform predictions which are on the basis of time series data, most commonly networks

used are LSTM networks because in important time series events there can be uneven or unknown duration lags. While training the traditional recurrent neural networks various vanishing and exploding gradient problems occur and LSTMs are developed to handle such problems. In various applications, relative insensitivity to gap length makes LSTM better than recurrent neural networks, hidden Markov models and other sequence learning methods.

II. Our Approach

The proposed system is done in various steps like text segmentation, tokenizing, part of speech tagging, word map and theme, suggesting, CNN, LSTM and decoder as shown in Figure 1.

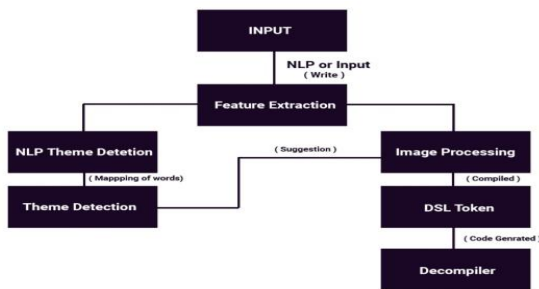


Figure 1: Steps of approach

2.1 Input Image and Text

In this system the user can input text in two ways:

- User can upload images of UI to convert UI to code.
- User can type text description which will be used to suggestion and from that user can select the type of UI they went from the suggestion and that will generate code.



Figure 2: User Input

2.2 Feature Extraction

Both the methods of input have feature extraction Algorithms which extracts characteristics from both inputs each of them has their own feature extraction algorithm.

2.2.1 Image input uses CNN as a feature extraction algorithm to get characteristics of input. CNN is widely used in computer vision problems because of its topology which allows them to extract minor details from the input. We used convolutional neural network which behaves like an encoder and performs unsupervised learning by comparing an input image to a fixed-length vector which was already made learn to system.

2.2.2 Text input uses NLTK python library to firstly tokenize the inputs from which Noun and Adjectives are extracted from the tokenized data.

EXAMPLE:

INPUT: "Need a website with red color navigation panel and black background"

OUTPUT:

[, 'Need', 'website', 'with', 'red', 'colour', 'navigation', 'panel', 'black', 'background']

2.3 Theme Detection

After Tokenizing these phases come into play which Entity Recognition or Chunking which extracts Noun and Adjectives from tokenized data which are called Chunked data from that data synonyms are extracted and mapped with root words.

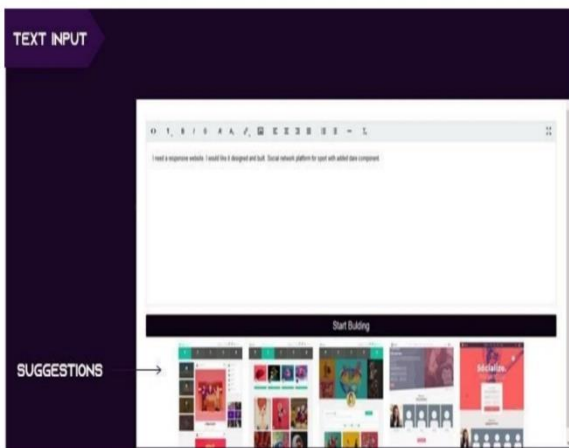


Figure 3: Theme Detection from text Input

2.4 Image Processing and DSL Token generation:

After the extraction of features from the image using CNN, we used DSL tokens to describe UI Components. To find different graphical components and their relation between each other DSL Token generation is used. DSL reduces the size of search space by reducing the total number of tokens of vocabulary of the DSL. By providing a discrete input, our system language model performs the modeling at token-level which

uses one-hot encoded vectors, eliminating the need for word embedding techniques such as word2vec.

2.5 Decoder

By using supervised learning method, model is trained by inputting an image I and x_t is a contextual sequence of X of T tokens, $t \in \{0 \dots T - 1\}$ as inputs; and x_T token is taken as the target label. Input image I is encoded into vector representation p by using CNN-based vision model. LSTM Model is used to encode the input token x_t into an intermediate representation q_t which allows the model to concentrate more on certain type of tokens and less focus on others.

Each LSTM layer consists of 128 cells and the first language model consist of two such layers. A single feature vector r_t is formed by the concatenation of p(vision encoded vector) and q_t (language encoded vector) which is then given as input to a second LSTM-based model. This model will decode the image by mapping with various models like vision and language model. This makes the decoder able to learn to map the relation between the objects identified in GUI image provided as input and the tokens present in DSL code. In our decoder each LSTM layer consists of 512 cells and it is implemented as the combination of two LSTM layers.

The architecture discussed above can be represented in

mathematical form as:

$$p = \text{CNN}(I)$$

$$q_t = \text{LSTM}(x_t)$$

$$r_t = (q, p_t)$$

$$y_t = \text{softmax}(\text{LSTM}_0(r_t))$$

$$x_{t+1} = y_t$$

III. Result

This work will generate code for UI input provided by the user. Table 1 shows input provided by user as an image and code is the output generated by the system. Table 2 shows input provided by user as text and themes are generated by the system as output.



