International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 03 | Mar 2025 www.irjet.net p-ISSN: 2395-0072

Automated Font Recommendation System Using Deep Learning

Asma Begum[1], Dayala Akshitha[2], Nalla Bharat Kumar[3], Ms. V. Alekya[4]

[1]B-Tech 4th year, Dept. of CSE (DS), Institute of Aeronautical Engineering
[2] B-Tech 4th year, Dept. of CSE (DS), Institute of Aeronautical Engineering
[3] B-Tech 4th year, Dept. of CSE (DS), Institute of Aeronautical Engineering
[4]Assistant Professor, Dept. of CSE (DS), Institute of Aeronautical Engineering, Telangana, India

---

Abstract - *This project introduces an automated system that helps users choose the best fonts for different types of documents, such as academic papers, business reports, legal documents, and creative works. The system uses a machine learning model called DenseNet-201 to suggest fonts based on the user's needs and preferences. To develop this system, a collection of Google Fonts was organized into five categories: Sans-serif, Serif, Handwriting, Display/Decorative, and Mono-spaced. These fonts were then transformed into grayscale images, each 300x300 pixels, for training the DenseNet-201 model. After the model is trained, it analyzes input documents by extracting and classifying text, which helps it suggest the most suitable font style, size, and weight to improve readability and appearance. The system was tested with a dataset of fonts and proved to be effective at accurately recognizing different font styles and providing helpful recommendations for various document types. By automating the font selection and document analysis process, this system offers users a useful tool for improving their documents' visual quality and professionalism.*

Key Words: Font recommendation, DenseNet-201, Google Fonts, image conversion, text extraction, classification, automated system.

1. INTRODUCTION

In the digital age, font selection plays a critical role in enhancing the readability, aesthetic appeal, and professionalism of documents. Whether for academic, business, legal, or creative purposes, choosing the right font style and size can significantly influence how a document is perceived. Despite the availability of numerous font options, users often face challenges in selecting the most appropriate font for their specific needs. This research addresses this challenge by developing an automated font recommendation system that leverages machine learning to assist users in selecting optimal fonts for diverse document types.

To build the system, a comprehensive dataset of font images was manually collected from Google Fonts using the Google Fonts API. The API allowed us to programmatically access font metadata and download URLs, facilitating the creation of a diverse dataset. Fonts were categorized into six main styles: Serif, Sans-serif, Script, Display, Monospaced, and Handwritten, representing a wide variety of fonts commonly used in document.

Understanding the Font Selection:

Font Categories and Styles: Fonts are grouped into several categories, each serving distinct purposes based on the nature of the document. Serif fonts, for instance, are commonly used in formal documents such as academic and legal texts due to their traditional appearance, while Sans-serif fonts are preferred for modern and digital content because of their clean and minimal design. Understanding these categories—Serif, Sans-serif, Script, Display, Monospaced, and Handwritten—helps in selecting a font that complements the tone and purpose of the document.

Readability and Legibility: The primary goal of font selection is to ensure the text is easy to read. Fonts with clear and distinguishable characters enhance legibility, especially for long-form documents. Factors such as font size, weight, and spacing also play a significant role in improving readability. For instance, Sans-serif fonts are often used in digital documents due to their high legibility on screens, while Serif fonts are better suited for printed text, where they guide the reader's eye across lines.

Document Context and Audience: The context of the document and its intended audience are critical in font selection. Business reports, for example, demand professional, clean fonts like Sans-serif, whereas creative projects may benefit from more expressive fonts like Handwritten or Display. Understanding the preferences and expectations of the target audience helps in selecting a font that enhances the document's message and visual appeal without distracting from its content.

A custom function was developed to automate the font download process, retrieving font files from the API and saving them for further analysis. In addition, another function was created to generate character images using the downloaded fonts, providing visual representations of how each font renders different characters. These images were crucial for training the DenseNet-201 model used in the recommendation system. By converting the fonts into 300x300 pixel grayscale images, the system was able to classify font styles and provide tailored font recommendations for various document types.

This paper presents the process of dataset collection, image generation, and the subsequent application of machine

learning to recommend fonts. The aim is to streamline font selection and improve document presentation, making the system a valuable tool for users who seek to enhance the visual quality of their written materials.

