p-ISSN : 2443-2210
e-ISSN : 2443-2229

Analisis Implementasi Sistem Penilaian Otomatis Jawaban Singkat Berbahasa Indonesia

http://dx.doi.org/10.28932/jutisi.v12i1.12453

Riwayat Artikel
Received: 06 Juli 2025 | *Final Revision*: 30 Maret 2026 | *Accepted*: 30 Maret 2026

Natanael Tegar Pramudya [#1], Lucia Dwi Krisnawati[✉#2], Aditya Wikan Mahastama[#3]

[#] *Informatika, Universitas Kristen Duta Wacana*
Jl. Dr. Wahidin Sudirohusodo No.5-25, Yogyakarta, 55224, Indonesia
[1]natanael.tegar@ti.ukdw.ac.id
[2]krisna@staff.ukdw.ac.id
[3]mahas@staff.ukdw.ac.id
[✉]Corresponding author: krisna@staff.ukdw.ac.id

Abstrak — **Penerapan *Learning Management System* (LMS) dalam dunia pendidikan berkembang sangat pesat usai Pandemi Covid-19. Evaluasi pembelajaran melalui LMS sering diberikan dalam bentuk soal objektif seperti soal pilihan ganda, benar-salah maupun soal dengan pertanyaan terbuka. Penilaian untuk soal bertipa uraian singkat sering menyita banyak waktu dan objektifitas penilaian berkurang jika tidak diselesaikan dalam sekali waktu koreksi. Untuk mengatasi ini, dikembangkan sistem penilaian otomatis jawaban terbuka yang saat ini telah menjadi 2 bidang penelitian yaitu *Automatic Essay Scoring (AES)* dan *Automatic Short Answer Grading (ASAG).* Penelitian ini, awalnya menerapkan *Support Vector Machine* (SVM) Multikelas dalam pengembangan purwarupa sistem penilaian otomatis jawaban singkat -- ASAG. Dengan memanfaatkan kombinasi fitur *Unigram & Bigram Cosine Similarity*, *Type-Token Ratio*, dan *Word Count Ratio,* evaluasi model SVM berkernel RBF dengan $\gamma = 100$ menghasilkan nilai presisi tertinggi pada tahap validasi. Di tahap pengujian, presisi prediksi model dalam penilaian jawaban mencapai 0,49 dan nilai kesalahan prediksi dengan metrik *Root Mean Squared Error* (RMSE) mencapai 2,77. Kinerja model SVM dinilai kurang memuaskan, maka dikembangkan model lain seperti *Logistic Regression* dan *k-Nearest Neighbor* (k-NN). Dibangun dengan arsitektur yang sama dengan SVM, dan pelatihan serta pengujian dengan data yang sama, maka model k-NN menunjukkan kinerja prediksi tertinggi, namun kinerja prediksi model *Logistic Regression* lebih stabil dan tinggi di semua metrik evaluasi, termasuk nilai RMSE. Untuk itu, penelitian ini merekomendasikan model *Logistic Regression* dalam pengembangan purwarupa sistem Penilaian Otomatis Jawaban Pendek.**

Kata kunci— **k-NN; Pembelajaran Terbimbing; Regresi Logistik; Sistem Penilaian otomatis jawaban terbuka - ASAG; SVM Multikelas.**

Analysis on the Implementation of an Indonesian Automatic Short Answer Grading System

Abstract — ***The use of Learning Management Systems (LMS) in education has increased significantly since the COVID-19 pandemic. Assessments delivered through LMS commonly include objective questions such as multiple-choice, true–false, and open-ended items. However, grading short-answer questions is time-consuming and may reduce objectivity when evaluations are not conducted consistently in a single grading session. To address this challenge, automated open-ended answer grading systems have been***

developed, giving rise to two main research areas: Automatic Essay Scoring (AES) and Automatic Short Answer Grading (ASAG). This study focuses on developing a prototype ASAG system. Initially, a multiclass Support Vector Machine (SVM) model was implemented using a combination of textual features, including Unigram and Bigram Cosine Similarity, Type–Token Ratio, and Word Count Ratio. Evaluation results showed that an SVM with an RBF kernel and γ = 100 achieved the highest precision during the validation stage. However, in the testing phase, the model produced a prediction precision of 0.49 and a Root Mean Squared Error (RMSE) of 2.77, indicating unsatisfactory performance. To improve results, additional models—Logistic Regression and k-Nearest Neighbor (k-NN)—were developed using the same architecture and dataset. The k-NN model achieved the highest precision, while Logistic Regression demonstrated more highly stable predictive performances across evaluation metrics and lower RMSE values. Based on these findings, this study recommends Logistic Regression as the most suitable model for developing a prototype automatic short-answer grading system.

Keywords— ***Automatic Short Answer Grading; k-NN; Logistic Regression; Multiclass SVM; Supervised Learning Models***

I. PENDAHULUAN

Penerapan Teknologi Informasi dan Komunikasi di dunia pendidikan mengalami peningkatan pesat lima tahun terakhir ini. Salah satunya adalah penggunaan *Learning Management System* (LMS) yang terintegrasi dengan Sistem Cerdas (*Artificial Intelligence – AI*). Sebagai alternatif pembelajaran tatap muka, pembelajaran yang menggunakan LMS sering menerapkan sistem penilaian otomatis (*Automatic Grading System*) untuk menilai tugas-tugas maupun ujian yang diberikan. Penilaian otomatis jawaban soal objektif dan tertutup seperti Pilihan Ganda, Benar-Salah, maupun mencocokkan menunjukan presisi yang tinggi. Namun Algoritma sistem penilaian otomatis untuk soal terbuka dengan jawaban uraian masih memiliki keterbatasan. Peningkatan kinerja Sistem Penilaian Otomatis untuk jawaban terbuka terus dikembangkan yang sampai saat ini membuahkan 2 bidang penelitian yaitu *Automatic Essay Scoring* (AES) dan *Automatic Short Answer Grading* (ASAG). Krisnawati dkk [1] menyoroti perbedaan tugas dan deskripsi kerja algoritma AES dan ASAG yang didasarkan pada karakteristik soal dan jawaban, dimana soal esai menuntut jawaban argumentatif ataupun persuasif, jawaban bersifat sangat subyektif dan sangat terbuka, panjang jawaban bervariasi dari 2 paragraf sampai lebih dari 2 halaman [1]. Tugas algoritma AES adalah menilai jawaban esai peserta didik dengan menggunakan *Machine Learning* [2].

Berbeda dari esai, soal objektif terbuka menghendaki jawaban uraian singkat dalam bahasa natural, panjang jawaban bervariasi dari beberapa frasa sampai sekitar 2 paragraf dan jawaban bersifat objektif sehingga jawaban bisa ditentukan benar atau salahnya [1]. Contoh soal objektif terbuka ini adalah soal yang menanyakan fakta, definisi, narasi event ataupun deskripsi dan penjelasan konsep. Dengan demikian, soal objektif terbuka ini memerlukan kunci jawaban. Penilaian jawaban objektif terbuka dengan jawaban singkat ini masuk ranah ASAG yang oleh Li dkk didefinisikan sebagai sistem otomatis yang menilai jawaban singkat siswa sebagai respon dengan cara membandingkannya dengan kunci jawaban guru [3].

Pengembangan model sistem ASAG dengan performa akurasi penilaian yang tinggi bukanlah tugas yang mudah. Keragaman dalam penulisan jawaban serta kemungkinan derajat kebenaran, termasuk variasi penggunaan kata, struktur kalimat dan contoh-contoh yang berbeda dari kunci jawaban menjadi tantangan dalam pengembangan sistem ASAG. Oleh karena itu, dalam penelitian ini soal akan dibatasi untuk soal bertipe narasi dan penjelasan (*narrative & explanatory)*. Penelitian ini menggunakan model *Support Vector Machine* (SVM) Multikelas dalam membangun sistem penilaian otomatis. SVM Multikelas dipilih karena kemampuannya untuk mengklasifikasikan data ke dalam beberapa kategori. Model ini cocok untuk sistem penilaian otomatis di mana soal uraian dinilai berdasarkan beberapa tingkat kualitas jawaban atau kategori skor.

