INSANtek – Jurnal Inovasi dan Sains Teknik Elektro
Volume 7 No. 1 Mei 2026
E-ISSN: 2722-547X

Pengembangan Aplikasi Check Font Versi 2 Deteksi Font Pada Dokumen PDF Dengan Algoritma Distribusi Font

Mugi Raharjo[1], Musriatun Napiah[2], Sujiliani Heristian[3],Rachmat Adi Purnama[4]

[1]Universitas Nusa Mandiri
e-mail: [1]mugi.mou@nusamandiri.ac.id

[2,3,4]Universitas Bina Sarana Informatika
e-mail : [2]musriatun.mph@bsi.ac.id, [3]Sujiliani.she@bsi.ac.id, [4]Rachmat.rap@bsi.ac.id

Abstrak - Konsistensi format dokumen merupakan salah satu aspek penting dalam penulisan karya ilmiah seperti skripsi, tesis, maupun laporan penelitian. Namun, proses pemeriksaan format dokumen seperti jenis *font*, ukuran *font,* dan spasi antar baris masih sering dilakukan secara manual sehingga membutuhkan waktu yang cukup lama serta berpotensi menimbulkan kesalahan dalam proses validasi dokumen. Penelitian ini bertujuan untuk mengembangkan aplikasi *Check Font* versi 2 berbasis *Python* yang mampu mendeteksi dan menganalisis format dokumen secara otomatis pada file berformat *PDF*. Metode yang digunakan dalam penelitian ini adalah pendekatan analisis dokumen dengan memanfaatkan pustaka *PyMuPDF* untuk mengekstraksi informasi teks, termasuk jenis *font*, ukuran *font*, serta posisi teks pada setiap halaman dokumen. Sistem kemudian melakukan proses normalisasi keluarga *font (font family normalization)* untuk mengelompokkan berbagai variasi nama font yang masih berasal dari jenis *font* yang sama. Kebaruan pada penelitian ini terletak pada pengembangan fitur analisis yang tidak hanya mendeteksi jenis *font*, tetapi juga mampu mengidentifikasi distribusi ukuran font, estimasi spasi antar baris, serta visualisasi distribusi *font* dalam bentuk grafik. Selain itu, sistem juga memberikan penandaan warna pada teks untuk menunjukkan kesesuaian atau ketidaksesuaian font terhadap standar yang dipilih pengguna. Hasil penelitian menunjukkan bahwa aplikasi *Check Font* versi 2 mampu membantu proses pemeriksaan format dokumen secara otomatis, lebih cepat, dan lebih sistematis dibandingkan pemeriksaan manual maupun versi sebelumnya.

Kata Kunci: Deteksi Font, Analisis Dokumen PDF, Normalisasi *Font Family*

Abstract - Document formatting consistency is an important aspect in the preparation of academic documents such as theses, dissertations, and research reports. However, the process of verifying document formatting, including font type, font size, and line spacing, is still commonly performed manually. This process is time-consuming and may lead to errors during document validation. This study aims to develop Check Font Application Version 2, a Python-based tool designed to automatically detect and analyze document formatting in PDF files. The proposed method utilizes a document analysis approach by extracting textual metadata using the PyMuPDF library. The system analyzes each page of the document to identify font types, font sizes, and the spatial position of text elements. A font family normalization process is applied to group different font name variations that belong to the same font family. The novelty of this research lies in the enhanced features introduced in version 2 of the application, which not only detect font types but also analyze font size distribution, estimate line spacing, and provide visualization of font distribution through graphical representation. In addition, the system implements color-based highlighting to indicate whether the detected fonts comply with the selected standard font. Experimental results show that the Check Font Version 2 application can assist users in automatically analyzing document formatting, enabling faster and more systematic identification of font inconsistencies compared to manual checking and the previous version of the application.

