JURNAL SISTEM INFORMASI BISNIS (JUNSIBI)
OPEN ACCES
p-ISSN: 2774-3446
e-ISSN: 2774-3454
diterbitkan oleh:
Program Studi Sistem Informasi
Institut Bisnis dan Informatika (IBI) Kosgoro 1957
Vol. 4, No. 2, Oktober 2023, pp. 52-58
DOI: https://doi.org/10.55122/junsibi.v4i2.736

KLASIFIKASI KEBUTUHAN PERANGKAT LUNAK BERDASARKAN KATEGORI ISO/IEC 25000 MENGGUNAKAN TEXTRANK DAN SVM

PENULIS

[1)]Mutia Rahmi Dewi, [2)]Fatimatus Zulfa, [3)]Dady Khairul Imam, [4)]Daniel Oranova Siahaan

ABSTRAK

Perancangan kebutuhan perangkat lunak merupakan langkah pertama yang harus dilakukan dalam perencanaan membangun perangkat lunak. Untuk mempermudah melakukan analisis, kebutuhan perangkat lunak dapat dikategorikan ke dalam standar kualitas. Salah satu standar kualitas dapat menggunakan ISO/IEC 25000. Seri ISO/IEC 25000 terdiri dari 8 karakteristik yaitu *Functional Suitability*, *Performance Efficiency*, *Compatibility*, *Usability*, *Reliability*, *Security*, *Maintainability*, dan *Portability*. Untuk mempermudah dan mempercepat analisis diperlukan klasifikasi secara otomatis. Berbagai metode klasifikasi terhadap standar kualitas kebutuhan perangkat lunak telah diusulkan. Pada penelitian ini, melakukan ekstraksi kata kunci dari kebutuhan perangkat lunak menggunakan metode TextRank untuk melakukan klasifikasi terhadap standar kualitas ISO/IEC 25000. Kata kunci yang telah diekstraksi mewakili istilah pada kebutuhan perangkat lunak. Sebanyak 154 kebutuhan dari 5 perangkat lunak diekstraksi menjadi 66 kata kunci yang akan digunakan untuk melakukan klasifikasi terhadap standar kualitas ISO/IEC 25000. Pada penelitian ini, didapatkan hasil presisi sebesar 83% dan *recall* sebesar 80.3% dengan menggunakan klasifikasi *Support Vector Machine* (SVM).

Kata Kunci: ISO/IEC 25000, Kebutuhan Perangkat Lunak, TextRank, SVM

AFILIASI

Program Studi: [1)]Jurusan Teknologi Informasi, Program Studi Teknologi Rekayasa Perangkat Lunak
[2,3,4)]Teknik Informatika, Fakultas Teknik Elektro dan Informatika Cerdas

Nama Institusi: [1)]Politeknik Negeri Padang
[2,3,4)]Institut Teknologi Sepuluh Nopember

Alamat Institusi: [1)]Jl. Kampus, Limau Manis, Kec. Pauh, Kota Padang, Sumatera Barat – 25164
[2,3,4)]Jl. Teknik Kimia, Keputih, Kec. Sukolilo, Surabaya, Jawa Timur – 60111

KORESPONDENSI

Penulis: Mutia Rahmi Dewi
Email: mutiarahmi@pnp.ac.id

I. PENDAHULUAN

Perancangan kebutuhan perangkat lunak merupakan langkah pertama yang harus dilakukan dalam perencanaan membangun perangkat lunak. Kebutuhan perangkat lunak biasanya ditulis secara alami yang biasanya memiliki berbagai istilah. Untuk memudahkan melakukan analisis, kebutuhan perangkat lunak dapat dikategorikan ke dalam standar kualitas (*software quality*). Standard kualitas dapat menggunakan ISO/IEC 25000. Standard kualitas ISO/IEC 25000 terdiri dari 8[1] karakteristik sebagai berikut:

