KLASIFIKASI ULASAN APLIKASI SIREKAP 2024 DENGAN EKSTRAKSI FITUR *WORD2VEC* DAN METODE *SUPPORT VECTOR MACHINE* (SVM)

Muthmainnah, Naufal Azmi Verdikha, Fendy Yulianto
Universitas Muhammadiyah Kalimantan Timur
2111102441170@umkt.ac.id

ABSTRAK

Pemilu Indonesia memanfaatkan teknologi, termasuk aplikasi SIREKAP 2024, untuk meningkatkan transparansi dan efisiensi. Penelitian ini menganalisis ulasan pengguna aplikasi SIREKAP 2024 dari *Google Play Store* dengan pendekatan machine learning. Ekstraksi fitur dilakukan menggunakan *Word2Vec (Skip-gram)*, sementara klasifikasi menggunakan *Support Vector Machine (SVM)*. Data ulasan dikumpulkan melalui teknik scraping dan diproses melalui tahapan praproses serta penyeimbangan data menggunakan *class_weight='balanced'*. Hasil menunjukkan bahwa tanpa penyeimbangan data, model menghasilkan *F1-Score* sebesar 29,02%. Dengan penerapan *class_weight='balanced'*, skor meningkat menjadi 32,05%. Optimasi parameter dengan nilai *C=1* dan *max_iter=65* memberikan *F1-Score* tertinggi sebesar 36,02%, meningkat 7% dari konfigurasi awal. Studi ini menyoroti pentingnya penyeimbangan data dan konfigurasi parameter yang tepat dalam meningkatkan performa model klasifikasi. Namun, model yang digunakan masih belum sepenuhnya sesuai untuk data yang tersedia.

Keyword : *Klasifikasi, Ulasan, SIREKAP, Word2Vec, SVM*

1. PENDAHULUAN

Indonesia menganut sistem demokrasi yang memungkinkan warga negara untuk memilih pemimpin melalui pemilihan umum yang diadakan setiap lima tahun sekali [1].

Integrasi teknologi, seperti aplikasi SIREKAP 2024, meningkatkan proses pemilu dengan meningkatkan transparansi, efisiensi, dan akurasi penghitungan suara [2].

SIREKAP 2024 mempercepat proses penghitungan suara, meminimalisir potensi kecurangan, dan memungkinkan pemantauan secara real-time, meskipun analisis lebih lanjut terhadap umpan balik pengguna diperlukan untuk pengembangan lebih lanjut [3].

Ulasan pengguna di *Google Play Store* mencerminkan pengalaman yang beragam, mulai dari kepuasan hingga keluhan. Melakukan analisis manual terhadap ulasan-ulasan ini memakan waktu, sehingga pendekatan berbasis Pemrosesan Bahasa Alami (*Natural Language Processing*/NLP) sangat penting untuk efisiensi yang lebih besar [4].

Metode ekstraksi fitur seperti *Word2Vec*, *FastText*, dan *GloVe* digunakan, dengan penelitian yang menunjukkan bahwa *Word2Vec* dan *FastText* mencapai kinerja yang sama yaitu 73,15%, sementara *GloVe* mencapai 60,10% [5].

Untuk klasifikasi ulasan, algoritme seperti *Decision Tree*, SVM, dan CNN biasanya digunakan. Studi menunjukkan bahwa SVM mengungguli metode lain, terutama ketika dikombinasikan dengan *TF-IDF* dan *Word2Vec* [6][7].

Selain itu, SVM telah menunjukkan akurasi yang tinggi dalam analisis sentimen data dari *Twitter* dan berbagai aplikasi [8].

Urgensi penelitian ini terletak pada pentingnya menganalisis ulasan pengguna aplikasi SIREKAP 2024. Ulasan tersebut dapat mencerminkan tingkat kepuasan, kendala teknis, serta potensi perbaikan aplikasi. Dengan menggunakan teknik analisis klasifikasi teks, dapat dihasilkan informasi yang berguna bagi pengembang untuk meningkatkan kualitas aplikasi SIREKAP di masa mendatang.

Penelitian ini bertujuan untuk menguji ulasan pengguna terhadap aplikasi SIREKAP 2024 dengan menggunakan *Word2Vec* untuk representasi kata dan SVM untuk klasifikasi. Performa dievaluasi dengan menggunakan *F1-Score* sebagai metrik utama. Temuan dari penelitian ini diharapkan dapat membantu para pengembang dalam meningkatkan kualitas layanan berbasis teknologi melalui analisis ulasan pengguna secara otomatis.

2. TINJAUAN PUSTAKA

2.1. Analisis Klasifkasi Ulasan

Analisis klasifikasi ulasan merupakan teknik dalam pemrosesan bahasa alami (*Natural Language Processing*/NLP) yang digunakan untuk mengelompokkan opini dalam teks berdasarkan karakteristik tertentu [9].

Dalam konteks aplikasi SIREKAP 2024, ulasan pengguna di *Google Play Store* dapat dimanfaatkan untuk mengevaluasi tingkat kepuasan, mengidentifikasi kendala teknis, serta menilai respons terhadap fitur yang disediakan.

2.2. Metode Ekstraksi Fitur Word2Vec

Word2Vec adalah teknik pembelajaran mesin yang menghasilkan representasi kata sebagai vektor, menangkap makna semantiknya dalam sebuah teks. Metode ini terdiri dari dua model utama, *Continuous Bag-of-Words* (CBOW) dan *Skip-Gram*, yang menganalisis hubungan kata dalam korpus dan mengubahnya menjadi representasi vektor. Vektor

yang dihasilkan digunakan sebagai fitur dalam berbagai aplikasi NLP dan pembelajaran mesin [10].