Penelitian sebelumnya dengan objek teks berbahasa Indonesia lebih sering berfokus pada AES, seperti meningkatkan sistem AES dengan model *single layer neural network* dan BERT *sentence embedding* [2], pengembangan model untuk menggenerasi pola jawaban dan mengevaluasi jawaban siswa dengan *word2vec* dan SVM [4], dan menggunakan model SVM untuk memprediksi keberhasilan esai kandidat beasiswa [5]. Munir et al. [6] menunjukkan bahwa *cosine similarity* memiliki performa yang baik dibandingkan *jaccard similarity* dari kombinasi metode *term frequency* dan *n-gram* untuk menilai kemiripan jawaban dan kunci jawaban siswa. Hal ini memperkuat pentingnya pemilihan metode representasi teks dan metrik kesamaan dalam sistem penilaian otomatis. Dalam konteks ASAG, umumnya sistem menilai jawaban siswa berdasarkan kunci jawaban yang sudah ditentukan. Seperti, Lubis et al. [7] yang membangun sistem ASAG menggunakan kemiripan semantik antara jawaban siswa dan kunci jawaban tunggal dengan menggunakan teknik *word embedding*. Terdapat studi lain yang berfokus pada eksplorasi metode klasifikasi. Madala et al. [8] membandingkan tiga model klasifikasi yaitu, SVM, k-NN, dan *Linear Regression* dengan menggunakan 9 fitur. Penelitian tersebut menunjukkan bahwa model k-NN memberikan akurasi terbaik pada dua set data, sementara SVM unggul pada satu set data lainnya. Ini menunjukkan bahwa pemilihan model klasifikasi juga memainkan peran penting dalam performa sistem ASAG.

Penelitian ini bertujuan untuk menganalisis implementasi 3 model Pembelajaran Terbimbing, yakni SVM multikelas, k-NN, dan Regresi Logistik ke dalam sistem penilaian soal uraian otomatis guna mempertahankan sistem pembelajaran digital yang ringan. Kelayakan model ini dilihat berdasarkan hasil dari nilai metrik evaluasi. Hasil dari penelitian ini diharapkan

dapat menjadi referensi dalam pengembangan sistem penilaian soal uraian otomatis, serta memberikan kontribusi sebagai acuan bagi penelitian selanjutnya di bidang yang sama.

II. METODE PENELITIAN

A. Dataset Penelitian

Yang menjadi objek dalam penelitian ini adalah dataset berisi 12 soal bertipe *narrative* dan *explanatory* yang didapatkan dengan metode pengumpulan data sumber sekunder. Dalam dataset yang digunakan terdapat 90 hingga 100 jawaban siswa beserta nilai dan nilai maksimal untuk tiap soal yang ada. Tiap jawaban siswa ini akan diberi label A hingga E berdasarkan nilai jawaban dan nilai maksimal tiap soal. Dari 12 soal pada dataset, hanya beberapa soal yang akan digunakan dalam penelitian, yaitu soal yang memiliki populasi jawaban lebih dari 5 di seluruh kelas. Persebaran label nilai untuk tiap soal ditampilkan pada Tabel 1.

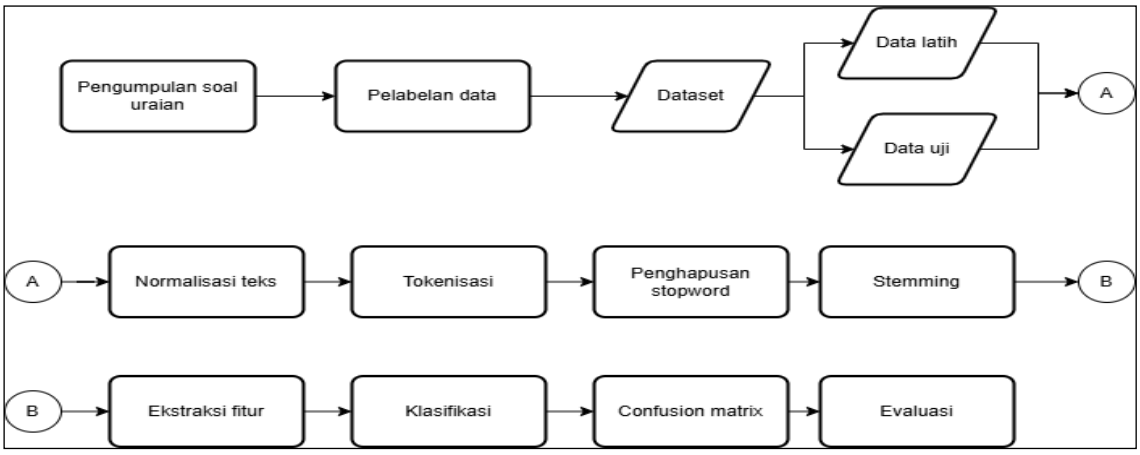
TABEL 1
PERSEBARAN LABEL PADA DATASET

ID / Label	A	B	C	D	E	Total
IDSoal1	42	14	25	10	10	101
IDSoal2	5	10	5	5	16	42
IDSoal3	7	6	10	5	5	33
IDSoal4	5	5	5	12	5	32
IDSoal5	5	5	5	5	22	42
IDSoal6	7	5	5	5	22	44
IDSoal7	13	5	5	8	9	40
IDSoal8	63	10	10	11	18	112
IDSoal9	48	12	10	10	23	103
IDSoal10	40	41	10	10	13	114
IDSoal11	22	10	13	10	65	120
IDSoal12	5	6	14	5	5	35
Total	262	129	117	96	213	817

B. Rancangan Proses Penelitian

Kunci jawaban dan jawaban siswa dari dataset akan melalui pra-pemrosesan teks, dimana akan dilakukan normalisasi teks, tokenisasi, penghapusan stopword, dan stemming, sehingga didapatkan luaran token bersih. Selanjutnya, token bersih ini dikomputasi menjadi vektor yang berfungsi sebagai fitur. Fitur yang diekstraksi adalah *word count ratio*, *type-token ratio*, dan nilai kemiripan dari jawaban dan kunci jawaban dengan unit leksikal *n-gram*, yaitu *unigram* dan *bigram*. Hasil yang didapat dari tahap ekstraksi fitur disatukan ke dalam sebuah matriks. Sumbu X dari matriks digunakan untuk merepresentasikan nilai tiap fitur beserta dan sumbu Y merepresentasikan label nilai. Fitur dan label dalam himpunan (X,Y) ini dibentuk dari jawaban tiap siswa dari masing-masing pasangan soal dan kunci jawaban. Jumlah data di himpunan (X,Y) kemudian akan dibagi menjadi data latih dan data uji.

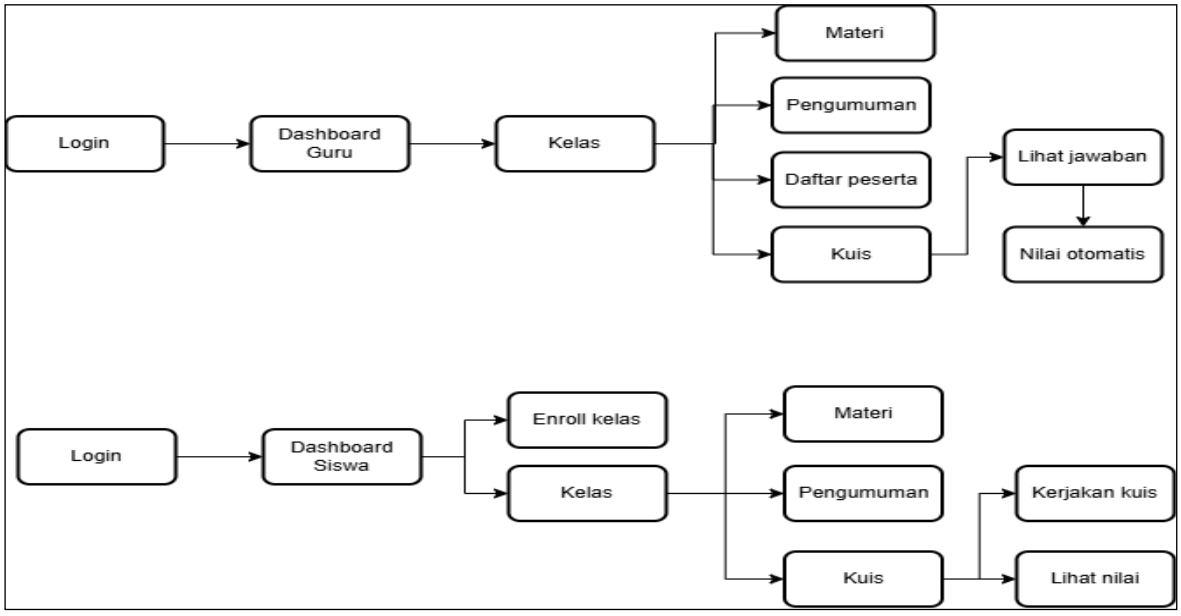
Lalu dengan 3 model yakni SVM multikelas, k-NN, dan Regresi Logistik dilakukan klasifikasi jawaban dari data uji dengan memprediksi label tiap sampel data berdasarkan pengetahuan yang didapat dari pelatihan dengan data latih. Hasil klasifikasi sistem akan dijadikan input ke dalam sebuah *confusion matrix* berukuran 5x5. Sumbu Y' dari matriks tersebut merupakan hasil prediksi nilai, yang dibagi ke dalam 5 kelas, yakni nilai A hingga E, sedangkan sumbu Y merupakan label yang sebenarnya. Hasil prediksi sistem (Y') akan di cocokkan dengan label nilai yang sebenarnya (Y) menggunakan paket yang disediakan Pustaka Scikit-Learn *confusion matrix*. Metrik evaluasi yang digunakan adalah R*ecall*, *Precision*, *f1-score*, dan *accuracy*, serta *Root Mean Square Error* (RMSE). Visualisasi rancangan penelitian dapat dilihat pada Gambar 1.