Keywords: Font Detection, PDF Document Analysis, Font family Normalization

PENDAHULUAN

Dokumen akademik seperti skripsi, tesis, laporan penelitian, dan artikel ilmiah memiliki standar format tertentu yang harus dipenuhi oleh penulis. Standar tersebut biasanya meliputi konsistensi penggunaan jenis *font*, ukuran *font*, serta pengaturan spasi antar baris. Konsistensi format dokumen menjadi penting karena berkaitan dengan keterbacaan dokumen, keseragaman format akademik, serta kemudahan dalam proses evaluasi oleh pihak akademik (Yusuf & Prasetyo, 2023). Dokumen memiliki peran yang penting karena berisi data-data yang dibutuhkan, baik dalam pekerjaan maupun kehidupan sehari-hari. Seiring dengan perkembangan teknologi,

dokumen kini menjadi catatan penting yang perlu didigitalisasi (Pratama et al., n.d.). Beberapa pedoman penulisan jurnal bahkan secara eksplisit menentukan jenis *font* tertentu, misalnya *Times New Roman* dengan ukuran dan spasi tertentu sebagai standar penulisan artikel ilmiah.

Namun dalam praktiknya, proses pemeriksaan kesesuaian format dokumen masih banyak dilakukan secara manual. Pemeriksaan manual sering kali memerlukan waktu yang cukup lama dan berpotensi menimbulkan kesalahan karena jumlah halaman dokumen yang banyak serta variasi format yang digunakan penulis. Seiring berkembangnya teknologi informasi, berbagai penelitian mulai mengembangkan pendekatan otomatis untuk menganalisis dokumen digital dengan memanfaatkan teknik ekstraksi teks dan analisis metadata dokumen(Pratama et al., 2024).

Penelitian sebelumnya telah menunjukkan bahwa analisis dokumen digital dapat dilakukan dengan memanfaatkan berbagai teknik komputasi seperti pengolahan citra, ekstraksi fitur teks, dan pemanfaatan bahasa pemrograman *Python* yang menyediakan berbagai pustaka pendukung untuk pemrosesan dokumen digital (Santoso & Yan, 2024). Penggunaan *python* dinilai efektif karena memiliki banyak *library* yang dapat digunakan untuk ekstraksi data teks, analisis dokumen, serta pengembangan aplikasi berbasis data (Maulidya, Suadaa, Wijayanto, & Ridho, 2025).

Beberapa penelitian juga telah mengembangkan sistem yang mampu mendeteksi ketidakkonsistenan font pada dokumen digital sebagai indikator perubahan atau manipulasi dokumen (Krismona, Ashari, Setiawan, & Rosnelly, 2025). Salah satu pendekatan yang digunakan adalah pemanfaatan metode klasifikasi berbasis pembelajaran mesin untuk mengenali variasi font pada dokumen digital (Alamin, Mutmainah, & Hayun, n.d.).

Selain itu, penelitian lain dalam bidang analisis dokumen juga memanfaatkan teknik ekstraksi informasi teks untuk mendukung berbagai proses analisis dokumen seperti deteksi plagiarisme, ekstraksi kata kunci, dan pemrosesan teks otomatis (Wibowo et al., 2024). Pendekatan tersebut menunjukkan bahwa analisis dokumen digital dapat dilakukan secara otomatis untuk meningkatkan efisiensi pengolahan dokumen dalam berbagai bidang akademik dan penelitian (Maulidya et al., 2025).

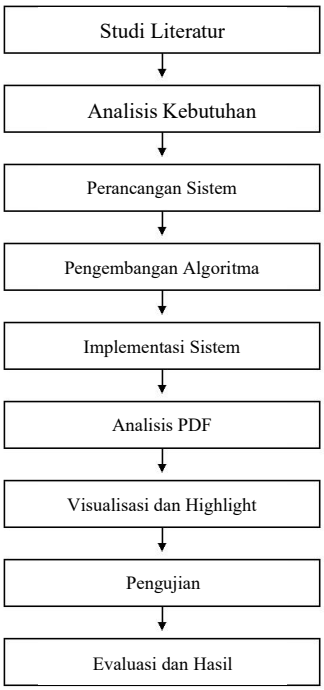
Meskipun berbagai penelitian telah membahas analisis dokumen digital, sebagian besar penelitian masih berfokus pada analisis konten teks atau pengenalan karakter. Penelitian yang secara khusus menganalisis konsistensi format dokumen seperti jenis font, ukuran font, dan spasi antar baris masih relatif terbatas (Muniroh, Rolliawati, & Amrozi, 2021). Oleh karena itu, diperlukan suatu sistem yang mampu melakukan analisis format dokumen secara otomatis sehingga dapat membantu proses pemeriksaan format dokumen akademik secara lebih cepat dan sistematis.