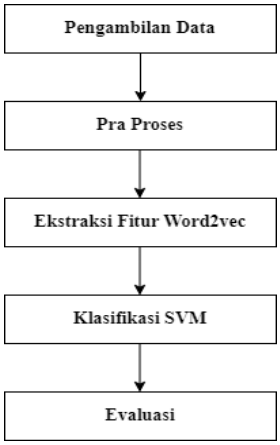
2.3. Algoritma Klasifikasi *Support Vector Machine* (SVM)

Support Vector Machine (SVM) adalah algoritma klasifikasi yang menentukan hyperplane optimal untuk memisahkan data ke dalam berbagai kelas [11].

Algoritma ini banyak diterapkan dalam penelitian klasifikasi teks karena kemampuannya dalam menangani data dengan dimensi tinggi serta mengurangi risiko overfitting [12].

3. METODE PENELITIAN

Penelitian ini dilakukan melalui beberapa tahapan, yaitu pengumpulan data, pembersihan data, pengujian, serta analisis hasil pengujian, yang dapat dilihat pada Gambar 1.



Gambar 1. Alur Penelitian

3.1. Pengambilan Data

Data dikumpulkan menggunakan teknik *scraping* untuk mengambil komentar mengenai aplikasi SIREKAP 2024 dari *Google Play Store*. Ulasan aplikasi ini dipilih karena mencakup berbagai tanggapan pengguna, mulai dari apresiasi terhadap inovasi hingga kritik terhadap kinerjanya. Proses pengumpulan data dilakukan pada 6 Februari 2024, pukul 22.00 WITA, dengan total 8.538 ulasan dan rata-rata penilaian 2,7 dari 5. Data yang diambil dapat dilihat pada Gambar 2.

	userName	score	at	content	thumbsUpCount
6671	Nency Miranda	5	2024-02-05 14:07:12	Mantap	0
5783	ichal sabiel	1	2024-02-05 14:06:17	Sudah tau pemakainya masyarakat biasa Malah sp...	0
7909	Evandra Aditya	1	2024-02-05 14:06:16	Apk nya nggk bisa buat log in	0
6307	Ganesh insann	1	2024-02-05 14:05:38	Susah masuk dih	0
8008	Yohana Frediana	1	2024-02-05 14:04:32	Tidak bisa masuk inisiliasi	0

Gambar 2. Hasil Pengambilan Data

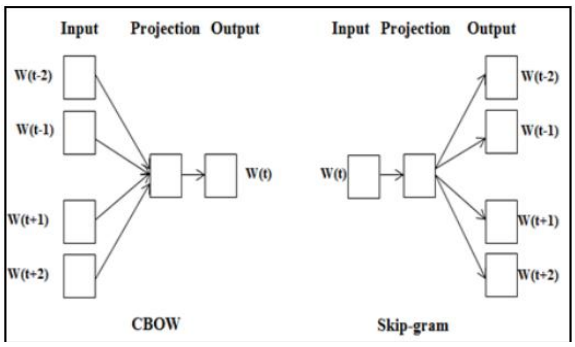
Berdasarkan Gambar 2 hasil dari data yang dikumpulkan dari ulasan pengguna aplikasi SIREKAP mencakup beberapa informasi penting, yaitu *userName*, yang menunjukkan nama pengguna yang memberikan ulasan, serta *score*, yaitu skor atau rating yang diberikan pengguna dalam skala 1-5. Selain itu, terdapat *at*, yang mencatat tanggal dan waktu ulasan diberikan, serta *content*, yang berisi teks ulasan dari pengguna mengenai pengalaman mereka terhadap aplikasi. Terakhir, *thumbsUpCount* menunjukkan jumlah suka (*likes*) yang diterima oleh ulasan tersebut dari pengguna lain, yang dapat mencerminkan tingkat relevansi atau kesetujuan terhadap ulasan yang diberikan.

3.2. Pra Proses

Pada tahap selanjutnya, semua komentar yang terkumpul menjalani tahap pra-pemrosesan, yang melibatkan langkah-langkah seperti mengubah teks menjadi huruf kecil, menghapus karakter yang tidak perlu, pemeriksaan ejaan, dan stemming. Selama proses ini, analisis data dilakukan untuk mendapatkan pemahaman yang lebih mendalam tentang karakteristik ulasan yang diperoleh. Berbagai analisis statistik dan teknik visualisasi diterapkan untuk memeriksa aspek-aspek seperti distribusi peringkat, panjang komentar, dan korelasi antara peringkat dan panjang komentar. Temuan dari analisis ini diharapkan dapat memberikan wawasan yang berharga mengenai pola dan karakteristik ulasan di aplikasi SIREKAP 2024.

3.3. Ekstraksi Fitur *Word2Vec*

Word2Vec memiliki dua arsitektur utama, yaitu *Continuous Bag-of-Words* (CBOW) dan *Skip-Gram*. Model *Word2Vec* diilustrasikan pada Gambar 3.



Gambar 3. Model *Word2Vec*

Perbedaan antara CBOW dan *Skip-gram* terletak pada pendekatan mereka terhadap prediksi kata. CBOW memprediksi kata target dengan mempertimbangkan konteks kata-kata di sekitarnya, sedangkan *Skip-gram* memprediksi konteks (sekumpulan kata-kata di sekitarnya) dari satu kata target [13].