The system operates in two key phases. First, it trains on a comprehensive dataset of font images, learning to recognize and classify different font styles. Second, it processes input documents by extracting and analyzing text, enabling the system to provide tailored font recommendations based on the document's content and purpose. This dual approach not only ensures accurate font recognition but also aligns the recommended fonts with the specific requirements of the document type.

The resulting automated system offers a practical solution for users seeking to enhance the visual quality and professional appearance of their documents. By streamlining the font selection process and ensuring that fonts are chosen with consideration of both aesthetic appeal and functionality, the system has the potential to significantly improve the overall document creation experience. The rest of the paper is organized as follows. Section II summarizes the related literature. Section III introduces the proposed anomaly detection approach. Section IV details the experiments and presents the evaluation results, while Section V further discusses the interpretation of the results and makes comparisons. Finally, the conclusion and future research directions are presented in Section VI.

2. LITERATURE SURVEY

The selection of appropriate fonts is crucial for enhancing the readability and appearance of various documents. Over time, researchers have developed different techniques for font recognition, generation, and recommendation using machine learning and neural networks. The growing availability of large font datasets and advancements in deep learning have made it possible to develop sophisticated systems that automatically suggest suitable fonts based on text characteristics and user preferences.

[1] Spohr, Hollink, and Cimiano (2011) explored the use of machine learning for ontology matching across multiple languages, emphasizing the importance of matching data based on specific linguistic characteristics. Although their focus was on language processing, their methods provide insight into how font classification across various styles and languages can be approached in a similar way, contributing to the development of automated font recommendation systems.

[2] Rapee and Igarashi (2010) introduced an example-based method for automatic font generation, where new fonts were created by learning from existing examples. This idea parallels the concept of training deep learning models on existing fonts, as it allows machines to understand the characteristics of fonts and generate recommendations accordingly.

[3] Kim and Byun (2007) focused on optimizing font sets for multilingual input systems, highlighting the importance of font recognition in diverse language contexts. Their work is particularly relevant to font recommendation systems, as fonts must be selected based on language-specific requirements in addition to aesthetic preferences.

[4] Chen et al. (2014) presented a large-scale visual font recognition system using deep learning techniques. Their approach involved the classification of fonts based on visual features extracted from font images. This work demonstrates how machine learning models, such as DenseNet-201, can effectively classify fonts by learning from visual data. The use of grayscale images of fonts in the current research draws inspiration from this method, as it allows the model to recognize font styles with high accuracy.

[5] O'Donovan et al. (2014) explored font selection based on crowdsourced attributes, allowing users to select fonts based on the perceived personality of each font. This work underlines the subjective nature of font choices and how user preferences can be incorporated into automated systems. The current research integrates user preferences as part of the font recommendation process, aiming to deliver fonts that are both aesthetically pleasing and functionally appropriate.

3. METHODOLOGY

Automated font recommendation system using deep learning. It starts with Font Dataset Collection and Preprocessing, followed by Feature Engineering using data augmentation. The DenseNet Model Training phase trains the model on the prepared data, and Model Testing and Validation ensures its accuracy. The system then provides Font Recommendations based on user input, and its effectiveness is assessed in the Performance Evaluation step.

3.1 Proposed Work

For this project, DenseNet-201 was selected due to its ability to efficiently handle image classification tasks by capturing intricate patterns and features in the input data. DenseNet's architecture, which promotes feature reuse through densely connected layers, makes it particularly suitable for recognizing font characteristics from image data. Its deep learning capabilities allow for accurate classification and recommendation of fonts by learning complex patterns across various font styles, making it an ideal choice for this automated font recommendation system.

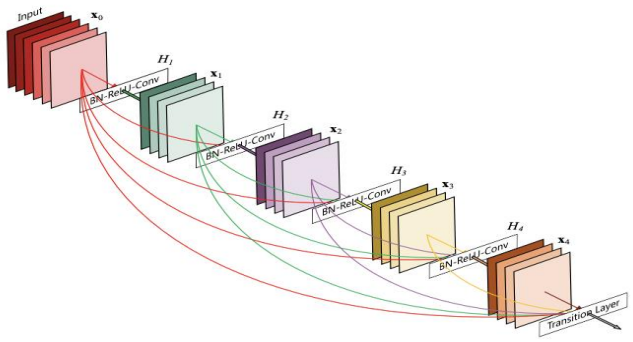

Fig -1: An example of a Dense Net Convolution Network

3.2 System Architecture

The model training process in this system uses DenseNet201, a deep learning architecture known for its ability to handle image classification tasks. Font images are preprocessed into 300x300 grayscale images and undergo augmentation techniques such as rotation, zooming, and flipping to create a diverse training set. DenseNet-201 is trained on this enhanced dataset to learn various font patterns and styles. This training enables the model to provide font recommendations that match the user's preferences and the document's context.