Berdasarkan permasalahan tersebut, penelitian ini mengembangkan *Check Font App* versi 2, yaitu aplikasi berbasis *Python* yang mampu mendeteksi penggunaan jenis font, ukuran font, serta estimasi spasi antar baris pada dokumen berformat *PDF*. Sistem ini juga dilengkapi dengan proses normalisasi keluarga *font* (*font family normalization*) sehingga berbagai variasi nama *font* yang berasal dari satu jenis font yang sama dapat diidentifikasi sebagai satu kategori *font* yang sama.

Kebaruan pada penelitian ini terletak pada pengembangan sistem analisis format dokumen yang tidak hanya mendeteksi jenis font, tetapi juga mampu melakukan analisis distribusi *font*, distribusi ukuran *font*, estimasi spasi dokumen, serta visualisasi hasil analisis dalam bentuk grafik. Selain itu, sistem juga memberikan penandaan warna pada dokumen untuk menunjukkan kesesuaian penggunaan *font* terhadap standar *font* yang dipilih oleh pengguna. Dengan adanya sistem ini diharapkan proses pemeriksaan format dokumen akademik dapat dilakukan secara lebih efisien dan sistematis.

METODE PENELITIAN

Diagram tahapan penelitian yang ditampilkan menggambarkan alur pengembangan sistem *Check Font App versi 2* secara sistematis dan terstruktur. Proses penelitian diawali dengan tahap studi literatur, yaitu mengkaji berbagai referensi terkait analisis dokumen digital, deteksi *font*, serta teknologi yang relevan. Selanjutnya dilakukan analisis kebutuhan sistem untuk mengidentifikasi fitur utama yang diperlukan, seperti deteksi jenis *font*, ukuran *font*, dan spasi dokumen. Tahap berikutnya adalah perancangan sistem, yang mencakup desain arsitektur aplikasi serta alur kerja sistem.



Sumber : Penelitian (2026)

Gambar 1. Tahapan Penelitian

1. Studi Literatur
   Tahap ini dilakukan dengan mengkaji berbagai penelitian terdahulu yang berkaitan dengan analisis dokumen digital, deteksi *font*, serta teknologi pemrosesan dokumen berbasis *Python*. Studi literatur bertujuan untuk memperoleh landasan teori serta mengidentifikasi celah penelitian yang dapat dikembangkan pada penelitian ini.
2. Analisis Kebutuhan Sistem
   Pada tahap ini dilakukan identifikasi kebutuhan sistem yang akan dikembangkan, meliputi fitur utama seperti deteksi jenis *font,* ukuran *font*, serta estimasi spasi antar baris. Selain itu, ditentukan juga kebutuhan input, output, serta spesifikasi sistem yang akan dibangun.
3. Perancangan Sistem Deteksi *Font*
   Tahap perancangan dilakukan untuk menyusun arsitektur sistem dan alur kerja aplikasi. Pada tahap ini dirancang bagaimana sistem akan membaca dokumen *PDF*, mengekstraksi data teks, serta menampilkan hasil analisis kepada pengguna.
4. Pengembangan Algoritma (*Font Detection + Normalisasi Font*)
   Tahap ini merupakan inti dari penelitian, yaitu mengembangkan algoritma untuk mendeteksi jenis font pada dokumen serta melakukan normalisasi *font family*. Normalisasi ini bertujuan untuk mengelompokkan berbagai variasi nama font yang berasal dari satu keluarga font yang sama agar hasil analisis lebih akurat.
5. Implementasi Sistem (*Python, PyMuPDF, Streamlit*)
   Pada tahap ini dilakukan implementasi sistem menggunakan bahasa pemrograman *Python* dengan memanfaatkan pustaka *PyMuPDF* untuk ekstraksi data dokumen dan *Streamlit* sebagai antarmuka pengguna berbasis web.
6. Analisis Dokumen PDF (*Font, Size, Spacing*)
   Sistem melakukan proses analisis terhadap dokumen *PDF* dengan mengekstraksi informasi jenis font, ukuran font, serta menghitung estimasi jarak antar baris untuk menentukan nilai spasi dokumen.
7. Visualisasi dan *Highlighting*
   Hasil analisis ditampilkan dalam bentuk visualisasi, seperti grafik distribusi *font*, serta penandaan warna pada dokumen. Warna digunakan untuk membedakan antara *font* yang sesuai dengan standar (benar) dan font yang tidak sesuai.