Gambar 1. Diagram Alir Penelitian

C. Perancangan Learning Management System (LMS)

Pada penelitian ini, proses penilaian otomatis akan diimplementasikan pada aplikasi ASAG berbasis web yang merupakan simulasi LMS khusus soal dengan jawaban uraian singkat. Gambar 2 menampilkan hirarki menu dari LMS yang akan digunakan berdasarkan kategori penggunanya, yakni guru/pengajar/dosen dan peserta didik/mahasiswa.



Gambar 2. Hirarki menu LMS

Gambar 2 menunjukkan halaman LMS dibagi menjadi halaman guru dan halaman siswa. Ketika *login* dengan akun guru, halaman *dashboard* guru akan ditampilkan. Di halaman *dashboard* guru ini ditampilkan daftar seluruh kelas yang sudah dibuka oleh guru yang bersangkutan dan dapat memilih kelas untuk melakukan aksi lebih lanjut pada kelas tersebut. Pengguna gurua dapat menambahkan kelas baru dan menghapus kelas yang sudah ada. Di halaman kelas, terdapat 4 pilihan menu yaitu materi, pengumuman, daftar peserta, dan kuis yang tiap menu yang dipilih akan menuju ke halaman detail masing-masing menu. Pada halaman materi, pengguna guru dapat menambahkan *file* materi baru, menghapus *file* materi, dan melihat seluruh *file* materi yang dibuat di kelas tersebut. Mirip seperti halaman materi, guru juga dapat menambahkan pengumuman baru di halaman pengumuman, menghapus pengumuman yang ada serta melihat pengumuman yang telah dibuat di kelas tersebut. Di halaman daftar peserta, guru dapat melihat seluruh siswa yang sudah mendaftar (*enroll*) ke kelas tersebut. Di halaman ini

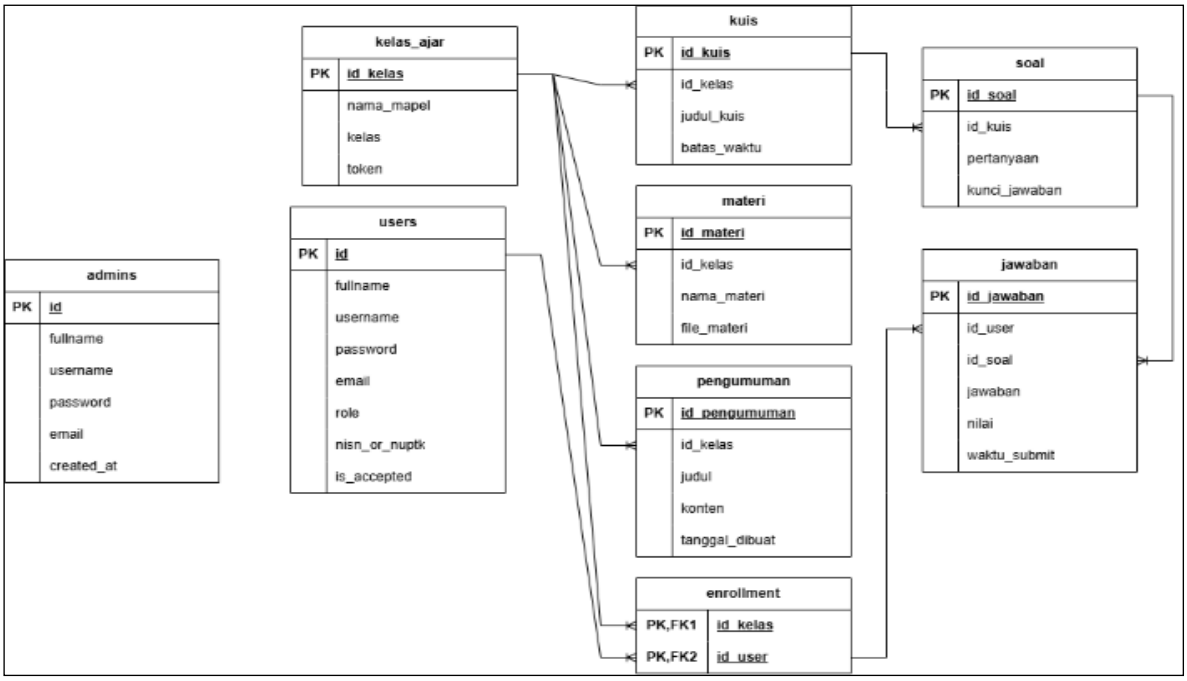
ditampilkan pula kode *enrollment* yang dapat digunakan oleh siswa untuk bergabung dengan kelas yang dibuat. Selain melihat seluruh peserta, di halaman ini guru juga dapat menghapus peserta yang telah bergabung.

Menu yang terakhir adalah kuis, pada halaman kuis akan ditampilkan daftar seluruh kuis yang dibuat di kelas terpilih. Selain itu, guru juga bisa membuat kuis baru dengan judul kuis dan batas waktu pengerjaan, selanjutnya pengguna diarahkan ke halaman edit kuis untuk membuat soal dan kunci jawaban. Ketika guru memilih kuis pada halaman kuis, maka akan ada menu untuk melihat jawaban siswa, di halaman ini jawaban siswa untuk seluruh soal akan ditampilkan.

Untuk mekanisme penilaian otomatis, ketika tombol 'Nilai Semua Jawaban' ditekan, piranti guru atau secara teknis *client* akan mengirimkan tabel berisi id jawaban, id soal, kunci jawaban, dan jawaban yang ada ke server *back-end*. Pada *endpoint* dari *back-end* inilah akan dijalankan sistem penilaian otomatis dengan *input* yang sudah diberikan dan akan mengembalikan luaran *(ouput)* berupa nilai prediksi.

Untuk pengembangan LMS ini, baik *front-end* maupun *back-end* menggunakan *Flask*. Flask merupakan salah satu *framework web* Python yang paling populer dan menawarkan fleksibilitas serta kemudahan dalam membangun aplikasi *web* secara efektif. Flask dipilih juga karena sifatnya yang ringan namun tetap menyediakan fitur-fitur esensial yang diperlukan untuk membangun aplikasi *web*. Selain itu, Flask memberikan keleluasaan pengaturan struktur proyek, sehingga mempermudah penyesuaian dengan kebutuhan spesifik sistem, seperti pengelolaan data, penilaian jawaban uraian singkat secara otomatis.

Dalam pengembangan sistem ini, DBMS yang dipilih adalah PostgreSQL, karena memiliki kemampuan yang andal dalam menangani data dalam jumlah besar dengan integritas yang tinggi. Pilihan ini juga didasarkan pada kompatibilitasnya dengan *framework* Flask yang digunakan dalam pengembangan sistem, sehingga mempermudah integrasi antara antarmuka pengguna dengan *database*. Pada Gambar 3, dapat dilihat *Entity Relation Diagram* dari basisdata yang digunakan.