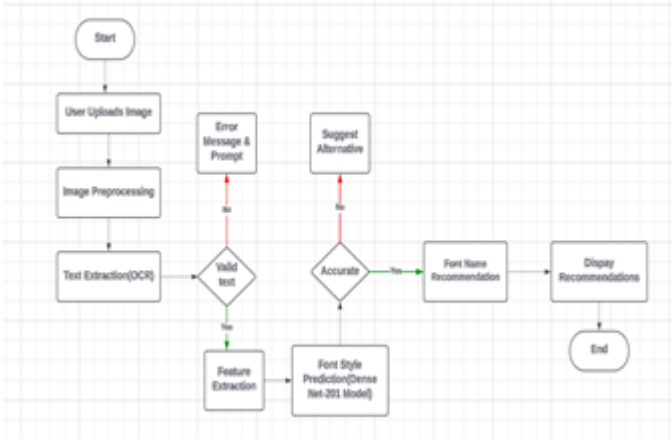

Fig -2: Process of feature selection and extraction

3.3 Dataset

The dataset for the automated font recommendation system was manually collected from Google Fonts using their API. It includes various font categories such as Serif, Sans-serif, Script, Display, Monospaced, and Handwritten. Each font is represented by images of characters rendered in a standardized format to ensure consistency.

```
from google.colab import files

# List of files to download
files_to_download = [
    'Libre Baskerville_x.png',
    'Libre Baskerville_X.png',
    'Libre Baskerville_y.png',
    'Libre Baskerville_Y.png',
    'Libre Baskerville_z.png',
    'Libre Baskerville_Z.png',
]

for file in files_to_download:
    if os.path.exists(file):
        files.download(file)
    else:
        print(f"File not found: {file}") # Print a message if t
```

Fig -3: Code to download specific character image files all in equal size

This diverse dataset provides a comprehensive foundation for training the model to recommend fonts based on different styles and use cases.

Fig -4: These are the final views of dataset collection

3.4 Data Processing

Data preprocessing began with cleaning the raw font files, ensuring they were properly downloaded and in the correct format. Any corrupted files were removed or replaced. We then standardized the font images by rendering characters at a consistent size and resolution to ensure uniformity across different styles. Finally, the images were normalized to

grayscale, removing any noise or color inconsistencies. The dataset was then split into training, validation, and test sets, preparing it for accurate model evaluation in future tasks.

3.5 Feature Selection

The implementation of a deep learning model using the DenseNet201 architecture for image classification tasks. Initially, the model is set up by loading the pre-trained DenseNet201 with weights from ImageNet, excluding the top classification layer, and specifying an input shape suited for RGB images. To adapt the model for a custom task, global average pooling is applied to reduce the dimensionality of the features extracted by the Dense Net base model. Following this, a dense layer with 1024 units and RLU activation is added to introduce a fully connected layer. The final output layer uses a SoftMax activation function to classify the input into one of several categories, depending on the number of classes.

The code also demonstrates a common technique in transfer learning: freezing and unfreezing layers. In the initial phase of training, all layers of the Dense Net base model are frozen to retain the pre-trained features and focus on training the newly added layers. After this, fine-tuning is performed by unfreezing the last 40 layers of the base model, allowing these layers to be updated during training with a reduced learning rate. This gradual fine-tuning helps the model learn task-specific features without overfitting or losing the generalization power acquired from pre-trained weights.

1. Loading the Pre-trained Model: The DenseNet architecture is initialized with pre-trained weights derived from ImageNet (weights='imagenet') while omitting the fully connected top layers (include top=False). The input shape for the architecture is defined through the input shape parameter, presented as (img height, img width, 3), representing RGB images with sizes indicated by img height and img width.