8. Pengujian Sistem
   Tahap pengujian dilakukan untuk mengevaluasi kinerja sistem dalam mendeteksi *font*, ukuran *font*, dan spasi dokumen. Pengujian dilakukan dengan menggunakan beberapa dokumen uji untuk memastikan sistem bekerja dengan baik.
9. Evaluasi dan Analisis Hasil
   Tahap terakhir adalah melakukan evaluasi terhadap hasil pengujian sistem. Pada tahap ini dianalisis tingkat akurasi, kelebihan, serta kekurangan sistem yang dikembangkan sebagai dasar untuk pengembangan lebih lanjut.

HASIL DAN PEMBAHASAN

Aplikasi *Check Font App versi 2* berhasil dikembangkan sebagai sistem berbasis *web* menggunakan *Python* dengan *framework Streamlit* serta pustaka *PyMuPDF* untuk melakukan analisis dokumen *PDF*. Sistem ini mampu melakukan ekstraksi informasi teks dari dokumen, kemudian menganalisis jenis *font*, ukuran *font*, serta estimasi spasi antar baris. Berdasarkan hasil pengujian terhadap beberapa dokumen *PDF*, aplikasi mampu menampilkan distribusi penggunaan *font* dalam bentuk persentase, distribusi ukuran *font,* serta estimasi nilai spasi dokumen. Selain itu, sistem juga menghasilkan *file PDF* baru yang telah diberi penandaan warna untuk menunjukkan kesesuaian penggunaan font terhadap standar yang dipilih pengguna.

Aplikasi yang dikembangkan memiliki beberapa fungsi utama sebagai berikut:

1. Pemilihan *Font* Standar
   Pengguna dapat memilih jenis font utama (misalnya *Times New Roman, Arial*, atau *Calibri)* sebagai acuan dalam analisis dokumen.
2. Analisis Jenis *Font*
   Sistem mendeteksi seluruh *font* yang digunakan dalam dokumen dan menghitung distribusinya dalam bentuk persentase.
3. Analisis Ukuran *Font*
   Sistem mengidentifikasi variasi ukuran font yang digunakan dalam dokumen dan menyajikannya dalam bentuk distribusi frekuensi.
4. Estimasi Spasi Antar Baris
   Sistem menghitung jarak antar baris berdasarkan posisi koordinat teks dalam PDF untuk menentukan estimasi *line spacing*.
5. Visualisasi Data
   Hasil analisis ditampilkan dalam bentuk grafik distribusi *font* sehingga memudahkan pengguna dalam memahami komposisi *font* dalam dokumen.
6. *Highlighting* Dokumen
   Sistem memberikan penandaan warna pada dokumen:
   Hijau : *font* sesuai standar
   Merah : *font* tidak sesuai

Secara fungsional, aplikasi ini memberikan kemudahan bagi pengguna dalam melakukan pemeriksaan format dokumen secara otomatis. Pengguna dapat menentukan jenis *font* standar yang diinginkan sebelum melakukan proses analisis. Sistem kemudian akan mengidentifikasi seluruh jenis *font* yang digunakan dalam dokumen dan membandingkannya dengan *font* standar yang dipilih. Hasil analisis ditampilkan dalam bentuk distribusi persentase penggunaan *font*, distribusi ukuran *font,* serta estimasi nilai spasi dokumen. Selain itu, sistem juga menyediakan visualisasi dalam bentuk grafik distribusi *font* yang membantu pengguna memahami komposisi penggunaan font dalam dokumen secara lebih intuitif.