Gambar 3. Rancangan *Entity Relation Diagram* LMS

III. HASIL DAN PEMBAHASAN

A. Implementasi Sistem

Sistem ASAG ini dikembangkan dengan bahasa pemrograman python yang mengolah dataset dalam format csv. Sistem ini juga menggunakan beberapa pustaka dari python untuk memudahkan pengembangannya, yaitu *pandas, numpy, sklearn*, *Sastrawi, imblearn*, dan *pickle*. Penelitian ini menggunakan skenario *prompt-specific* ASAG, sehingga sampel soal yang diuji berasal dari sampel soal yang dilatihkan dengan membagi sampel jawaban untuk data latih dan data uji. Karena pembagian ini, beberapa soal pada dataset tidak memenuhi jumlah sampel jawaban yang diperlukan, sehingga tidak digunakan dalam penelitian. IDSoal yang memenuhi syarat dan digunakan dalam penelitian adalah IDSoal 1, IDSoal 8, IDSoal 9, IDSoal 10,

dan IDSoal 11 yang soal dan kunci jawabannya dapat dilihat pada Tabel 2. Dataset yang dipilih tersebut akan dibagi tiap soal dengan rasio 8:2 menggunakan fungsi *train_test_split* dari sklearn.

TABEL 2
SOAL DAN KUNCI JAWABAN DARI SAMPEL SOAL TERPILIH

IDSoal	Soal	Kunci Jawaban
1	Ada empat macam karakter atau olah yang diusulkan oleh Ki Hajar Dewantara untuk membangun kerja sama dan gotong royong. Jelaskan karakter-karakter tersebut?	Olah pikir = Berpikiran kritis dalam segala situasi yang terjadi. Olah hati = Melakukan suatu kegiatan berdasarkan hati nurani kita. Olah raga = Memiliki sifat dan mental yang kuat. Olah hati dan karsa = Kreatif dalam melakukan sesuatu.
8	Dalam pembelajaran agama kelas 8 kita mempelajari dan memahami tantang Sakramen Baptis. Jelaskanlah tahapan-tahapan yang harus dilalui dalam proses Baptis dewasa!	1. Masa Prakatekumenat - Masa pemurnian motivasi calon 2. Masa Katekumenat - masa pengajaran dan pembinaan iman serta latihan hidup dalam jemaat 3. Masa persiapan akhir - Masa khusus untuk mempersiapkan diri menerima sakramen 4. Masa Mistagogi - Masa pembinaan lanjutan setelah seseorang menerima sakramen baptis
9	Jelaskan perbedaan populasi, ekosistem, dan komunitas. Dan berilah masing-masing contohnya	1. populasi adalah kumpulan individu sejenis yang hidup dan berinteraksi di tempat tertentu, contohnya sekumpulan kambing di padang rumput 2. komunitas adalah kumpulan berbagai mahkluk hidup yang berinteraksi dan hidup di area tertentu, contohnya seluruh organisme yang ada di sawah: padi, tikus, belalang 3. ekosistem adalah interaksi antar mahkluk hidup di suatu wilayah dengan lingkungannya yang saling memengaruhi, contohnya ekosistem danau yang terdiri atas organisme dan komponennya
10	Jelaskan tahapan untuk memisahkan suatu campuran yang terdiri serbuk besi, pasir dan garam sehingga didapatkan besi, pasir dan garam yang terpisah!	Tahapan untuk memisahkan campuran serbuk besi, pasir, dan garam: 1. Magnetis: Untuk memisahkan besi dari garam dan pasir. 2. Filtrasi: Pasir dan garam diberi air lalu disaring untuk memisahkan pasir dari garam. 3. Evaporasi: Larutan garam akan dipanaskan sehingga air pada larutan garam menguap dan garam akan mengkristal.
11	Tulislah naskah Sumpah Pemuda yang dikumandangkan oleh para pemuda pada tanggal 28 Oktober 1928!	Kami Putra dan Putri Indonesia, mengaku bertumpah darah yang satu, tanah air Indonesia. Kami Putra dan Putri Indonesia, mengaku berbangsa yang satu, bangsa Indonesia. Kami Putra dan Putri Indonesia, menjunjung bahasa persatuan, bahasa Indonesia.

Untuk penelitian, sistem penilaian akan dibagi ke dalam 4 berkas python, yaitu *processFunction.py, preprocess.py, train.py*, dan *test.py*. Berkas processFunction.py berisi fungsi-fungsi yang diperlukan selama pemrosesan yaitu praproses teks, ekstraksi fitur, dan menggabungkan fitur menjadi satu DataFrame. Fitur yang diekstraksi adalah *unigram cosine similarity*, *bigram cosine similarity*, *word count ratio*, dan *type-token ratio*.

Berkas preprocess.py melakukan praproses untuk data latih dan data uji sekaligus melakukan ekstraksi fitur. Luaran dari berkas preprocess.py adalah berkas berformat csv yang berisi IDSoal dan IDJawaban, serta nilai semua fitur untuk tiap sampel. StemmerFactory dari Sastrawi digunakan untuk melakukan praproses pada jawaban dan kunci jawaban seluruh sampel di data latih dan data uji, yang kemudian diekstraksi fiturnya.Visualisasi algoritma praproses ini dapat dilihat di Gambar 4.

p-ISSN : 2443-2210
e-ISSN : 2443-2229

```
LOAD dataset_csv

FOR jawaban IN dataset_csv DO
    preprocess(jawaban)
END FOR

FOR kunci_jawaban IN dataset_csv DO
    preprocess(kunci_jawaban)
END FOR

unigram_feature ← unigram_cosim(kolom_jawaban, kolom_kunci_jawaban)
bigram_feature ← bigram_cosim(kolom_jawaban, kolom_kunci_jawaban)
wc_ratio_feature ← wc_ratio(kolom_jawaban, kolom_kunci_jawaban)
tt_ratio_feature ← tt_ratio(kolom_jawaban)

fitur_dataset ← features_combine(id_soal, unigram_feature, bigram_feature, wc_ratio_feature, tt_ratio_feature, label)

SAVE fitur_dataset ke file CSV
```

Gambar 4. *Pseudocode* praproses

Fitur *unigram cosine similarity* dan *bigram cosine similarity* memproses jawaban dan kunci jawaban menggunakan fitur ekstraksi teks dari sklearn, yaitu TfIdfVectorizer, dengan parameter ngram_range = (1,1) untuk unigram dan ngram_range = (2,2) untuk bigram. Pada tiap sampel, vektor jawaban dan kunci jawaban akan dihitung nilai kemiripannya dengan fungsi cosine_similarity dari sklearn. Pada Gambar 5 dan Gambar 6, dapat dilihat algoritma proses ekstraksi fitur *unigram cosine similarity* dan *bigram cosine similarity*.

```
FUNCTION unigram_cosim(kolom_jawaban, kolom_kunci_jawaban)
    vectorizer_tfidf_unigram ← TfidfVectorizer WITH ngram_range = (1, 1)
    tfidf_jawab ← vectorizer_tfidf_unigram.fit_transform(kolom_jawaban)
    tfidf_kunci ← vectorizer_tfidf_unigram.fit_transform(kolom_kunci_jawaban)
    n ← jumlah baris tfidf_features_unigram
    FOR i ← 0 TO n - 1 DO
        IF tfidf_features_unigram[i] ≠ vektor nol THEN
            unigram_cosim[i] ← cosine_similarity antara tfidf_features_unigram[i] dan tfidf_kunci_unigram[i]
        ELSE
            unigram_cosim[i] ← 0
        END IF
    END FOR
    RETURN unigram_cosim
END FUNCTION
```

Gambar 5. *Pseudocode* proses ekstraksi fitur *unigram cosine similarity*

Seperti yang ditunjukkan di Gambar 5 dan 6, alur kerja ekstraksi fitur unigram dan bigram *cosine similarity* memiliki kemiripan yang diawali dengan pembobotan *tf-idf* tiap unit fiturnya, dimana unigram menggunakan token, sedangkan bigram menggunakan 2 token yang bertumpang tindih dan berurutan. Satuan unit ini menjadi parameter fungsi *TfidfVectorizer* ([1,1], [2,2]). Tahap selanjutnya adalah menghitung nilai kemiripan *Cosine* antara jawaban siswa dengan kunci jawaban dari sebuah soal. Perhitungan ini diulang sampai seluruh jawaban telah dibandingkan dan mendapatkan nilai persamaan Cosinenya.

```
FUNCTION bigram_cosim(kolom_jawaban, kolom_kunci_jawaban)
    vectorizer_tfidf_bigram ← TfidfVectorizer dengan parameter ngram_range = (2, 2)

    tfidf_features_bigram ← vectorizer_tfidf_bigram.fit_transform(kolom_jawab)
    tfidf_kunci_bigram ← vectorizer_tfidf_bigram.transform(kolom_kunci)

    n ← jumlah baris tfidf_features_bigram
    FOR i ← 0 TO n - 1 DO
        IF tfidf_features_bigram[i] ≠ vektor nol THEN
            bigram_cosim[i] ← cosine_similarity antara tfidf_features_bigram[i] dan tfidf_kunci_bigram[i]
        ELSE
            bigram_cosim[i] ← 0
        END IF
    END FOR

    RETURN bigram_cosim
END FUNCTION
```