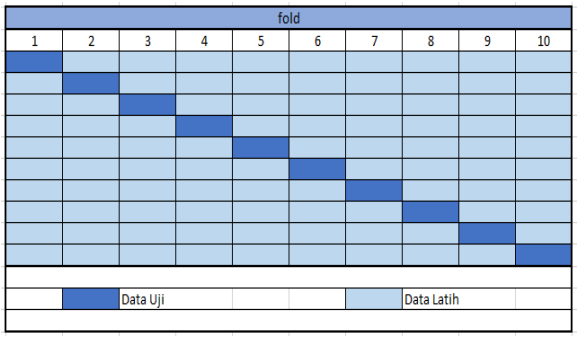
Penelitian ini menggunakan parameter *Skip-gram*, yang menurut [14] menghasilkan *F1-score* yang lebih tinggi yaitu 61,1%, berbeda dengan CBOW yang mencapai F1-score 55,3%.

3.4. Klasifikasi SVM

Setelah langkah ekstraksi fitur, data mengalami validasi menggunakan *Cross Validation*, sebuah teknik yang digunakan untuk menilai kinerja model

dengan membagi data ke dalam set pelatihan dan pengujian. Metode yang umum digunakan adalah *K-fold Cross Validation*, yang efektif dalam menghasilkan model yang tidak bias [15].

Konsep *Cross Validation* dengan *k=10* diilustrasikan pada Gambar 4.



Gambar 4. lustrasi *Cross Validation*

Cross validation memastikan bahwa setiap data digunakan untuk pelatihan dan validasi, yang menghasilkan penilaian yang lebih akurat [16] .

Untuk klasifikasi, regresi, dan prediksi, algoritma *Support Vector Machines* (SVM) memisahkan data menjadi kelas-kelas dengan menggunakan *hyperplane* untuk memaksimalkan jarak antar kelas [17].

3.5. Evaluasi

Evaluasi kinerja model klasifikasi, seperti SVM, dilakukan untuk memastikan model tersebut dapat membuat prediksi yang akurat. Evaluasi ini menggunakan *Confusion Matrix* untuk kasus multi-kelas, yang membantu dalam menghitung *Precision*, *Recall*, dan *F1-Score*. *F1-Score*, rata-rata harmonik dari *precision* dan *recall*, memiliki nilai maksimum 1 (performa terbaik) dan minimum 0 (performa buruk) [18].

Parameter utama untuk evaluasi adalah *y_test* (label aktual) dan *y_pred* (prediksi model). Matriks kebingungan mengklasifikasikan hasil prediksi ke dalam empat kategori: *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) [19].

Penjelasan lebih rinci dari setiap elemen dalam *Confusion Matrix* adalah sebagai berikut:

1. *True Positive* (TP): Kelas positif yang diprediksi sebagai positif.
2. *False Positive* (FP): Kelas negatif yang diprediksi sebagai positif.
3. *True Negatif* (TN): Kelas negatif diprediksi sebagai negatif.
4. *False Negative* (FN): Kelas positif yang diprediksi sebagai negatif.

Evaluasi model dilakukan menggunakan *Confusion Matrix*, dengan perhitungan *Precision*, *Recall* dan *F1-Score* sebagai berikut:

$$precision = \frac{TP}{(TP + FP)}$$

$$Recall = \frac{TP}{(TP + FN)}$$

$$F1\ score = 2\ x\ \frac{Precision\ x\ Recall}{Precision + Recall}$$

Dalam mengevaluasi kinerja model klasifikasi, metrik *Precision* dan *Recall* sangat penting. *Precision* menilai keakuratan prediksi positif model, yaitu proporsi prediksi positif yang benar dari semua prediksi positif. *Recall*, di sisi lain, mengukur seberapa baik model mendeteksi semua data positif, yaitu proporsi data positif yang diidentifikasi dengan benar dibandingkan dengan total data positif yang sebenarnya. Kedua metrik ini saling melengkapi dalam menilai kinerja model, terutama dalam kasus-kasus di mana penting untuk menyeimbangkan antara prediksi yang benar dan prediksi yang meleset.

3.6. Optimasi Parameter

Untuk meningkatkan kinerja model klasifikasi dalam penelitian ini, parameter dioptimalkan. Dalam proses optimasi, parameter class_weight ditambahkan, nilai K ditetapkan untuk validasi cross-K-fold, dan penyesuaian parameter utama algoritma *Support Vector Machine* (SVM), yaitu *class_weight*, *C*, dan *max_iter*. Parameter yang diuji selama proses optimasi dapat dilihat pada Tabel 1.