Perbedaan utama antara versi 1 dan versi 2 terletak pada peningkatan algoritma analisis yang digunakan. Pada versi sebelumnya, deteksi font dilakukan menggunakan metode pencocokan string sederhana sehingga variasi nama *font* yang masih berasal dari satu keluarga font sering kali dianggap berbeda. Pada versi 2, sistem telah dilengkapi dengan mekanisme *font family normalization* yang mampu mengelompokkan berbagai variasi nama *font* ke dalam satu kategori yang sama. Dengan pendekatan ini, font seperti *TimesNewRomanPSMT*, *TimesNewRoman-Bold*, dan variasi lainnya dapat dikenali sebagai *Times New Roman*. Selain itu, versi 2 juga menambahkan kemampuan analisis ukuran font dan estimasi spasi antar baris, yang sebelumnya belum tersedia. Peningkatan ini menjadikan sistem lebih akurat dan komprehensif dalam melakukan analisis format dokumen.

Dari sisi tampilan, aplikasi dirancang dengan antarmuka yang sederhana dan interaktif sehingga mudah digunakan oleh berbagai kalangan pengguna. Halaman utama aplikasi menyediakan fitur pemilihan *font* standar, unggah *file PDF*, serta tampilan hasil analisis secara langsung. Informasi hasil analisis ditampilkan secara terstruktur, meliputi distribusi *font* dalam bentuk persentase, distribusi ukuran *font*, serta estimasi spasi dokumen. Selain itu,

grafik distribusi *font* juga ditampilkan untuk memberikan gambaran visual yang lebih jelas mengenai penggunaan font dalam dokumen.

```
1    import streamlit as st
2    import fitz
3    from collections import Counter
4    import matplotlib.pyplot as plt
5    import io
6    import numpy as np
```

Sumber : Penelitian (2026)

Gambar 2. *Library python*

Hasil utama dari sistem ini adalah keluaran berupa *file PDF* hasil analisis yang telah diproses secara otomatis. *File PDF* tersebut terdiri dari halaman *cover*, halaman ringkasan analisis, serta dokumen asli yang telah diberi penandaan warna. Pada bagian ringkasan, sistem menampilkan informasi distribusi font, distribusi ukuran *font*, serta estimasi spasi yang digunakan dalam dokumen. Sementara itu, pada bagian dokumen, sistem memberikan penandaan warna untuk membedakan *font* yang sesuai dan tidak sesuai dengan standar yang dipilih. Warna hijau digunakan untuk menunjukkan *font* yang sesuai, sedangkan warna merah digunakan untuk menandai font yang tidak sesuai, disertai anotasi informasi jenis *font* yang terdeteksi.

Berdasarkan hasil pengujian, aplikasi *Check Font App versi 2* menunjukkan kinerja yang lebih baik dibandingkan versi sebelumnya. Sistem mampu mengidentifikasi variasi penggunaan *font* secara lebih akurat berkat penerapan normalisasi keluarga *font*. Selain itu, penambahan fitur analisis ukuran *font* dan spasi memberikan informasi yang lebih lengkap mengenai struktur dokumen. Visualisasi dalam bentuk grafik serta penandaan warna pada dokumen juga terbukti mempermudah pengguna dalam mengidentifikasi kesalahan format. Dengan demikian, aplikasi ini mampu meningkatkan efisiensi dan akurasi dalam proses pemeriksaan format dokumen akademik, sehingga dapat menjadi solusi yang efektif untuk menggantikan proses pemeriksaan manual yang selama ini digunakan.

Pada penelitian ini, sistem dikembangkan menggunakan bahasa pemrograman *Python* dengan memanfaatkan pustaka *Streamlit* sebagai antarmuka pengguna, *PyMuPDF* untuk pemrosesan dokumen *PDF*, serta *matplotlib* untuk visualisasi data. Sistem diawali dengan konfigurasi antarmuka berbasis web yang memungkinkan pengguna memilih *font* standar dan mengunggah dokumen PDF. Salah satu komponen utama dalam sistem ini adalah mekanisme normalisasi keluarga *font* (*font family normalization*), yang direalisasikan melalui struktur *dictionary* berisi berbagai variasi nama *font.* Implementasi normalisasi ini ditunjukkan pada potongan kode berikut :

```
FONT_ALIASES = {
    "Times New Roman": [
        "timesnewroman",
        "timesnewromanps",
        "timesnewromanpsmt",
        "timesnewromanps-boldmt"
    ]
}
```

Sumber : Penelitian (2026)