2. Adding Custom Layers: The output from the base architecture is linked to a GlobalAveragePooling2D layer, which compresses the spatial dimensions of the feature maps into a single vector for each channel. A dense (fully connected) layer containing 1024 units with ReLU activation is incorporated to capture more intricate features. The concluding layer consists of a dense layer with num classes units and softmax activation for the purpose of multi-class classification.

3. Freezing the Base Model: At first, the layers within the base architecture are frozen (layer. trainable = False), preventing their weights from being modified during the training phase. This approach is typical when utilizing the pre-trained architecture's features without alterations.

4. Compiling the Model: The architecture is compiled utilizing the Adam optimizer, categorical cross-entropy loss (appropriate for multi-class classification), and accuracy as the metric for evaluation.

5. Fine-tuning the Model: Certain layers within the base architecture are unfrozen to allow for fine-tuning, specifically the last 40 layers in this instance (base model. Layers [-40:]). The trainable attribute for these layers is set to True, allowing them to be modified during the training process.

6. Recompiling with a Lower Learning Rate: Subsequent to unfreezing the layers, the architecture is recompiled with a reduced learning rate (learning rate=0. 0001) to make sure that the fine-tuning procedure does not result in significant weight changes, which may interfere with the features learned by the pre-trained architecture.

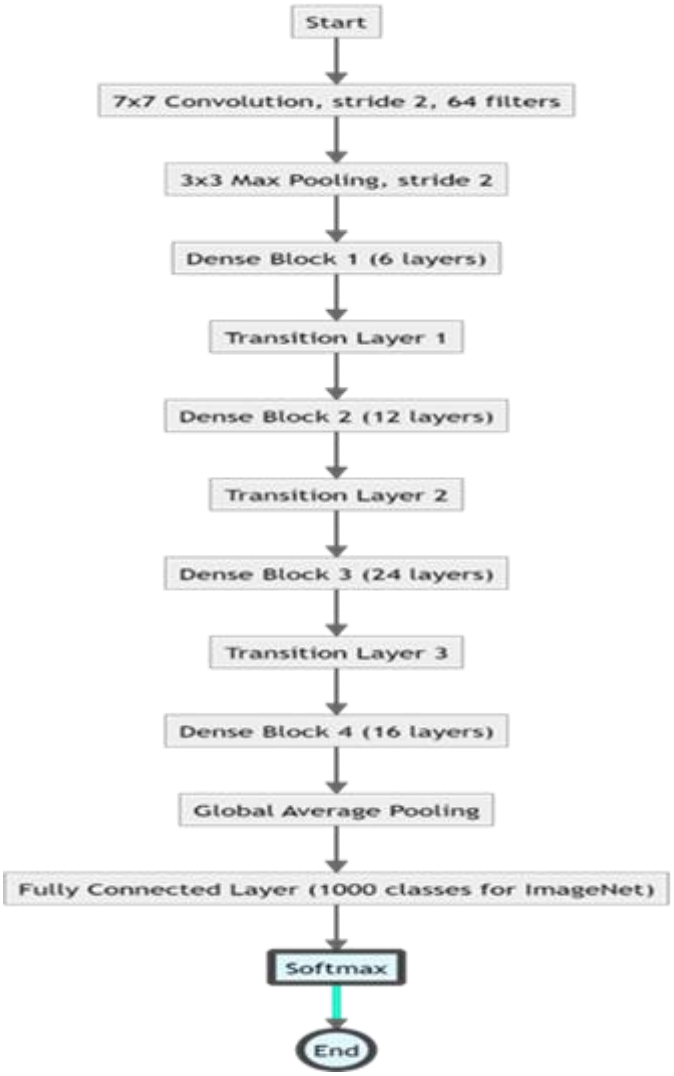

Fig -5: Process for feature extraction

3.6 Training & Testing

The training dataset for the automated font recommendation system consists of carefully curated font images sourced

from Google Fonts. This dataset includes a wide variety of font styles, encompassing categories such as Serif, Sans-serif,

Script, Display, Monospaced, and Handwritten. Each font is represented by images of multiple characters, allowing the model to learn the unique features and characteristics of different typefaces. To ensure effective training, the dataset was preprocessed to standardize image size and format, facilitating consistent input for the deep learning model. This diverse and well-structured training dataset enables the Dense Net model to accurately capture font distinctions, improving its ability to recommend suitable fonts based on user input. The inclusion of various styles and designs further enhances the model's robustness and adaptability in real-world applications.

