LOW LEVEL DESIGN AND IMPLEMENTATION DOCUMENT

STYLE-CONSISTENT KANNADA FONT GENERATION

UE17CS490B – Capstone Project Phase – 2

Submitted by:

Saahil B Jain **PES1201700241**
Jeevana R Hegde **PES1201700633**

Under the guidance of

Prof. K.S. Srinivas
Associate Professor
PES University

January - May 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

LOW LEVEL DESIGN AND IMPLEMENTATION DOCUMENT

TABLE OF CONTENTS

1. Introduction 3
   - 1.1 Overview 3
   - 1.2 Purpose 3
   - 1.3 Scope 4
2. Proposed Methodology / Approach 5
   - 2.1 Algorithm and Pseudocode 6
   - 2.2 Implementation and Results 7
   - 2.3 Further Exploration Plans and Timelines 9

Appendix A: Definitions, Acronyms and Abbreviations 9

Appendix B: References 9

Appendix C: Record of Change History 9

Appendix D: Traceability Matrix 10

1. Introduction

1.1. Overview

Indian languages currently have a limited set of fonts available and Generation of new fonts is a manual process where designers use tools to draw letters in different ways. No current work exists in automating the generation of fonts for Indian languages like Kannada. This brings the need for our project which aims to automate the generation of new fonts for the Kannada scripts while simultaneously ensuring generation of maatras.

Currently fonts are generated manually by designers. These designers use software drawing tools to draw and create each letter of the alphabet set. Not only is this process hard because the designers have to ensure style consistency among every letter, but it is time consuming as well. Recent research has attempted to generate fonts for the English language, however they require a large amount of pre-existing fonts to start generating new fonts. This amount of data doesn't exist for Indian languages like Kannada, which have only about 4-5 available fonts.

Hence a different approach is required to generate fonts for Indian languages so that fonts can be generated without the need for pre-existing fonts. This is where our solution comes into picture where we can generate new style consistent fonts without the need for pre-existing fonts.

Our project could finally be used by a website which would offer users multiple auto- generated fonts to choose from for the Kannada script. Each option would be available separately where users can browse through all options and choose and download the ones they like most.

1.2. Purpose

We notice that logo fonts can make or break your logo design. Eg: Disney, Netflix, The Times of India, etc. With continuous advancements in web-technologies and the need to deliver a great experience to non-English speaking users is creating a greater demand for high-quality, multi-script typefaces

We particularly loo at use cases of this problem statement:

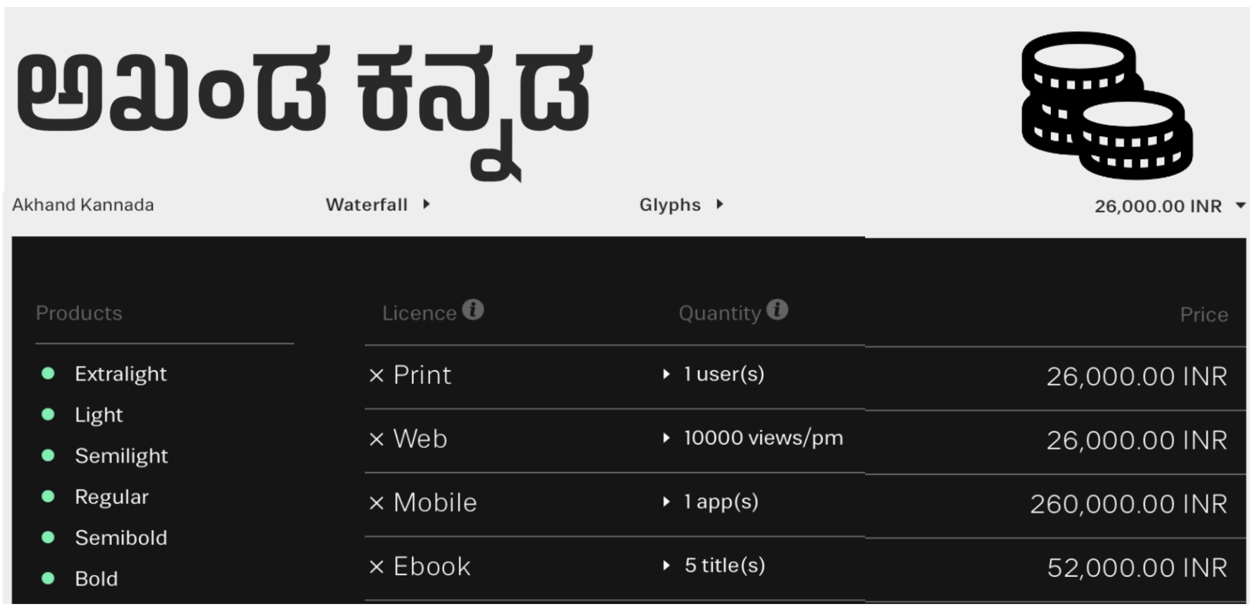
1. ***Television channels*** - A crisp, bold, and clear font to depict the name of the channel with the logo on the right top corner.
2. ***Newspapers, Magazines and Books*** - A large heading on the front cover page and consecutively on Dept. of CSE Jan - May 2020 Page 9 each page to mark the name of newspaper. Also includes a consistent font style of the entire text or content.
3. ***Movies*** - An eye catchy font to grasp public attention with a single glance.