Tabel 1. Optimasi Parameter

No	Parameter	Nilai
1	*Class_weight*	*Auto/Balanced*
2	*K-Fold*	*K* = 3, 5, 7, 10, 12
3	*C*	*C* = 1, 5, 7, 13, 15, 19, 20, 25, 28, 32, 35, 40, 45, 48, 50
4	*Max_iter*	*Max_iter* = 20, 35, 43, 65, 70, 90, 100, 170, 200, 225, 300, 500, 700, 777, 1000

Peneliti melakukan eksperimen untuk mendapatkan *F1-score* paling tinggi, seperti yang ditunjukkan dalam Tabel 2.5. Untuk menangani ketidakseimbangan data, parameter *class_weight* harus diubah dengan nilai *balanced* atau *auto*. Selanjutnya, data dibagi menggunakan metode validasi cross-fold K dengan nilai K yang berbeda, yaitu *K = 3, 5, 7, 10*, dan *12*. Kemudian dilakukan pencarian parameter *C* (kompleksitas), dengan nilai *C = 1, 5, 7, 13, 15, 19, 20, 25, 28, 32, 35, 40, 45, 48, 50.* Terakhir, dicari nilai max_iter (jumlah maksimum iterasi) menggunakan pengujian beberapa nilai, yaitu *max_iter = 20, 35, 43, 65, 70, 90, 100, 170, 200, 225, 300, 500, 700, 777, 1000.* Tujuan dari penelitian ini adalah untuk menemukan kombinasi parameter yang paling efektif untuk menghasilkan performa model terbaik dengan *F1-Score* tertinggi.

4. HASIL DAN PEMBAHASAN

4.1. Hasil

Hasil menujukan bahwa model SVM dengan parameter *C=10* dan *Max_iter=50* tanpa penerapan *Class_weight='balanced'* menghasilkan *F1-Score* sebesar 29,02%. Setelah optimasi parameter dengan parameter *C=1* dan *Max_iter=65*, *F1-Score* meningkat menjadi 36,02%, menunjukan peningkatan sebesar 7%.

4.2. Optimasi *K-Fold*

Proses *Cross Validation* menggunakan metode *K-Fold* dengan *10 folds* dilakukan. Data dibagi menjadi data latihan (training) dan data uji (test), yang diambil dari kolom "komentar_stemming" dan "rating" pada titik ini. Jumlah lipatan (10) dalam pembagian data dihitung dengan menggunakan parameter *n_splits*. Dengan demikian, setiap lipatan memiliki data yang seimbang untuk pelatihan dan pengujian model. Hasil proses *Cross Validation* pada *fold 1* dapat dilihat pada Gambar 5.

```
Fold 1:
  - Train data: 7468 samples
  - Test data: 830 samples
  - Train Index: [ 830  831  832 ... 8295 8296 8297]
  - Test Index: [  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
 18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
 36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53
 54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71
 72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89
 90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107
108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125
126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161
162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179
180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197
198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233
234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251
252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269
270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287
288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305
306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323
324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341
342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359
360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377
...
 8253 8254 8255 8256 8257 8258 8259 8260 8261 8262 8263 8264 8265 8266
 8267 8268 8269 8270 8271 8272 8273 8274 8275 8276 8277 8278 8279 8280
 8281 8282 8283 8284 8285 8286 8287 8288 8289 8290 8291 8292 8293 8294
 8295 8296 8297]
```

Gambar 5. Nilai *fold 1*

Data latih yang digunakan untuk melatih model terdiri dari 7468 sampel dengan indeks 830–8297, dan data uji yang digunakan untuk menguji model terdiri dari 830 sampel dengan indeks 0–8297. Hasil dari fold 10 dapat dilihat pada Gambar 5.

```
Fold 10:
  - Train data: 7469 samples
  - Test data: 829 samples
  - Train Index: [   0    1    2 ... 7466 7467 7468]
  - Test Index: [7469 7470 7471 7472 7473 7474 7475 7476 7477 7478 7479 7480 7481 7482
 7483 7484 7485 7486 7487 7488 7489 7490 7491 7492 7493 7494 7495 7496
 7497 7498 7499 7500 7501 7502 7503 7504 7505 7506 7507 7508 7509 7510
 7511 7512 7513 7514 7515 7516 7517 7518 7519 7520 7521 7522 7523 7524
 7525 7526 7527 7528 7529 7530 7531 7532 7533 7534 7535 7536 7537 7538
 7539 7540 7541 7542 7543 7544 7545 7546 7547 7548 7549 7550 7551 7552
 7553 7554 7555 7556 7557 7558 7559 7560 7561 7562 7563 7564 7565 7566
 7567 7568 7569 7570 7571 7572 7573 7574 7575 7576 7577 7578 7579 7580
 7581 7582 7583 7584 7585 7586 7587 7588 7589 7590 7591 7592 7593 7594
 7595 7596 7597 7598 7599 7600 7601 7602 7603 7604 7605 7606 7607 7608
 7609 7610 7611 7612 7613 7614 7615 7616 7617 7618 7619 7620 7621 7622
 7623 7624 7625 7626 7627 7628 7629 7630 7631 7632 7633 7634 7635 7636
 7637 7638 7639 7640 7641 7642 7643 7644 7645 7646 7647 7648 7649 7650
 7651 7652 7653 7654 7655 7656 7657 7658 7659 7660 7661 7662 7663 7664
 7665 7666 7667 7668 7669 7670 7671 7672 7673 7674 7675 7676 7677 7678
 7679 7680 7681 7682 7683 7684 7685 7686 7687 7688 7689 7690 7691 7692
 7693 7694 7695 7696 7697 7698 7699 7700 7701 7702 7703 7704 7705 7706
 7707 7708 7709 7710 7711 7712 7713 7714 7715 7716 7717 7718 7719 7720
 7721 7722 7723 7724 7725 7726 7727 7728 7729 7730 7731 7732 7733 7734
 7735 7736 7737 7738 7739 7740 7741 7742 7743 7744 7745 7746 7747 7748
 7749 7750 7751 7752 7753 7754 7755 7756 7757 7758 7759 7760 7761 7762
 7763 7764 7765 7766 7767 7768 7769 7770 7771 7772 7773 7774 7775 7776
 7777 7778 7779 7780 7781 7782 7783 7784 7785 7786 7787 7788 7789 7790
 7791 7792 7793 7794 7795 7796 7797 7798 7799 7800 7801 7802 7803 7804
 7805 7806 7807 7808 7809 7810 7811 7812 7813 7814 7815 7816 7817 7818
 7819 7820 7821 7822 7823 7824 7825 7826 7827 7828 7829 7830 7831 7832
```