Gambar 6. *Pseudocode* proses ekstraksi fitur *bigram cosine similarity*

Fitur *type-token ratio* menghitung rasio kata unik pada jawaban di tiap sampel dengan formula jumlah *type* (kata unik) dibagi jumlah *token* (seluruh kata). Proses ini diulang untuk tiap teks jawaban peserta didik, kemudian hasilnya ditampung di list *tt_ratio*. Pseudocode dari proses perhitungan *type-token ratio* ditampilkan di Gambar 7.

```
FUNCTION tt_ratio(kolom_jawaban)
    tt_ratio ← array kosong
    jawaban_list ← daftar string dari kolom_jawab

    FOR EACH jawaban_siswa IN jawaban_list DO
        token_jawaban ← hasil pemisahan jawaban_siswa berdasarkan spasi
        type_jawaban ← token unik dari token_jawaban

        IF panjang dari type_jawaban = 0 THEN
            ttr ← 0
        ELSE
            ttr ← panjang type_jawaban dibagi panjang token_jawaban
        END IF

        APPEND ttr TO tt_ratio
    END FOR

    RETURN tt_ratio
```

Gambar 7. *Pseudocode* proses ekstraksi fitur *type-token ratio*

Fitur *word count ratio* menghitung rasio jumlah kata pada jawaban dengan batas kata yang diberikan. Dalam kasus ini, jumlah kata pada kunci jawaban menjadi batas kata. Tidak seperti tiga fitur yang lain, *word count ratio* memiliki rentang nilai yang besar, sehingga perlu dilakukan normalisasi, agar rentang nilainya sama dengan fitur-fitur yang lain. Normalisasi ini dilakukan dengan menggunakan fungsi sigmoid. Alasan penggunaan fungsi sigmoid ini karena fitur *word count ratio* tidak selalu memengaruhi kualitas teks secara linear. Fungsi sigmoid mengkomputasi fitur *word count* yang awalnya bertipe integer kemudian ditransmformasikan ke tipe float dengan rentang 0.00 -1.00 sehingga cocok digunakan untuk representasi fitur non-linear. Lebih jelasnya, algoritma proses ekstraksi fitur *word count ratio* dapat dilihat pada Gambar 8.

```
FUNCTION wc_ratio(kolom_jawaban, kolom_kunci_jawaban)
    k ← 5
    wc_ratio ← array kosong
    jawaban_list ← daftar string dari kolom_jawab
    kunci_jawaban ← elemen pertama dari kolom_kunci_jawaban
    word_limit ← jumlah kata dalam kunci_jawaban

    FOR EACH jawaban_siswa IN jawaban_list DO
        word_count ← jumlah kata dalam jawaban_siswa
        raw_wcr ← 1 - (word_count / word_limit)
        nilai_ratio ← 1 / (1 + exp(-k × raw_wcr))
        APPEND nilai_ratio TO wc_ratio
    END FOR

    RETURN wc_ratio
END FUNCTION
```

Gambar 8. *Pseudocode* proses ekstraksi fitur *word count ratio*

Berkas train.py berfungsi untuk melatih model berdasarkan luaran preprocess.py. Model dilatih berulang untuk tiap soal. Untuk menyeimbangkan rasio sampel di tiap kategori nilai, maka dilakukan proses augmentasi data menggunakan *Syntetic Minority Over-sampling technique* (*SMOTE*) sehingga jumlah populasi sampel jawaban di tiap kelas menjadi seimbang. SMOTE dipilih sebagai metode augmentasi karena *SMOTE* telah terbukti sebagai metode augmentasi data yang handal bagi data yang kotor dan tak seimbang [9]. Kemudian proses komputasi dilanjutkan dengan pelatihan dengan 3 model klasifikasi yang telah ditentukan. Pada tiap pasangan soal – jawaban siswa, dilakukan proses validasi *KFold* dengan *5-fold* dan dihitung nilai *precision* dari proses validasi tersebut. Proses validasi dilakukan setelah proses pelatihan. Dari tiap Nilai presisi dari tiap fold validasi disimpan dalam variable tampung untuk perhitungan rata-rata nilai *precision* dari nilai presisi model. Model yang sudah dilatih dibuat menjadi luaran berkas train.py dengan menggunakan pustaka pickle. Algoritma dari proses pelatihan model ini dapat dilihat lebih jelas pada Gambar 9.

```
LOAD fitur_data_latih

model ← inisialisasi model
smote ← inisialisasi smote

id_soal_list ← kumpulan nilai kolom id_soal unik pada fitur_data_latih
precision_list ← array kosong

FOR EACH idsoal IN id_soal_list DO
    subset ← seluruh baris fitur_data_latih yang memiliki nilai kolom id_soal = idsoal
    x ← kolom 'Cosim-Unigram', 'Cosim-Bigram', 'TypeTokenRatio', dan 'WordCountRatio' dari subset
    y ← kolom 'label' dari subset

    (x, y) ← smote(x,y)

    TRAIN model

    kfold ← KFold dengan 5 split, data diacak, dan seed acak = 42

    results_precision ← hasil cross_val_score dari model menggunakan data x dan y, dengan pembagian kfold

    precision_rata ← rata-rata dari results_precision
    tambahkan precision_rata ke dalam precision_list
END FOR

precision_mean ← rata-rata dari seluruh nilai dalam precision_list

PRINT precision_mean

SAVE model ke file
```

Gambar 9. *Pseudocode* proses pelatihan model

Berkas test.py menggunakan berkas model luaran train.py dan data uji luaran berkas preprocess.py untuk proses pengujian model. Luaran berkas ini adalah *confusion matrix* dan nilai metrik *precision*, *recall*, *f1-score*, *accuracy*, serta *Root Mean Square Error* (RMSE). Algoritma proses pengujian model ini dapat dilihat pada Gambar 10. Perulangan dilakukan di tiap Soal dengan mengambil ID soal, dimana fitur yang terbentuk dari tiap jawaban tadi diakses per baris dan disimpan dalam variabel *subset*. Data per baris kemudian disimpan dalam bentuk DataFrame di variable *features.csv* yang kemudian dilakukan prediksi. Hasil prediksi kemudian disimpan dalam bentuk array dan luaran dikembalikan dalam format JSON.

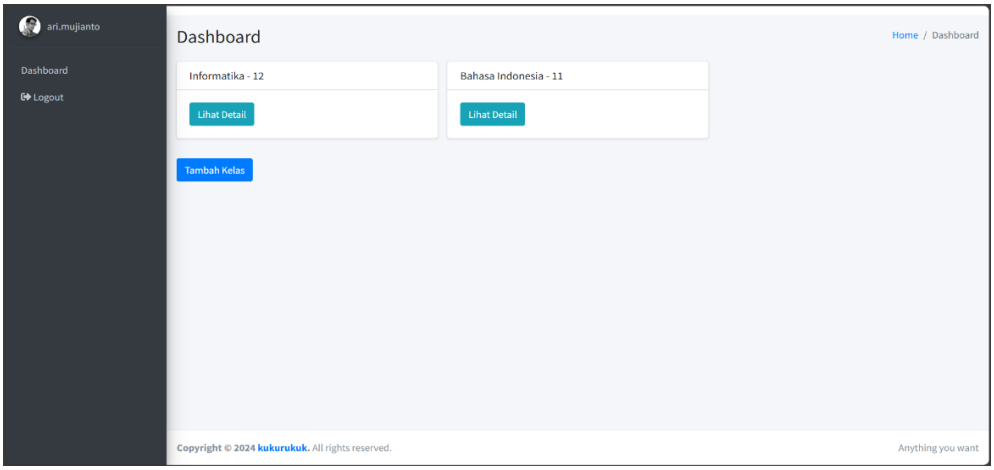
```
file ← file dari request
data ← CONVERT file ke DataFrame
features_data ← preprocess(data)
idsoal_list ← nilai unik dari kolom id_soal dalam features_data

all_pred ← dictionary kosong

FOR EACH idsoal IN idsoal_list:
    subset ← seluruh baris features_data yang memiliki nilai kolom id_soal = idsoal
    features_csv ← fitur 'Cosim-Unigram', 'Cosim-Bigram', 'TypeTokenRatio', dan 'WordCountRatio' dari subset
    pred ← hasil prediksi model terhadap features_csv
    pred_dict ← dictionary yang memetakan id_jawaban ke label prediksi

    APPEND pred_dict ke dalam all_pred
END FOR

RETURN all_pred dalam bentuk JSON
```