The testing dataset for the automated font recommendation system was specifically created to evaluate the model's performance on unseen data. It consists of a distinct set of font images that were not included in the training phase, ensuring that the model's recommendations are assessed fairly. Similar to the training dataset, this testing set encompasses a variety of font categories, including Serif, Sans-serif, Script, Display, Monospaced, and Handwritten.

Each font in the testing dataset is represented by images of multiple characters, allowing for a comprehensive evaluation of the model's ability to recognize and recommend fonts accurately. By using this separate dataset, we can gauge the effectiveness of the Dense Net model in real-world scenarios, measuring metrics such as accuracy, precision, and recall. This approach helps ensure that the model generalizes well beyond the training data and can effectively recommend fonts based on a wide range of inputs.

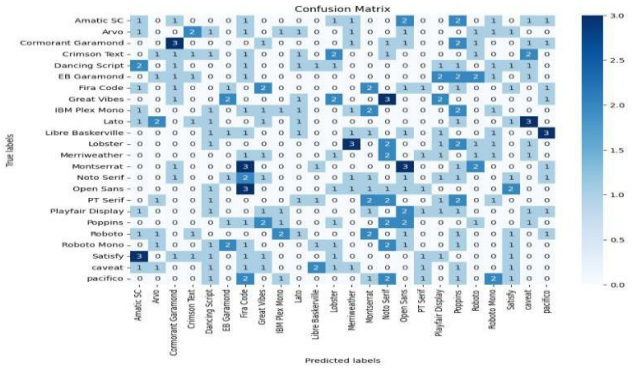

Fig -6: Confidence Matrix

4. EXPERIMENT RESULTS

The graphs display the model's training and validation performance across epochs in terms of accuracy and loss. In the accuracy plot, the training accuracy consistently improves, reaching approximately 95% by the end of the training. However, the validation accuracy remains relatively lower, fluctuating around 65%, indicating that the model might be overfitting to the training data, as it generalizes poorly on the validation set.

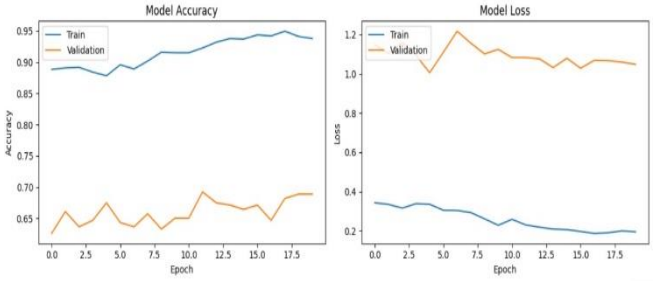

Fig -7: Model accuracy and model loss

In the loss plot, the training loss decreases steadily, suggesting that the model is learning effectively. On the other hand, the validation loss increases after a few epochs, a further indication of overfitting. The widening gap between training and validation performance suggests that while the model performs well on training data, it struggles to maintain the same level of performance on unseen data, which calls for adjustments like regularization or early stopping. The image shows the result from an Automated Font Recommendation System, which is designed to identify and suggest appropriate fonts for various document types based on an image of a font provided by the user. In this case, the system has predicted the font in the image to be "PT Serif" with a high confidence of 99.14%.

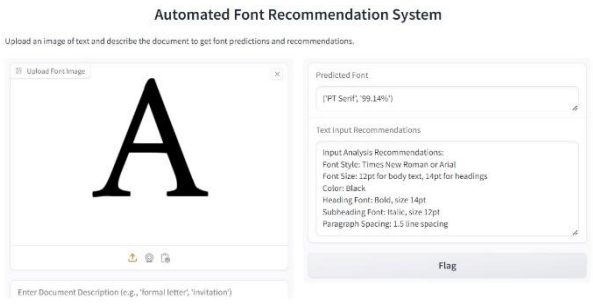

Fig -8: Automated Font Identification and Text Formatting Recommendations

Additionally, the system provides text input recommendations, suggesting that for documents like formal letters or invitations, the preferred font styles could be Times New Roman or Arial. The recommended font size is 12pt for body text and 14pt for headings. Other suggestions include using black color for text, bold font for headings (size

14pt), italic font for subheadings (size 12pt), and 1.5 line spacing for paragraph formatting. These recommendations aim to ensure clarity and professionalism in formal documents.