1) *Functional Portability*
   Kemampuan perangkat lunak untuk menyediakan fungsionalitas yang sesuai untuk tugas-tugas tertentu dan tujuan pengguna.
2) *Performance Effiency*
   Kinerja perangkat lunak dan jumlah sumber daya yang digunakan.
3) *Compability*
   Perangkat lunak ini mampu berjalan di berbagai perangkat, sistem operasi, aplikasi, atau lingkungan jaringan yang berbeda.
4) *Usability*
   Kemudahan perangkat lunak untuk digunakan dan dipelajari.
5) *Reliability*
   Kelayakan perangkat lunak, meliputi akurasi *fault tolerance*, konsistensi, dan kesederhanaan.
6) *Security*
   Keamanan perangkat lunak.
7) *Maintainability*
   Kemudahan perangkat lunak untuk dipelihara dan dikembangkan.
8) *Portability*
   Kemampuan perangkat lunak untuk beradaptasi dengan lingkungan baru.

Untuk mempermudah dan mempercepat analisis, diperlukan klasifikasi terhadap standar kualitas secara otomatis.

Penelitian ini mengacu pada penelitian[2] yang melakukan klasifikasi kebutuhan perangkat lunak berdasarkan standar kualitas ISO/IEC 25000. Pada penelitian tersebut dilakukan ekstraksi kata kunci dengan menggunakan TextRank untuk melakukan klasifikasi kebutuhan non fungsional pada perangkat lunak.

Penelitian lain yang dilakukan yaitu melakukan klasifikasi kualitas pada *website e-commerce* menggunakan standar kualitas ISO/IEC 25010[3]. Penelitian ini mengusulkan metode Fuzzy Mamdani untuk mengevaluasi kualitas *website e-commerce* berdasarkan bobot atribut dan kepentingan karakteristik dengan menggunakan *Analytical Hierarchy Process* (AHP).

Penelitian lain mengusulkan model situs *web-commerce* menggunakan Naive Bayes[4]. Metode tersebut digunakan untuk melakukan klasifikasi kualitas web *e-commerce* terhadap standar kualitas ISO 9126.

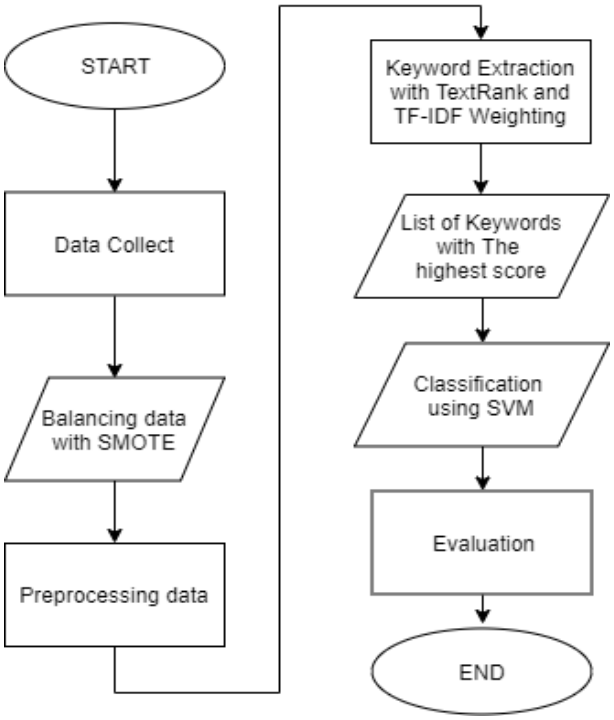
Pada penelitian ini, melakukan ekstraksi kata kunci dari kebutuhan perangkat lunak menggunakan metode TextRank untuk melakukan klasifikasi terhadap standar kualitas ISO/IEC 25000. Dengan menggunakan TextRank dalam melakukan ekstraksi kata kunci, mendapatkan hasil yang banyak relevan dengan kategori ISO/IEC 25000[2].

Kata kunci yang telah diekstraksi mewakili istilah pada kebutuhan perangkat lunak. Sebanyak 154 kebutuhan dari 5 perangkat lunak, diekstraksi menjadi 66 kata kunci yang akan digunakan untuk melakukan klasifikasi terhadap standar kualitas ISO/IEC 250000. *Keyword* hasil ekstraksi kebutuhan perangkat lunak digunakan untuk klasifikasi metode *Support Vector Machine*.