Gambar 3. *Font* alias

Struktur tersebut digunakan dalam fungsi pengecekan kesamaan keluarga *font,* sehingga variasi *font* yang berbeda tetapi masih satu keluarga dapat dikenali sebagai *font* yang sama. Hal ini diimplementasikan dalam fungsi berikut:

```python
def is_same_font_family(font_name, target_font):
    font_name = font_name.lower()
    for alias in FONT_ALIASES[target_font]:
        if alias in font_name:
            return True
    return False
```

Sumber : Penelitian (2026)

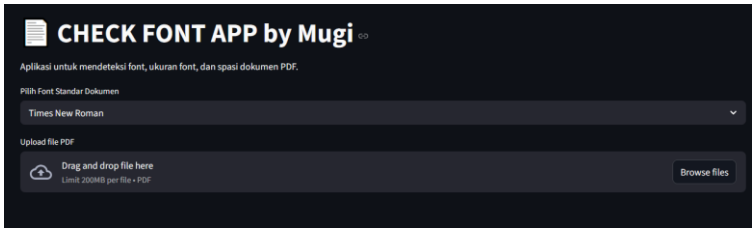
Gambar 4. Fungsi kesamaan

Tahapan utama sistem adalah proses analisis dokumen *PDF*, di mana sistem membaca setiap halaman dokumen dan mengekstraksi informasi teks menggunakan *PyMuPDF*. Proses ini dilakukan dengan mengakses struktur teks berupa *blocks*, *lines*, dan *spans* untuk memperoleh informasi font dan ukuran teks. Implementasi analisis ini ditunjukkan pada potongan kode berikut:

```python
for page in doc:
    blocks = page.get_text("dict")["blocks"]
    for b in blocks:
        if "lines" in b:
            for l in b["lines"]:
                for s in l["spans"]:
                    font_raw = s["font"]
                    size = s["size"]
```

Sumber : Penelitian (2026)

Gambar 5. *Analysis Code*



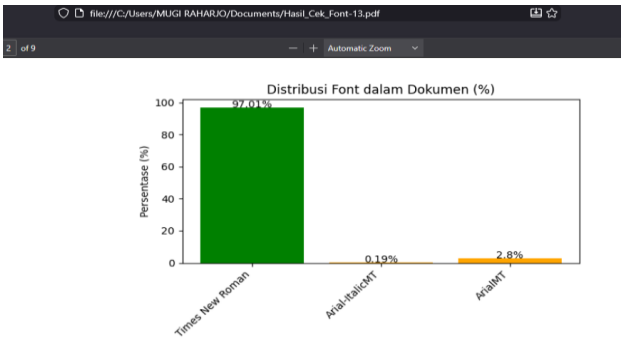
Sumber : Penelitian (2026)

Gambar 6. Tampilan Aplikasi

Sumber : Penelitian (2026)

Gambar 7. Tampilan Aplikasi Hasil

Hasil analisis kemudian diolah untuk menghasilkan distribusi *font* dan ukuran font menggunakan struktur data *Counter*. Selanjutnya, sistem menampilkan visualisasi dalam bentuk grafik distribusi *font* menggunakan *matplotlib*, dengan pewarnaan yang membedakan font standar dan *font* lainnya. Selain visualisasi, sistem juga menghasilkan dokumen *PDF* hasil analisis yang telah diberi penandaan warna (*highlighting*).



Sumber : Penelitian (2026)

Gambar 8. Visualisasi Hasil pada *PDF*

Pada tahap ini, setiap teks dalam dokumen diperiksa dan diberikan warna hijau apabila sesuai dengan font standar, serta warna merah apabila tidak sesuai. Tahap akhir adalah pembuatan laporan hasil analisis dalam bentuk *file PDF* yang terdiri dari halaman *cover*, ringkasan analisis, serta dokumen asli yang telah diberi *highlight.* Dengan demikian, sistem yang dikembangkan tidak hanya mampu mendeteksi kesesuaian format dokumen secara otomatis, tetapi juga menyajikan hasil analisis secara visual dan informatif, sehingga memudahkan pengguna dalam melakukan evaluasi dokumen secara lebih cepat dan akurat.