Figure 4: Input 1

```
START <!DOCTYPE html>
<html lang="en" dir="ltr">
<head>
<title>Basic 88</title>
<meta charset="iso-8859-1">
<link rel="stylesheet" href="styles/layout.css" type="text/css">
<!--[if lt IE 9]><script src="scripts/html5shiv.js"></script><![endif]-->
</head>
<body>
<div class="wrapper row1">
<header id="header" class="clear">
<div id="hgroup">
<h1><a href="#">Basic 88</a></h1>
<h2>Free HTML5 Website Template</h2>
</div>
<nav>
<ul>
<li><a href="#">Text Link</a></li>
<li><a href="#">Text Link</a></li>
<li><a href="#">Text Link</a></li>
<li><a href="#">Text Link</a></li>
<li class="last"><a href="#">Text Link</a></li>
</ul>
</nav>
</header>
</div>
</body>
</html>
```

Figure 5: Output-1

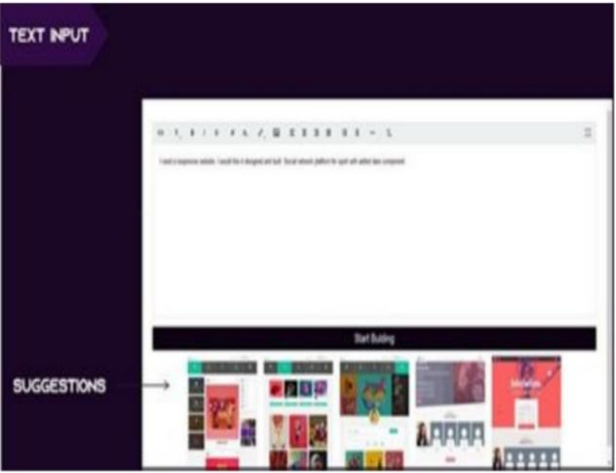


Figure 6: Input-2

```
<body>
<div class="wrapper row2">
<!-- main content -->
<div id="homepage">
<!-- Services -->
<section id="services" class="clear">
<article class="one_third">
<figure>
<figcaption>
<h2>Indonectetus facilis</h2>
<p>Nullamlacus dui ipsum consequ loborttis
non eisque morbi penas dapibulum orna.</p>
<footer class="more"><a href="#">Read More
&raquo;</a></footer>
</figcaption>
</figure>
</article>
<article class="one_third">
<figure>
<figcaption>
<h2>Indonectetus facilis</h2>
<p>Nullamlacus dui ipsum consequ loborttis
non eisque morbi penas dapibulum orna.</p>
<footer class="more"><a href="#">Read More
&raquo;</a></footer>
</figcaption>
</figure>
</article>
</section>
</body>
</html>
```

Figure 7: Output-2

TABLE 2: Text Input and OUTPUT.

INPUT
I need a responsive website. I would like it designed and built. Social network platform for sport with added dare component
OUTPUT


IV. CONCLUSION

This proposed system have discussed the theme detection technique and suggesting themes to user and generating code from the selected template. Current problem was that a web developer will take more than 15 days only to just make the basic structure of a website. This issue is resolved by our work which will generate the complete code of the webpage/ website in less amount of time. This system can be used by anyone to make a website from just a text input or screenshot of a website to generate code from it. It can be used to generate code for Android or

IOS Applications from there GUI Screenshots to fully functional code. Our model is trained on very small dataset increasing the size dataset can lead to more accurate results. Our work is concerned with the generation of static website code. This work can be further extended to developing or generating the code for both static as well as dynamic websites.

V. REFERENCES

- [1]. J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2625–2634.
- [2]. F. A. Gers, J. Schmidhuber, and F. Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural Computation*, 12(10), 451–247.
- [3]. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.
- [4]. A. Karpathy and L. Fei-Fei. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.
- [5]. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.
- [6]. W. Zaremba, I. Sutskever, and O. Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

- [7]. Gurpreet Kaur, Prateek Agrawal. 2016. "Optimisation of Image Fusion using Feature Matching Based on SIFT and RANSAC", Indian Journal of Science and Technology, 9(47), pp 1-7.
- [8]. H. Zhang, T. Xu, H. Li, S. Zhang, X. Huang, X. Wang, and D. Metaxas. Stackgan. 2016. Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv preprint arXiv:1612.03242.
- [9]. A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105.
- [10]. D. Bahdanau, K. Cho, and Y. Bengio. 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
- [11]. A. L. Gaunt, M. Brockschmidt, R. Singh, N. Kushman, P. Kohli, J. Taylor, and D. Tarlow. 2016. Terpret: A probabilistic programming language for program induction. arXiv preprint arXiv:1608.04428.
- [12]. W. Ling, E. Grefenstette, K. M. Hermann, T. Kociský, A. Senior, F. Wang, and P. Blunsom. 2016. Latent predictor networks for code generation. arXiv preprint arXiv:1603.06744.
- [13]. L. Yu, W. Zhang, J. Wang, and Y. Yu. 2016. Seqgan: sequence generative adversarial nets with policy gradient. arXiv preprint arXiv:1609.05473.
- [14]. S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. 2016. Generative adversarial text to image synthesis. In Proceedings of The 33rd International Conference on Machine Learning, volume 3.
- [15]. K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In ICML, volume 14, pages 77–81. View publication stats.