5. CONCLUSIONS

In conclusion, the automated font recommendation system developed using DenseNet-201 demonstrates significant potential in enhancing the process of font selection for various types of documents. By leveraging advanced deep learning techniques, the system effectively analyzes input text and user preferences, providing tailored font suggestions that improve both readability and aesthetic appeal. The implementation of a robust training process, along with thorough evaluation and validation, ensures that the model performs reliably across different font categories. This project not only streamlines the font selection process but also contributes to the broader field of automated design tools. Looking ahead, there are several avenues for further development of this project. Enhancing the model with additional data sources, including user feedback and real-time font usage patterns, could lead to more personalized recommendations.

Additionally, exploring the integration of user interface elements, such as a web-based application or plugin for design software, would make the tool more accessible to a wider audience. Investigating the potential for multilingual font recommendations could also broaden the system's applicability across different languages and cultural contexts. Finally, incorporating advanced user profiling and machine learning techniques, such as reinforcement learning, could continuously refine the recommendation process based on user interactions and preferences.

REFERENCES

[1] D. Spohr, L. Hollink, and P. Cimiano, "A machine learning approach to multilingual and cross-lingual ontology matching," in Proc. Semantic Web-ISWC 10th Int. Semantic Web Conf., Bonn, Germany. Berlin, Germany: Springer, Oct. 2011, pp. 665–680.

[2] S. Rapee and T. Igarashi, "Example-based automatic font generation," in Proc. Int. Symp. Smart Graphics. Berlin, Germany: Springer, 2010, pp. 127–138.

[3] K. Kim and J. Byun, "An implementation of optimized hangul font set for multilingual IM engine," in Proc. 6th Int. Conf. Adv. Lang. Process. Web Inf. Technol. (ALPIT), Aug. 2007, pp. 200–205.

[4] J. Ja-Yeon, "Automatic extraction of hangul stroke element using faster R-CNN for font similarity," J. Korea Multimedia Society, vol. 23, no. 8, pp. 953–964, 2020.

[5] S.-B. Lim, J. Lee, X. Zhao, and Y. Song, "Detection model of hangul stroke elements: Expansion of non- structured font and influence evaluation by stroke element combinations," Electronics, vol. 12, no. 2, p. 383, Jan. 2023.

[6] G. Chen, J. Yang, H. Jin, J. Brandt, E. Shechtman, A. Agarwala, and T. X. Han, "Large- scale visual font recognition," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 3598–3605.

[7] Y. Zhu, T. Tan, and Y. Wang, "Font recognition based on global texture analysis," IEEE Trans. Pattern Anal. Mach. Intell., vol. 23, no. 10, pp. 1192–1200, Oct. 2001.

[8] P. O'Donovan, J. L¯ibeks, A. Agarwala, and A. Hertzmann, "Exploratory font selection using crowdsourced attributes," ACM Trans. Graph., vol. 33, no. 4, pp. 1–9, Jul. 2014.

[9] S. Dawn and B. Chaparro,"Perception of fonts: Perceived personality traits and appropriate uses," Digital Fonts Reading. World Scientific, 2016, ch. 13, pp. 226–247. [Online]. Available: https://doi.org/10.1142/9789814759540 0013

[10] C. Hsien-Chih, M. Ma, and Y. C. Feng, "The features of Chinese typeface and its emotion," in Proc. Int. Conf. KANSEI Eng. Emotion Res., Mar. 2010, pp. 1–11.

[11] X. Kong and Z. Liu, "Reaserch on monolingual font recommendation based on target perception," in Proc. Int. Conf. Intell. Design (ICID), Dec. 2020, pp. 94–97.

[12] N. Samadiani and H. Hassanpour, "A neural network-based approach for recognizing multi-font printed English characters," J. Electr. Syst. Inf. Technol., vol. 2, no. 2, pp. 207– 218, Sep. 2015.