II. METODE PENELITIAN

Metode klasifikasi yang digunakan pada penelitian ini yaitu berdasarkan ekstraksi kata kunci perangkat lunak. Ekstraksi kata kunci perangkat lunak dilakukan dengan teknik TextRank. Untuk langkah-langkah klasifikasi dapat dilihat pada Gambar 1.

Sebelum dilakukan klasifikasi, dilakukan pengumpulan *dataset* yaitu daftar kebutuhan perangkat lunak. Daftar kebutuhan perangkat lunak diperoleh dari 5 dokumen SKPL di Departemen Teknik Informatika ITS. Kemudian dilakukan *preprocessing* untuk menghapus data yang tidak diperlukan dan tidak memiliki nilai. Langkah selanjutnya dilakukan ekstraksi kata kunci dengan teknik TextRank. Kemudian dilakukan penyetaraan bobot dengan cara mengalikan hasil dari ekstraksi dengan TF-IDF. Kemudian dilakukan klasifikasi sesuai dengan kata kunci yang telah diekstraksi. Dan yang terakhir dilakukan evaluasi metode. Untuk penjelasan langkah secara detail sebagai berikut.



Gambar 1. Metode Usulan

2.1 *Review Dataset*

Dataset yang digunakan pada penelitian ini yaitu kebutuhan perangkat lunak. Kebutuhan perangkat lunak yang digunakan berjumlah 154 kebutuhan dari 5 perangkat lunak. Kebutuhan perangkat lunak dibedakan menjadi kebutuhan fungsional yaitu yang menentukan sistem apa yang harus dilakukan dan kebutuhan non fungsional yaitu yang menentukan sistem yang seharusnya [5], [6].

Pada penelitian ini menggunakan kebutuhan fungsional maupun kebutuhan non fungsional dari perangkat lunak.

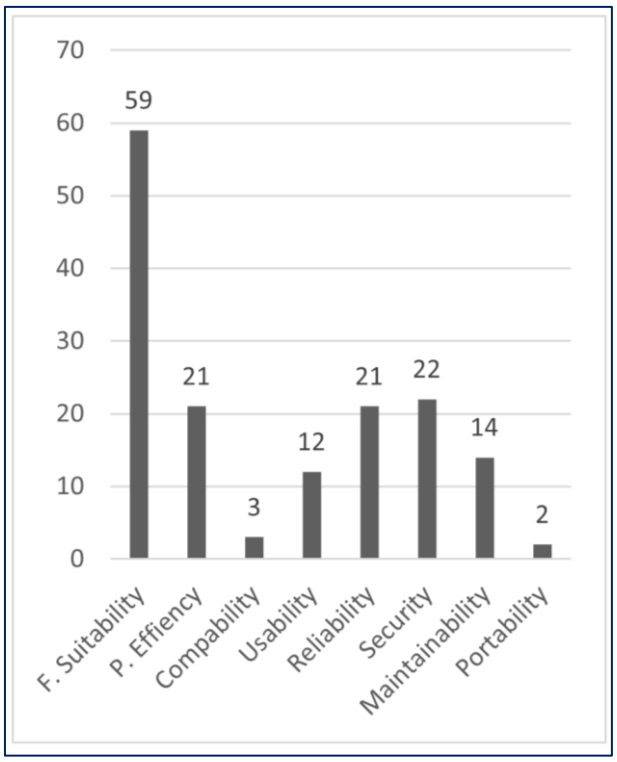
2.2 SMOTE

SMOTE (*Minority Synthetic Over-Sampling Technique*) adalah teknik *oversampling* yang baik dan efektif untuk mengatasi *overfitting* pada proses *oversampling* dengan menangani ketidakseimbangan di kelas modul yang memiliki kecacatan di kelas minoritas[7]. Pada penelitian ini, menggunakan *tools* WEKA untuk melakukan *balancing data*.