Gambar 5. Nilai Fold 10

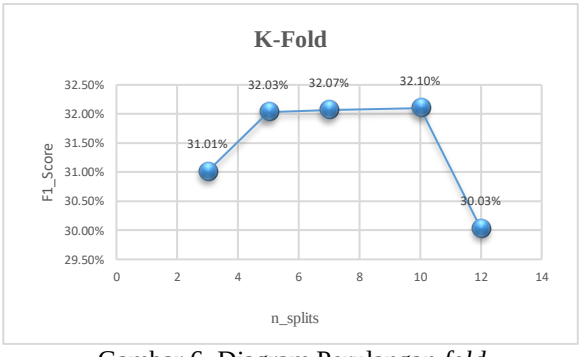
Berdasarkan Gambar 5, Data latihan terdiri dari 7469 sampel dengan indeks mulai dari 0 hingga 7468, sedangkan data uji terdiri dari 829 sampel dengan indeks mulai dari 7469 hingga 8297.

Setelah tahap *Cross Validation*, algoritma *Support Vector Machine* (SVM) digunakan untuk melanjutkan proses ke tahap klasifikasi. Untuk menilai kinerja model secara keseluruhan dan menghitung metrik evaluasi, Model *LinearSVC* digunakan. Proses ini menghasilkan evaluasi untuk setiap *fold*, yang menunjukkan performa model berdasarkan metrik seperti *Precision*, *Recall*, dan *F1-Score*. Hasil evaluasi untuk setiap *fold* disajikan pada Tabel 2.

Tabel 2. Hasil *fi-score* setiap *fold*

K-Fold	***Precision***	***Recall***	***F1-Score***
Fold 1	38,35%	36,84%	36,91%
Fold 2	36,09%	35,05%	34,67%
Fold 3	36,79%	33,39%	34,23%
Fold 4	38,20%	35,64%	36,36%
Fold 5	38,55%	37,44%	37,43%
Fold 6	38,05%	37,41%	36,93%
Fold 7	36,26%	35,81%	35,76%
Fold 8	38,23%	35,71%	36,18%
Fold 9	36,55%	34,61%	34,33%
Fold 10	34,09%	32,64%	32,64%

Para peneliti mengoptimalkan jumlah *fold* dengan mengevaluasi kinerja model di berbagai konfigurasi *fold*, termasuk 3, 5, 7, 10, dan 12 *fold*. Proses ini menggunakan seluruh dataset untuk menentukan konfigurasi yang menghasilkan akurasi terbaik. Berdasarkan analisis *10 fold*, ditemukan bahwa konfigurasi ini memberikan kinerja paling optimal dibandingkan dengan yang lain, yang mengarah pada pemilihan *10 fold* untuk Validasi Silang. Temuan dari analisis jumlah *fold* ini disajikan dalam sebuah diagram, yang membandingkan kinerja model untuk setiap konfigurasi *fold* yang diuji, seperti yang ditunjukkan pada Gambar 6.



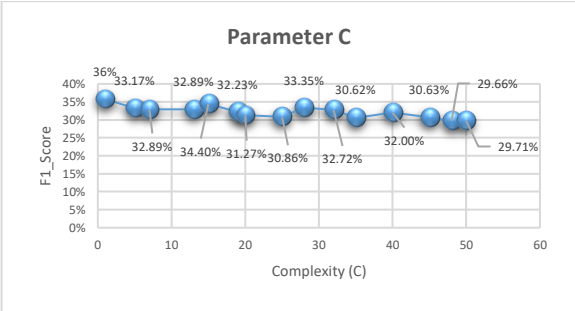
Gambar 6. Diagram Perulangan *fold*

Berdasarkan Gambar 6, analisis jumlah *fold* yang berbeda (3, 5, 7, 10, dan 12 *fold*) menunjukkan bahwa *10 fold* menghasilkan nilai tertinggi yaitu 32,10%. Hal ini menegaskan bahwa *10 folds* memberikan performa terbaik dalam evaluasi model jika dibandingkan dengan jumlah *folds* lainnya. Hasilnya, penelitian ini memilih metode validasi 10 kali lipat karena metode

ini mendistribusikan data secara lebih merata, mengurangi variasi hasil evaluasi, dan meningkatkan akurasi prediksi model. Pilihan ini didasarkan pada analisis yang menunjukkan bahwa 10 kali lipat memberikan hasil evaluasi yang lebih konsisten dan optimal.

4.3. Optimasi Parameter *C*

Selain itu, analisis terhadap parameter *linear* pada *Support Vector Classifier* (SVC) menunjukkan bahwa parameter *C* dan *max_iter* secara signifikan mempengaruhi kinerja model. Dalam penelitian ini, menggunakan *C=10* dengan *max_iterasi = 50* dan tanpa fitur *class_weight = 'balanced'* hanya menghasilkan *F1-Score* sebesar 29,02%. Namun, setelah memperkenalkan fitur *class_weight = 'balanced'*, akurasi meningkat menjadi 32,05%. Hal ini menyoroti pentingnya fitur *class_weight='balanced'* dalam meningkatkan kinerja model dengan memastikan distribusi data yang lebih seimbang, yang memungkinkan model untuk mempelajari pola secara lebih efektif. Selain parameter *C=10*, penelitian ini juga menilai efek dari modifikasi nilai parameter *C* untuk menentukan konfigurasi yang optimal. Diagram yang menggambarkan hasil analisis perubahan parameter *C* dapat dilihat pada Gambar 7.