Gambar 10. *Pseudocode* proses pengujian model

B. Integrasi Sistem

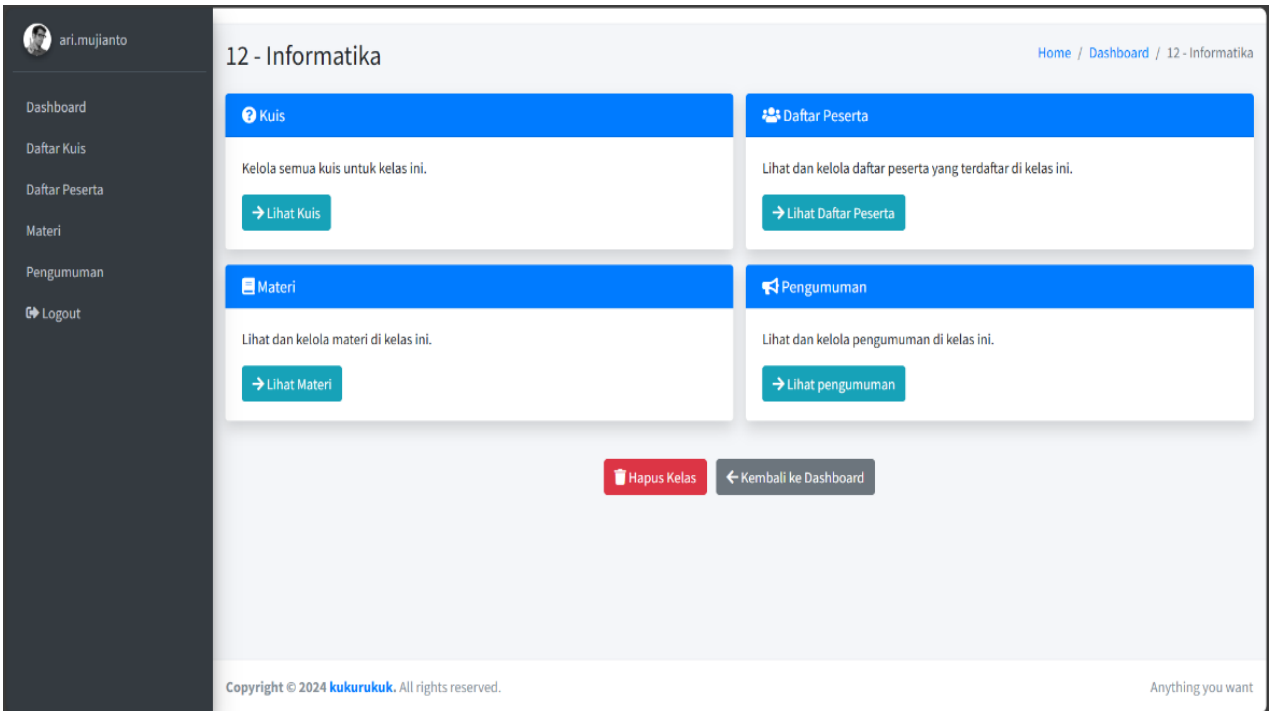
1) Pengembangan Front-End

Berikut adalah beberapa tangkapan layar hasil pengembangan *front-end website* LMS khusus system ASAG. Gambar 11 merupakan tampilan *dashboard* guru, halaman ini berisi daftar kelas yang sudah dibuka oleh guru yang bersangkutan. Untuk melakukan aksi lebih lanjut terhadap kelas tertentu, dilakukan dengan menekan tombol “Lihat Detail” pada kelas yang dipilih. Contoh kelas di Gambar 11 adalah Informatika-12 atau Bahasa Indonesia-11.



Gambar 11. Halaman *dashboard* guru

Setelah memilih detail kelas yang diinginkan, akan ditampilkan halaman kelas, seperti tertampil pada Gambar 12. Di halaman ini terdapat beberapa menu dengan fungsinya masing-masing, yaitu materi, pengumuman, daftar peserta, dan kuis. Pembuatan maupun penambahan kuis, materi, maupun pengumuman dilakukan dengan mengklik tombol "Lihat Kuis", "Lihat Materi", maupun tombol "Lihat Pengumuman". Jika guru memilih dan mengklik tombol Kuis, maka di sana ditampilkan menu untuk memilih Soal dan kunci jawaban, melihat semua jawaban dari soal kuis yang dipilih, maupun untuk menghapus kuis.



Gambar 12. Halaman kelas guru

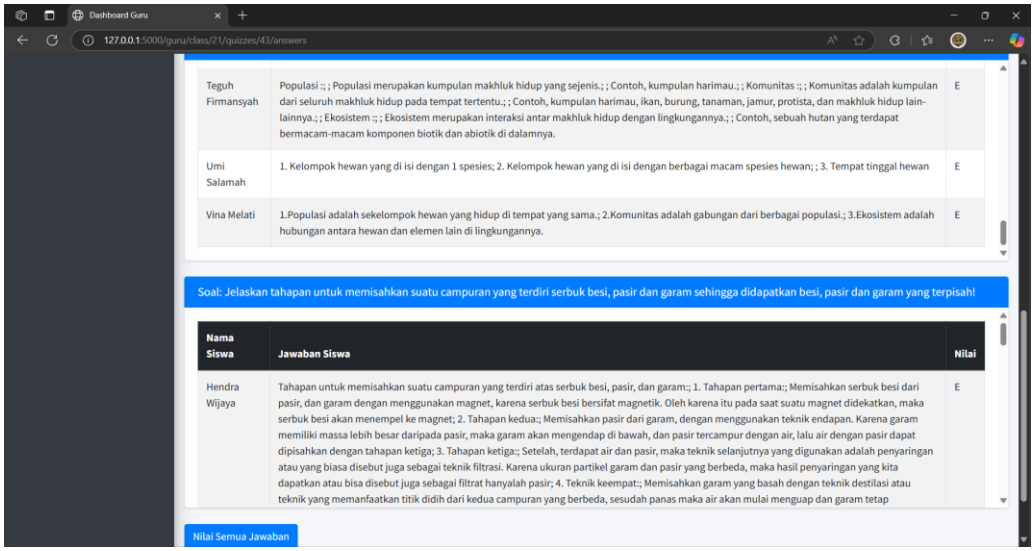
2) Pengembangan Back-End

Model yang sudah dilatih dan dikembangkan kemudian diintegrasikan dengan sistem LMS yang berfungsi sebaga sistem *back-end*. Ketika pengguna guru menekan tombol "Nilai Semua Jawaban", *client* akan melakukan *request* ke *endpoint /evaluate* di *back-end* dengan mengirimkan seluruh jawaban siswa dari kuis terpilih. *Server* di *back-end* lalu mengolah masukan yang diterima, yaitu memprediksi nilai dengan model yang dipilih. Struktur algoritma pada *endpoint /evaluate* dapat

dilihat pada Gambar 13 yang sebenarnya memanggil fungsi di tiap tahapan proses dari pemanggilan fitur, prediksi, perhitungan dengan confusion matrix, menampilkan laporan klasifikasi dari luaran confusion matrix dan memanggil fungsi perhitungan nilai RMS. Setelah itu mencetak hasilnya di layar. Setelah seluruh jawaban diprediksi, nilai dan id jawaban akan dikembalikan dengan format json untuk ditampilkan. Terlihat pada Gambar 14 tampilan ketika tombol "Nilai Semua Jawaban" ditekan.

```
LOAD fitur_data_uji
LOAD file_model
model ← file_model

x ← kolom 'Cosim-Unigram', 'Cosim-Bigram', 'TypeTokenRatio', dan 'WordCountRatio' dari fitur_data_uji
y ← kolom 'label' dari fitur_data_uji

y_pred ← hasil prediksi model terhadap x

cm ← confusion_matrix dari y dan y_pred

classification_report ← classification_report dari y dan y_pred

rmse ← root_mean_squared_error dari y dan y_pred

PRINT cm, classification_report, rmse
```