4. ***Company name font designers*** - Generic Kannada brand names to attract local audience.

1.3. Scope

- Indian Type Foundry (ITF) is a company which ***manually*** creates customized fonts for various languages.
- The charges range from ***26,000INR - 2,60,000INR*** for a specific typeface.

  Design Constraints, Assumptions, and Dependencies Design Description



- Currently font generation is done where the designers must manually draw and further trace Glyphs (characters) using software tools such as Font Developer.
- For each font, the time varies from a couple of hours to a couple of days.
- Our project aims to automate this process while simultaneously trying to incorporate style inspired from fonts available for other languages (Style transfer).
- Hence makes the process less time-consuming, more efficient (using style consistency) and provides a varied set of options to choose from at a cheaper price.

2. Proposed Methodology / Approach

We have currently chosen two approaches to obtain the style transfer from English language.

1. Neural style transfer

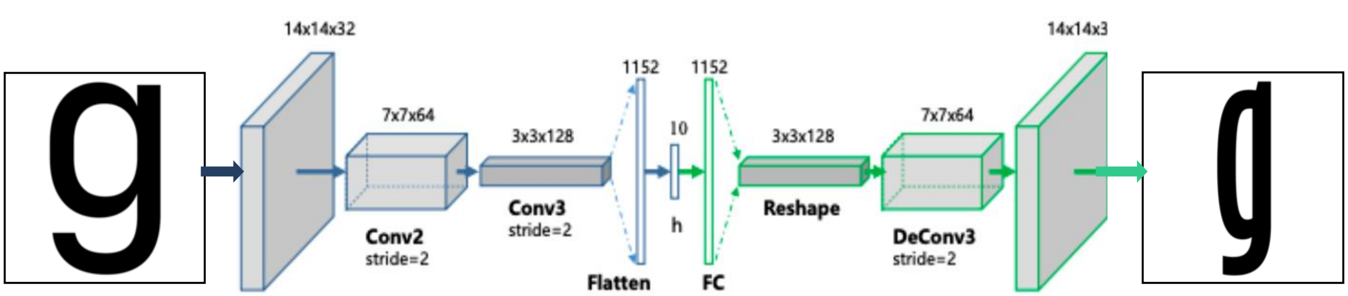
a. Model architecture

b. Basic approach

The inputs for the model three images. These include the style image, target character and the generated image. The three inputs are passed to the Neural network iteratively and the output is obtained. The loss which is calculated is back propagated. And the output of each iteration is passed as input to the next iteration. Hence, improving the image each time. The process is repeated until the generated image is satisfactory.

2. Autoencoders

a. Model architecture



b. ***Basic approach***

The idea is to train an autoencoder which takes a character in basic font as input and converts it into the required font. As compared to the previous approach, three inputs aren't taken and simply a single input is given. Currently, English letters of Screen Sans style(for example) are being taken as input images and the letters of the required font are taken as output images. Based on this the autoencoder is trained.

Later when Kannada letters are provided as inputs, the outputs obtained are expected to have the desired style. Hence style transfer has been obtained.

2.1 Algorithm and Pseudocode

Neural style transfer

The loss function, being an essential part of neural style transfer, is broken into two parts- Content loss and style loss.

Content loss refers to the difference in the structure from the input image.

Whereas style loss refers to the difference in style between the output and style image.

Content loss:

$$J_{\text{content}}(C,G)=\frac{1}{2}\left\|\underbrace{a^{[\ell](C)}}_{\text{Original/Content}}-\underbrace{a^{[\ell](G)}}_{\text{Generated}}\right\|^2$$

Style loss:

$$G_{ij}^{l}=\sum_{k} g_{ik}^{l} g_{jk}^{l}$$

$$S_{ij}^{l}=\sum_{k} s_{ik}^{l} s_{jk}^{l}$$

Gram Matrix for Style and Generated Image

$$\mathcal{L}_{\text{style}}(g,s)=\sum_{i,j}\left(G_{ij}^{l}-S_{ij}^{l}\right)^2$$

Total loss:

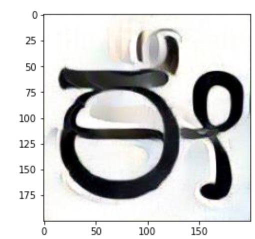
$$\mathcal{L}_{\text{total}}(G) = \alpha \mathcal{L}_{\text{content}}(C, G) + \beta \mathcal{L}_{\text{style}}(S, G)$$

Here, alpha and beta are coefficients that are used to build some sort of ratio.