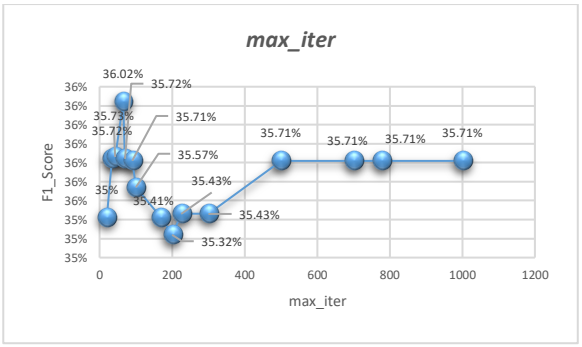


Gambar 7. Modifikasi Parameter *C*

Menurut analisis yang dilakukan, seperti yang ditunjukkan pada Gambar 7, nilai *F1-Score* tertinggi sebesar 36% dicapai ketika *C=1* dipilih. Temuan ini menunjukkan bahwa, dibandingkan dengan nilai parameter *C* lainnya, pilihan *C=1* memberikan keseimbangan optimal antara regularisasi dan kompleksitas model, yang menghasilkan performa yang lebih baik. Ini menunjukkan betapa pentingnya penyesuaian parameter *C* secara menyeluruh untuk memaksimalkan kinerja model.

4.4. Optimasi Parameter *Max_iter*

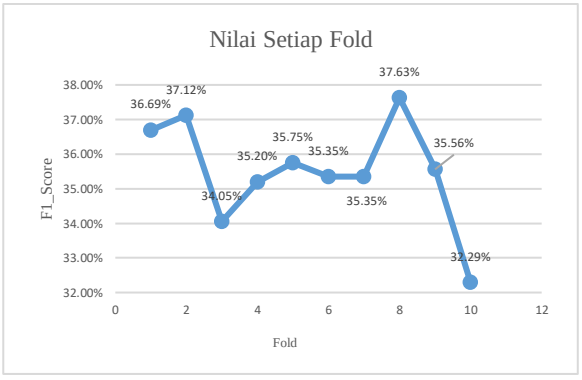
Peneliti juga melihat bagaimana perubahan parameter *max_iter* berdampak pada kinerja model. Percobaan parameter *Max_iter* dapat dilihat pada Gambar 8.



Gambar 8. Modifikasi Parameter *max_iter*

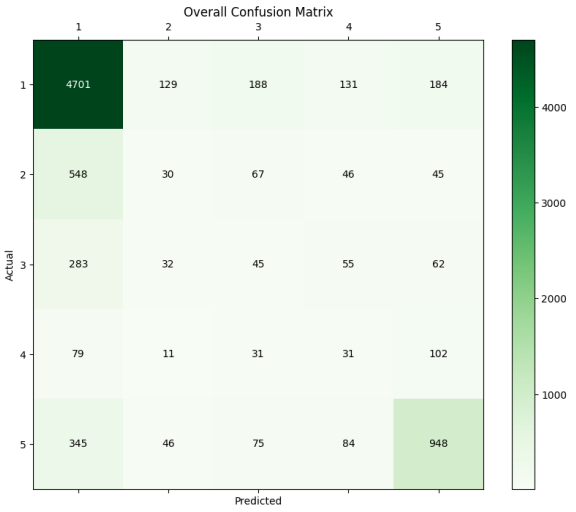
Berdasarkan Gambar 8, analisis menunjukkan bahwa nilai optimal untuk *max_iter* dicapai pada *max_iter = 65* dengan *F1-Score* sebesar 36.02%. Hal ini menunjukkan bahwa meningkatkan jumlah iterasi maksimum hingga batas tertentu memungkinkan model untuk mempelajari pola data dengan lebih baik, sehingga menghasilkan kinerja yang lebih baik. Diagram yang menunjukkan hubungan antara parameter *max_iter* dan performa model menggarisbawahi pentingnya memilih nilai *max_iter* yang tepat untuk meningkatkan performa model.

Dengan mengkombinasikan parameter *C=1* dan *max_iter=65*, model mencapai kinerja yang lebih optimal, memperkuat efektivitas pendekatan yang digunakan dalam penelitian ini. Selain itu, penggunaan validasi silang *K-Fold* sangat penting untuk menilai kemampuan generalisasi model secara komprehensif. Diagram yang relevan untuk *10 fold* dapat dilihat pada Gambar 9.



Gambar 9. Diagram Hasil *f1-score* setiap *fold*

Gambar 9 menampilkan hasil evaluasi keseluruhan untuk setiap *fold*. *Fold* 8 menunjukkan hasil evaluasi tertinggi dengan nilai 37,63%, sedangkan *fold 10* mencatatkan hasil evaluasi terendah sebesar 32,93%. Gambaran konfusi matrik secara keseluruhan dapat dilihat pada Gambar 10.