Gambar 133. *Pseudocode endpoint* /evaluate



Gambar 14. Tampilan halaman "jawaban siswa" setelah dinilai otomatis

C. Pengujian dan Analisis

Untuk mendapatkan hasil pelatihan model yang paling baik, penting untuk mengetahui kombinasi fitur yang paling berpengaruh serta pemilihan kernel model SVM multikelas yang tepat. Untuk itu perlu dilakukan eksperimen untuk menentukan kernel dan kombinasi fitur dengan nilai Presisi tertinggi di hasil validasi untuk diimplementasikan dalam model purwarupa. Hasil validasi setelah pelatihan model SVM dengan kombinasi fitur serta kernel ditampilkan di Tabel 1. Dua Kernel SVM yang dieksperimenkan adalah kernel *Radial Basis Function* (RBF) dan kernel Polinomial. Keempat fitur yang diekstrasi dikombinasikan dengan jumlah 3 dan 4 fitur dan menghasilkan 5 kombinasi fitur. Kelima kombinasi fitur ini bisa dilihat juga di Tabel 3.

TABEL 3
HASIL VALIDASI KOMBINASI FITUR DAN KERNEL

	UC-BC-TTR	UC-BC-WCR	UC-TTR-WCR	BC-TTR-WCR	UC-BC-TTR-WCR
Linear	62.46	60.33	61.64	62.58	61.07

Poly d=2	65.30	67.75	69.60	62.75	68.98
Poly d=3	67.92	67.19	73.42	63.52	71.49
Poly d=4	69.47	69.93	75.02	65.15	74.90
Poly d=5	70.64	71.50	75.74	69.04	76.92
RBF γ=0.1	73.41	68.54	68.92	66.75	67.28
RBF γ=1	60.98	62.62	60.55	63.19	62.14
RBF γ=10	66.77	68.91	70.77	65.28	72.35
RBF γ=100	74.80	74.14	76.40	69.01	**78.14**

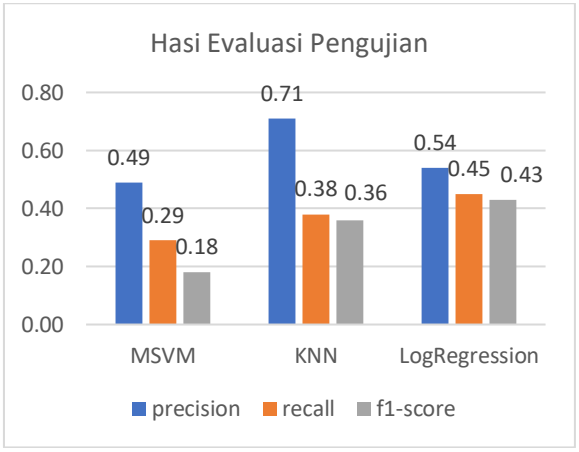
Catatan: ***UC = Unigram Cosine Similarity; BC = Bigram Cosine Similarity; TTR = Type-Token Ratio; WCR = Word Count Ratio; Poly = Polynomial; RBF = Radial Basis Function.***

Dari Tabel 3, dapat diketahui bahwa kombinasi empat fitur yang diimplementasikan dengan kernel RBF dengan nilai γ = 100 memiliki nilai Presisi tertinggi pada proses validasi. Maka, model purwarupa dilatih menggunakan kernel tersebut dan seluruh fitur yang telah diekstraksi. Hasil pengujian purwarupa SVM multikelas dengan data uji ditampilkan di Tabel 4. Dengan data uji yang baru, maka presisi tertinggi yang dicapai adalah 0.49, nilai Akurasi mencapai 0.29 dan nilai RMSE mencapai 2.77.

TABEL 4
NILAI METRIK EVALUASI MODEL SVM MULTIKELAS

	precision	*recall*	*f1-score*	*accuracy*	RMSE
Macro avg	0.39	0.25	0.17	0.29	2.77
Weighted avg	0.49	0.29	0.18		

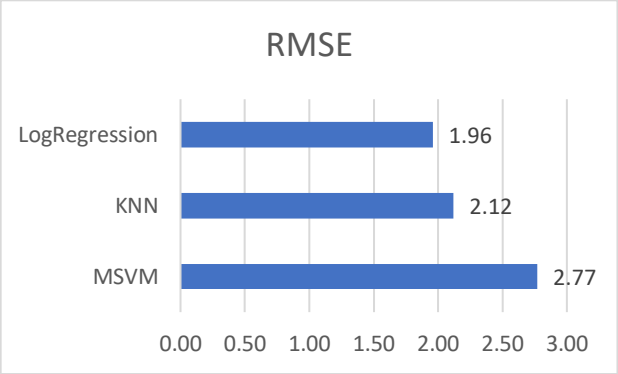
Hasil evaluasi model SVM multikelas yang diperlihatkan Tabel IV mengindikasikan performa model yang kurang baik. Berbeda dengan hasil yang dilaporkan oleh Suhanjoyo dkk [10] yang menggunakan SVM untuk mendeteksi tindak kecurangan, SVM menunjukkan prediksi dengan akurasi tertinggi dibandingkan model lainnya. Keunggulan SVM juga terbukti saat digunakan untuk klasifikasi judul tugas akhir [11]. Sayangnya dalam memprediksi nilai untuk jawaban singkat berbasis bahasa natural, kinerja SVM kurang memuaskan. Hasil tersebut dapat dibandingkan dengan hasil evaluasi model klasifikasi lain, dalam hal ini k-NN dan Regresi Logistik, untuk melihat apakah kedua model tersebut memberikan performa yang lebih baik dalam memprediksi nilai dengan baik. Cara prediksi nilai jawaban dilakukan dengan pendekatan klasifikasi dengan 5 tahapan nilai yang diberi label A, B, C,D dan E. Hasil uji coba purwarupa dengan model SVM Multikelas, k-NN dan Regresi Logistik dengan metrik evaluasi Presisi, Recall dan F1-score yang terboboti ditunjukkan di Gambar 15.



Gambar 15. Perbandingan nilai metrik evaluasi model dengan *weighted average*

Dari Gambar 15, dapat diamati bahwa seluruh nilai metrik evaluasi dari model SVM multikelas lebih rendah dibandingkan kedua model lainnya. Jika didasarkan pada nilai presisi saja, model k-NN merupakan model klasifikasi dengan performa terbaik dengan nilai presisi mencapai 0.7. Namun nilai *recall* dan *f1-score* dari k-NN berada di bawah 0.4. Sekalipun presisi prediksi Regresi Logistik mencapai 0.55, namun nilai Recall dan F1-score diatas 0.4 dan tidak terpaut jauh dari nilai Presisinya. Ini menunjukkan kalau Regresi Logistik memiliki kinerja yang stabil dalam memprediksikan nilai jawaban siswa.

Hasil dari metrik RMSE pada Gambar 16 menampilkan nilai yang kurang memuaskan, dengan nilai RMSE terbaik ada pada model *Regresi Logistik* dengan nilai 1,96. Sedangkan nilai RMSE terburuk diperoleh model SVM dengan nilai RMSE 2.75. Ini berarti banyak sampel dari data uji yang rentang nilainya diprediksi terlalu jauh dari label yang seharusnya.



Gambar 146. Grafik perbandingan nilai RMSE

D. Temuan dan Hasil

Terdapat beberapa temuan dari hasil penelitian yang sudah dilakukan. Untuk dataset yang digunakan, kombinasi fitur *Unigram & Bigram Cosine Similarity*, *Type-Token Ratio*, dan *Word Count Ratio* diimplementasikan bersama dengan model SVM kernel RBF dengan $\gamma = 100$ menghasilkan nilai presisi tertinggi, yaitu 78,14% pada tahap validasi.

Karena sebaran sampel di tiap kelas pada data uji tidak merata, perhitungan metrik evaluasi dengan *weighted average* diutamakan dalam analisis. Terlihat bahwa nilai Presisi pada tahap evaluasi mengalami penurunan bila dibandingkan dengan tahap validasi. Penurunan ini merupakan hal yang wajar karena dalam proses validasi menggunakan 5-Fold Cross Validation, data diuji selama lima kali iterasi, di mana setiap iterasi, sampel secara bergantian berperan sebagai data uji dan data latih. Namun, penurunan presisi yang cukup besar ini memiliki kemungkinan terjadinya *overfitting* pada model SVM multikelas pada proses validasi. Overfitting ini mengindikasikan bahwa model SVM belajar sangat baik dengan data latih, namun kesulitan melakukan generalisasi untuk data uji yang memiliki distribusi sampel kelas yang variatif.