2.2 Implementation and Results

Neural style transfer

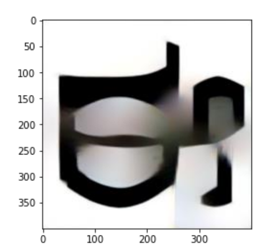
Style 1



Result 1

Input image

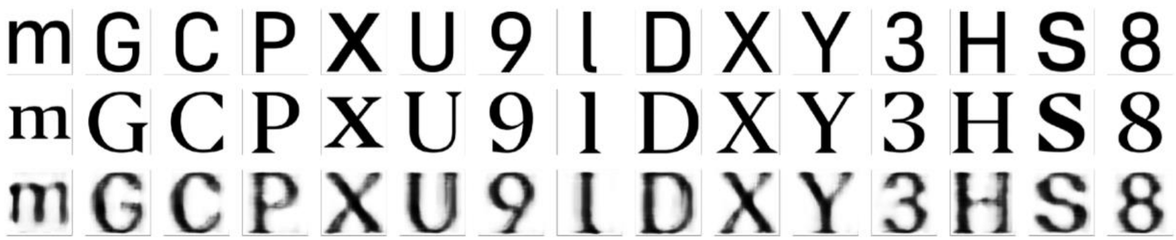
Style 2



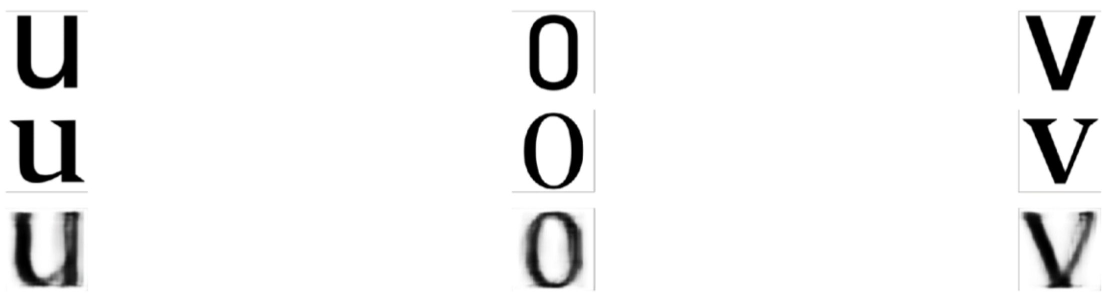
Result 2

Autoencoders

Train



Test



Kannada letters and digits

2.3 Further Exploration Plans and Timelines

Although the approaches are providing us with results, we intend to improvise it further to obtain more satisfactory results. To do so, we plan on creating our own loss function and training the model on the same. This fine tuning process might take us about 3 weeks. Prior to this, we want to extent data and this is expected to be finished in the following week.

Once the fine tuning has been completed, we will strictly focus on the documentation for the rest of the time.

Appendix A: Definitions, Acronyms and Abbreviations

Autoencoders: An autoencoder is a type of artificial neural network used to learn efficient data coding in an unsupervised manner. The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction, by training the network to ignore signal "noise"

Appendix B: References

Research paper

[1] Samaneh Azadi, Matthew Fisher, Vladimir Kim, Zhaowen Wang, Eli Shechtman, Trevor Darrell, "Multi-Content GAN for Few-Shot Font Style Transfer", Dec. 01, 2017. [Online]

Available: https://arxiv.org/pdf/2006.06676.pdf [Accessed Jan. 16,2021]

Research paper

[2] Yonggyu Park, Junhyun Lee, Yookyung Koh, Inyeop Lee, Jinhyuk Lee, Jaewoo Kang, "Typeface Completion with Generative Adversarial Networks", Dec. 13, 2018. [Online],

Available: https://arxiv.org/pdf/1811.03762.pdf [Accessed Jan. 05, 2020]

Research paper

[3] Yizhi Wang, Peking University, "Attribute2Font", May. 16, 2020. [Online],

Available: https://arxiv.org/pdf/2005.07865v1.pdf [Accessed Jan. 22, 2021]

Research paper

[4] Leon A. Gatys, Alexander S. Ecker, Matthias Bathge, "A Neural Algorithm of Artistic Style", Sep. 02, 2015. [Online],

Available: https://arxiv.org/pdf/1508.06576.pdf [Accessed Feb. 01, 2021]

Appendix C: Record of Change History

[This section describes the details of changes that have resulted in the current Low-Level Design document.]

#	Date	Document Version No.	Change Description	Reason for Change
1.	18.02.2021	1	Primary completion	Primary completion
2.				
3.				

Appendix D: Traceability Matrix