Gambar 10. Konfusi Metrik

Gambar 10 menunjukkan hasil klasifikasi tertinggi, disorot dengan warna hijau tua, di mana 4.701 sampel uji dengan rating 1 diklasifikasikan secara akurat ke dalam rating 1. Di sisi lain, hasil klasifikasi terendah ditunjukkan dengan warna hijau muda, di mana hanya 30 sampel uji dengan rating 2 yang diklasifikasikan dengan benar ke dalam rating 2.

Untuk rating 1, 4.701 sampel uji diklasifikasikan dengan benar. Untuk rating 2, 30 sampel uji diklasifikasikan secara akurat dari total jumlah data dalam rating tersebut. Pada peringkat 3, 45 sampel uji diklasifikasikan dengan benar dari total data untuk peringkat 3. Untuk rating 4, 31 sampel uji diklasifikasikan dengan benar dari total data pada rating 4. Terakhir, untuk rating 5, 948 sampel uji diklasifikasikan dengan benar dari seluruh data pada rating 5. Klasifikasi, berdasarkan confusion matrix, menghasilkan evaluasi F1-Score sebesar 36,02%.

Penelitian ini memanfaatkan ulasan dari aplikasi SIREKAP 2024 di *Google Play Store*, yang dikumpulkan dengan teknik *Web Scraping*. Data yang terkumpul kemudian melalui tahap pra-pemrosesan untuk menghilangkan elemen-elemen yang tidak relevan, termasuk mengubah teks menjadi huruf kecil, menghilangkan karakter yang tidak perlu, melakukan pengecekan ejaan, dan menerapkan stemming dengan pustaka Sastrawi. Setelah dibersihkan, data ditransformasikan ke dalam representasi vektor menggunakan model *Word2Vec*.

Selanjutnya, ekstraksi fitur dilakukan dengan menggunakan *Word2Vec* untuk mencapai representasi vektor yang optimal. Data yang telah diekstraksi dibagi menggunakan Metode *K-Fold Cross Validation* dengan *10 fold*, dilanjutkan dengan proses klasifikasi menggunakan metode *Support Vector Machine* (SVM). Model SVM dengan parameter *C=10* dan *max_iter=50*, tanpa menggunakan *class_weight='balanced'*, menghasilkan *F1-Score* sebesar 29,02%. Namun, dengan menggunakan *class_weight = 'balanced'*, *F1-Score* meningkat menjadi 32.05%.

Percobaan lebih lanjut dilakukan dengan mengoptimalkan parameter *C* dan jumlah iterasi maksimum untuk meningkatkan performa model. Hasil terbaik dicapai dengan *C = 1* dan *max_iter = 65*, menghasilkan *F1-Score* sebesar 36,02%, meningkat 7% dari hasil sebelumnya. Optimasi ini menyoroti pentingnya menerapkan *class_weight = 'balanced'* dan mengkonfigurasi parameter dengan benar untuk meningkatkan kinerja model klasifikasi secara keseluruhan. Dapat disimpulkan bahwa model yang digunakan dalam penelitian ini tidak cocok untuk data yang ada.

5. KESIMPULAN DAN SARAN

Penelitian ini menyimpulkan bahwa dengan menggunakan parameter *C=1, max_iter=65*, dan fitur *class_weight='balanced'* dapat meningkatkan performa model, dengan mencapai *F1-Score* sebesar 36,02%, meningkat 7% dari hasil sebelumnya dengan *C=10, max_iter=50,* dan tanpa *class_weight='balanced'*, yang hanya menghasilkan *F1-Score* sebesar 29,02%. Namun, hasil tersebut mengindikasikan bahwa model yang digunakan dalam penelitian ini tidak sepenuhnya sesuai dengan data yang ada. Untuk penelitian selanjutnya, disarankan untuk meningkatkan kualitas data dengan membersihkan data secara lebih menyeluruh atau menggunakan dataset yang berbeda dengan karakteristik yang lebih sesuai. Selain itu, disarankan untuk meningkatkan kemampuan model dalam membedakan pola kelas melalui teknik rekayasa fitur atau pemilihan fitur yang lebih relevan. Bereksperimen dengan model lain, seperti *Random Forest, Gradient Boosting,* atau *Deep Learning,* juga dapat bermanfaat untuk menangani pola data yang lebih kompleks. Selain itu, menganalisis penyebab variasi yang signifikan di antara lipatan dengan memeriksa distribusi data di setiap lipatan akan memastikan pembagian data yang lebih proporsional. Terakhir, mengganti dataset dengan dataset yang lebih sesuai dengan karakteristik model dapat meningkatkan kinerja klasifikasi secara keseluruhan.

DAFTAR PUSTAKA

[1] Juri and Rahmad Sugianto, “ANANILIS PERILAKU PEMILIH DI KELURAHAN KEDABANG KECAMATAN SINTANG PADA PEMILIHAN UMUM PRESIDEN DAN WAKIL PRESIDEN TAHUN 2019,” vol. 5, no. 2, 2020.

[2] M. F. Y. Herjanto and C. Carudin, “ANALISIS SENTIMEN ULASAN PENGGUNA APLIKASI SIREKAP PADA PLAY STORE MENGGUNAKAN ALGORITMA RANDOM FOREST CLASSIFER,” *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 2, Apr. 2024, doi: 10.23960/jitet.v12i2.4192.

[3] A. Rahman Hakim, W. Gata, A. Zevana Putri Widodo, O. Kurniawan, and A. Rama Syarif, “Analisis Perbandingan Algoritma Machine

Learning Terhadap Sentimen Analis Pemindahan Ibu Kota Negara," *Jurnal Teknologi Informasi dan Komunikasi)*, vol. 7, no. 2, 2023, doi: 10.35870/jti.