KESIMPULAN

Berdasarkan hasil penelitian dan pengembangan yang telah dilakukan, aplikasi *Check Font App versi 2* berhasil dikembangkan sebagai sistem berbasis web yang mampu melakukan analisis format dokumen *PDF* secara otomatis. Sistem ini mampu mendeteksi jenis *font*, ukuran *font,* serta estimasi spasi antar baris dalam dokumen dengan memanfaatkan ekstraksi metadata teks menggunakan pustaka *PyMuPDF*. Selain itu, penerapan algoritma *font family normalization* terbukti mampu meningkatkan akurasi dalam mengelompokkan berbagai variasi nama font yang masih berada dalam satu keluarga *font* yang sama.

Hasil pengujian menunjukkan bahwa aplikasi mampu mengidentifikasi distribusi penggunaan *font* dalam dokumen serta membedakan *font* yang sesuai dan tidak sesuai dengan standar yang ditentukan pengguna. *Fitur* visualisasi dalam bentuk grafik serta penandaan warna pada dokumen juga membantu pengguna dalam memahami hasil analisis secara lebih intuitif. Dibandingkan dengan versi sebelumnya, aplikasi versi 2 memiliki peningkatan signifikan, baik dari sisi algoritma maupun fitur analisis, yaitu dengan ditambahkannya deteksi ukuran *font,* estimasi spasi, serta visualisasi hasil yang lebih informatif.

Dengan demikian, aplikasi ini dapat digunakan sebagai alat bantu dalam proses pemeriksaan format dokumen akademik secara lebih cepat, sistematis, dan akurat. Penggunaan aplikasi ini diharapkan dapat meningkatkan efisiensi serta mengurangi kesalahan dalam proses validasi dokumen, sehingga mendukung kualitas penulisan karya ilmiah yang lebih baik.

REFERENSI

Alamin, Z., Mutmainah, S., & Hayun, M. (n.d.). Optimasi Ekstraksi Fitur Citra Karakter Font Menggunakan Algoritma Support Vector Machines (SVM) untuk Klasifikasi Tipografi. https://doi.org/10.34304/scientific.v2i1.344

Krismona, L., Ashari, A., Setiawan, A., & Rosnelly, R. (2025). Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD Deteksi Ketidakkonsistenan Font Sebagai Indikator Pemalsuan dan Penyuntingan Dokumen Digital Menggunakan Convolutional Neural Network (CNN) Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD, *8*, 214–226.

Marcoulides, G. a. (2005). *Discovering Knowledge in Data: an Introduction to Data Mining:Discovering Knowledge in Data: An Introduction to Data Mining*. *Journal of the American Statistical Association* (Vol. 100). https://doi.org/10.1198/jasa.2005.s61

Maulidya, L., Suadaa, L. H., Wijayanto, A. W., & Ridho, F. (2025). MULTI-SOURCE DATA FUSION FOR

DATA EXTRACTION AND INTEGRATION OF SCIENTIFIC PUBLICATIONS IN ACADEMIC INSTITUTION STIS, *14*(2), 348–363.

Muniroh, L., Rolliawati, D., & Amrozi, Y. (2021). Trend Information System Research Topics in the SINTA Journal for the Period 2015-2019, *10*, 283–290.

Pratama, Y. P., Buwono, P. F., Rabbani, M. A., Jakarta, P. N., Beji, K., & Depok, K. (2024). MENGGUNAKAN KERTAS THERMAL BERBASIS, *12*(3), 2819–2824.

Pratama, Y. P., Kartika, R. N., Buwono, P. F., Rabbani, A., Grafika, T., & Jakarta, P. N. (n.d.). Rancang Bangun Pemindai Dokumen Berbasis Python IDE Abstrak, *3*(1), 11–15.

Santoso, J. T., & Yan, S. (2024). A Hybrid Approach to Typo Correction in Indonesian Documents Using Levenshtein Distance, *3*(2), 151–168. https://doi.org/10.51903/jtie.v3i2.184

Wibowo, I. S., Witanti, A., Susilawati, I., Mercu, U., Yogyakarta, B., Sleman, K., & Informasi, F. T. (2024). Keyword Extraction Judul Berita Online Di Indonesia Menggunakan Metode TF-IDF 1,2, *11*(1).

Yusuf, A. R., & Prasetyo, A. (2023). The Use of Information Retrieval in Student Academic Document Plagiarism Detection System, *6*(2). https://doi.org/10.32877/bt.v6i2.1063