Selain itu, nilai metrik evaluasi lainnya dari model SVM multikelas juga menunjukkan hasil yang kurang optimal. Nilai akurasi dan *recall* menunjukkan nilai yang rendah, sehingga *f1-score* yang sensitif terhadap nilai recall juga otomatis memiliki nilai rendah. Namun, nilai metrik Presisi lebih diperhatikan karena metrik ini mengukur seberapa akurat prediksi positif yang dibuat oleh model.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - y'_i)^2} \qquad (1)$$

Berdasarkan rumus metrik RMSE pada Persamaan 1, rentang nilau RMSE adalah dari 0 sampai tak terbatas (∞) [12], namun tergantung pada skala data dan satuan variabel target yang digunakan dalam evaluasi [13]. Sebagai contoh, bila kelas dataset digunakan adalah 1 hingga 5, maka nilai RMSE terbesar yang bisa didapat adalah 5. Dengan nilai 2,77 pada model SVM multikelas, maka model ini dianggap memiliki tingkat kesalahan prediksi yang tinggi, yakni diatas 50%. Nilai RMSE yang tinggi ini bisa juga disebabkan oleh jarak antar kelas yang cukup jauh, dimana kelas yang digunakan pada penelitian ini *error minimum*-nya adalah 1. Dari temuan ini dapat disimpulkan bahwa model SVM multikelas dengan kernel dan fitur yang digunakan tidak cukup akurat untuk melakukan prediksi pada dataset yang digunakan pada penelitian.

Metrik evaluasi model klasifikasi lain, yaitu k-NN dan *Logistic Regression*, memiliki performa yang lebih baik bila dibandingkan dengan model SVM multikelas. Namun, kedua model tersebut tetap memiliki nilai metrik yang kurang memuaskan. Dari temuan ini disimpulkan bahwa dengan kombinasi fitur yang sama, model klasifikasi lain pun juga belum cukup akurat untuk melakukan prediksi pada dataset yang digunakan. Nilai evaluasi yang kurang baik dari ketiga model ini, memiliki kemungkinan disebabkan oleh model klasifikasi yang kurang tepat untuk diimplementasikan dengan dataset yang digunakan. Fitur yang dipilih juga kemungkinan kurang mewakili label pada sampel, sehingga proses klasifikasi belum bekerja dengan maksimal.

IV. SIMPULAN

Dari penelitian yang sudah dilakukan, sistem penilaian jawaban singkat (ASAG) untuk teks berbahasa Indonesia telah berhasil diintegrasikan pada sistem LMS yang ada. Integrasi ini menunjukkan bahwa penggabungan sistem ASAG dengan LMS dapat dilakukan dengan baik dan memungkinkan penggunaannya secara langsung dalam lingkungan pembelajaran digital untuk mempertahankan sistem pembelajaran yang ringan.

Namun, dari sisi performa, model SVM multikelas, dengan nilai metrik evaluasi presisi sebesar 0,49 dan RMSE sebesar 2,77, menunjukkan hasil yang kurang memuaskan. Lebih lagi, hasil evaluasi model k-NN dan *Logistic Regression* memiliki nilai yang lebih baik dibandingkan dengan model SVM multikelas. Sehingga, dapat disimpulkan bahwa model SVM multikelas kurang cocok untuk diaplikasikan pada sistem penilaian soal uraian otomatis ini.

Berdasarkan hasil penelitian, model yang direkomendasikan untuk digunakan adalah *Logistic Regression*. Meskipun model ini memiliki nilai presisi yang lebih rendah dibandingkan k-NN, *Logistic Regression* menunjukkan nilai RMSE terbaik, yaitu 1,96, serta nilai recall dan f1-score yang lebih tinggi dari k-NN. Oleh karena itu, model ini memberikan prediksi yang lebih stabil dan akurat.

Karena keterbatasan yang ada dalam penelitian ini, disarankan pada penelitian selanjutnya untuk mengembangkan sistem penilaian soal uraian otomatis ini dengan menggunakan berbagai model *machine learning* lainnya, dan bahkan penerapan model *deep learning*. Hal ini perlu dilakukan untuk mengevaluasi sejauh mana performa model-model tersebut dalam memprediksi nilai untuk jawaban uraian singkat dengan mempertahankan karakteristik model yang ringan untuk diaplikasikan ke sistem LMS.

Selain itu, perlu juga dilakukan analisis lebih lanjut terhadap fitur-fitur yang digunakan dalam pelatihan model. Disarankan untuk penelitian lebih lanjut yang menguji berbagai jenis fitur, guna mencari fitur yang memiliki pengaruh yang signifikan terhadap hasil prediksi label.

Disarankan pula untuk penelitian selanjutnya dapat mengeksplorasi berbagai teknik praproses teks yang lebih beragam. Penelitian lanjutan dapat mempertimbangkan teknik normalisasi yang lebih kompleks seperti koreksi ejaan otomatis, penanganan kata tidak baku, serta pengenalan entitas khusus yang mungkin relevan dalam konteks soal. Tujuannya agar dapat ditemukan strategi praproses yang optimal untuk meningkatkan akurasi model penilaian uraian otomatis.

DAFTAR PUSTAKA

[1] L. D. Krisnawati, A. Mahastaman and S. Haw, "Cross-Prompt Based Automatic Short Answer Grading System," *Communication and Information Technology (COMMIT) Journal,* vol. 19, no. 2, pp. 281-291, 2025.

[2] R. A. Rajagede, "Improving automatic essay scoring for Indonesian language using simpler model and richer feature," *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control,* vol. 6, no. 1, pp. 11-18, 2021.

[3] Z. Li, Y. Tomar and R. J. Passonneau, "A Semantic Feature-Wise Transformation Relation Network for Automatic Short Answer Grading," in *Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic, 2021.

[4] A. E. E. Elalfi, A. F. Elgamal and N. A. Amasha, "Automated Essay Scoring using Word2vec and Support Vector Machine," *International Journal of Computer Applications,* vol. 177, no. 25, pp. 20-29, 2019.

[5] Q. I. Susilowati and Z. Y. Acar, "Essay Scoring for Essay Evaluation Using Support Vector Machine in predicting Youth Break the Boundaries Scholarship Applicants," *Journal Elektronik Sistem Informasi,* vol. 1, no. 2, pp. 82-93, 2023.

[6] Munir, L. S. Riza and A. Mulyadi, "An Automatic Scoring System for Essay by Using Methods Combination of Term Frequency and n-Gram," *International Journal of Trend in Research and Development,* vol. 3, no. 6, pp. 403-407, 2016.

[7] F. F. Lubis, Mutaqin, A. Putri, D. Waskita, T. Sulistyaningtyas, A. A. Arman and Y. Rosmansyah, "Automated Short-Answer Grading using Semantic Similarity based on Word Embedding," *International Journal of Technology,* vol. 12, no. 3, pp. 571-581, 2021.

[8] D. S. V. Madala, "An empirical analysis of machine learning models for automated essay grading," *PeerJ Preprints,* 2018.

[9] J. Liu, "Importance-SMOTE: a synthetic minority oversampling method for noisy imbalanced data," *Soft Computing,* vol. 26, no. 3, pp. 1141-1163, 2022.

[10] B. W. Suhanjoyo, H. Toba and B. R. Suteja, "Fraud Detection in Sales of Distribution Companies Using Machine Learning," *JUTISI,* vol. 9, no. 2, pp. 300-312, 2023.

[11] W. M. Dhuhita, M. F. K. A. Darmawan, L. Triana and N. Ankisqiantari, "Comparison of Supervised Learning Algorithm for Classification of Thesis Titles Based on Lecturer Fields," *JUTISI,* vol. 8, no. 2, pp. 427-, 2022.

[12] T. O. Hodson, "Root Mean Square Error (RMSE) or Mean Absolute Error (MAE): When to use them or not," *Geoscientific Model Development Discussions,* vol. 2022, no. 1, pp. 1-10, 2022.

[13] D. Chicco, M. J. Warrens and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation.," *Peerj computer Science,* vol. 7, p. Article e623, 2021.