[4] M. A. Riza and N. Charibaldi, "Emotion Detection in Twitter Social Media Using Long Short-Term Memory (LSTM) and Fast Text," *International Journal of Artificial Intelligence & Robotics (IJAIR)*, vol. 3, no. 1, pp. 15–26, May 2021, doi: 10.25139/ijair.v3i1.3827.

[5] W. Kurnia Sari, D. Palupi Rini, R. Firsandaya Malik, and I. B. Saladin Azhar, "Terakreditasi SINTA Peringkat 2 Klasifikasi Teks Multilabel pada Artikel Berita Menggunakan Long Short-Term Memory dengan Word2Vec," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)* , vol. 4, no. 2, pp. 276–285, 2020.

[6] M. F. Naufal, "ANALISIS PERBANDINGAN ALGORITMA SVM, KNN, DAN CNN UNTUK KLASIFIKASI CITRA CUACA," *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, vol. 8, no. 2, pp. 311–318, 2021, doi: 10.25126/jtiik.202184553.

[7] M. F. Asshiddiqi and K. M. Lhaksmana, "Perbandingan Metode Decision Tree dan Support Vector Machine untuk Analisis Sentimen pada Instagram Mengenai Kinerja PSSI," 2020.

[8] A. Saputra, M. Subing, and R. Pratama, "Perbandingan Metode Naïve Bayes Classifier Dan Support Vector Machine Untuk Analisis Sentimen Pengguna Twitter Mengenai Piala Dunia Fifa 2022 COMPARISON OF NAIVE BAYES TAXONOMY AND SUPPORT VECTOR MACHINES FOR ANALYZING TWITTER USER SENTIMENT ON FIFA WORLD CUP 2022," *TEKNOMATIKA*, vol. 13, no. 01, 2023.

[9] R. I. Alhaqq and Y. Ruldeviyani, "Analisis Sentimen terhadap Penggunaan Aplikasi MySAPK BKN di Google Play Store," 2022. [Online]. Available: https://www.researchgate.net/publication/367216412

[10] S. Al-Saqqa and A. Awajan, "The Use of Word2vec Model in Sentiment Analysis: A Survey," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Dec. 2019, pp. 39–43. doi: 10.1145/3388218.3388229.

[11] I. E. dan M. M. S. Cindy Chairunnisa, "Klasifikasi Sentimen Ulasan Pengguna Aplikasi PeduliLindungi di Google Play Menggunakan Algoritma Support Vector Machine dengan Seleksi Fitur Chi-Square," 2022.

[12] S. Rabbani, D. Safitri, N. Rahmadhani, A. A. F. Sani, and M. K. Anam, "Perbandingan Evaluasi Kernel SVM untuk Klasifikasi Sentimen dalam Analisis Kenaikan Harga BBM," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 3, no. 2, pp. 153–160, Oct. 2023, doi: 10.57152/malcom.v3i2.897.

[13] E. Suryati, A. Ari Aldino, N. Penulis Korespondensi, and E. Suryati Submitted, "Analisis Sentimen Transportasi Online Menggunakan Ekstraksi Fitur Model Word2vec Text Embedding Dan Algoritma Support Vector Machine (SVM)," vol. 4, no. 1, pp. 96–106, 2023, doi: 10.33365/jtsi.v4i1.2445.

[14] F. Wahyu Kurniawan and W. Maharani, "Analisis Sentimen Twitter Bahasa Indonesia dengan Word2Vec," 2020. [Online]. Available: https://code.google.com

[15] A. Ilham, N. Azmi Verdikha, and A. J. Latipah, "Jurnal Explore IT|64 Klasifikasi Ujaran Kebencian di Twitter Menggunakan Fitur Ekstraksi Glove dengan Support Vector Machine (SVM)," 2023, doi: 10.35891/explorit.

[16] D. Leni, F. Earnestly, R. Sumiati, A. Adriansyah, and Y. P. Kusuma, "Evaluasi sifat mekanik baja paduan rendah bedasarkan komposisi kimia dan suhu perlakuan panas menggunakan teknik exploratory data analysis (EDA)," *Dinamika Teknik Mesin*, vol. 13, no. 1, p. 74, Apr. 2023, doi: 10.29303/dtm.v13i1.624.

[17] Fendy yulianto, *Improved Eliminate Particle Swarm Optimization on Support Vector Machine for Freshwater Fish Classification*. IEEE, 2019.

[18] T. A. A. H. Kusuma, K. Usman, and S. Saidah, "PEOPLE COUNTING FOR PUBLIC TRANSPORTATIONS USING YOU ONLY LOOK ONCE METHOD," *Jurnal Teknik Informatika (Jutif)*, vol. 2, no. 1, pp. 57–66, Feb. 2021, doi: 10.20884/1.jutif.2021.2.2.77.

[19] Arif Widiasan Subagio, Anggraini Puspita Sari, and Andreas Nugroho Sihananto, "Klasifikasi Lexicon-Based Sentiment Analysis Tragedi Kanjuruhan pada Twitter Menggunakan Algoritma Convolutional Neural Network," *Jurnal ilmiah Sistem Informasi dan Ilmu Komputer*, vol. 4, no. 1, pp. 166–177, Jan. 2024, doi: 10.55606/juisik.v4i1.759.