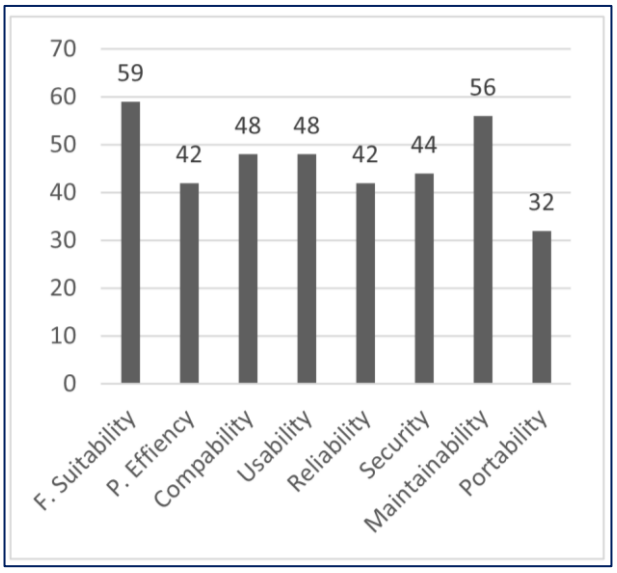
Sebelum dilakukan *balancing* data menggunakan SMOTE, jumlah data kebutuhan perangkat lunak setiap kelasnya seperti pada Gambar 2. Terdapat 8 kelas dengan jumlah data pada masing-masing yaitu *Functional Suitability* sebanyak 59 data, *Performance Efficiency* sebanyak 21 data, *Compatibility* sebanyak 3

data, *Usability* sebanyak 12 data, *Reliability* sebanyak 21 data, *Security* sebanyak 22 data, *Maintainability* sebanyak 14 data, dan *Portability* sebanyak 2 data.

Setelah dilakukan *balancing* data dengan SMOTE, jumlah data kebutuhan perangkat lunak setiap kelasnya seperti pada Gambar 3. Jumlah masing-masing kelasnya menjadi *Functional Suitability* sebanyak 59 data, *Performance Efficiency* sebanyak 42 data, *Compatibility* sebanyak 48 data, *Usability* sebanyak 48 data, *Reliability* sebanyak 42 data, *Security* sebanyak 44 data, *Maintainability* sebanyak 56 data, dan *Portability* sebanyak 32 data.



Gambar 2. Data Kebutuhan Perangkat Lunak Sebelum *Balancing* Data dengan SMOTE



Gambar 3. Data Kebutuhan Perangkat Lunak Setelah *Balancing* Data dengan SMOTE

2.3 *Preprocessing*

Sebelum dilakukan ekstraksi kata kunci terhadap kebutuhan perangkat lunak, dilakukan *preprocessing* terhadap dataset yang digunakan. *Preprocessing* yang dilakukan adalah *casefolding*, *stopword removal*, *lemmatization*, dan *tokenization*. *Casefolding* pada penelitian ini dilakukan bertujuan untuk menyamakan huruf pada data kebutuhan perangkat lunak dengan menjadikan penulisan huruf menjadi kecil semua. Selanjutnya dilakukan *stopward removal* dengan menghapus kata penghubung pada kalimat, seperti kata "itu", "atau" dan lain sebagainya. Kemudian dilakukan *lemmatization* dengan tujuan mencari kata dasar dari *dataset*. *Preprocessing* yang terakhir yaitu *tokenization*. *Tokenization* dilakukan dengan tujuan mendapatkan *term* dari kalimat kebutuhan perangkat lunak.

2.4 Ekstraksi Kata Kunci dengan TextRank

Metodologi yang digunakan dalam melakukan ekstraksi kebutuhan perangkat lunak yaitu Teknik TextRank. Teknik ini dilakukan dengan cara melakukan perangkingan terhadap kata kunci yang telah diekstraksi. Untuk langkah-langkah perangkingan dilakukan sebagai berikut[8]:.

1. Melakukan identifikasi *text* menjadi *vertex* pada *graph*.
2. Melakukan identifikasi hubungan antar *vertex*. Pada penelitian ini menerapkan *edge* yang tidak berarah.
3. Dilakukan iterasi algoritma TextRank.
4. Mengurutkan *vertex* sesuai dengan perangkingan terakhir.

Pada penelitian ini, masing-masing *term* dalam bentuk token yang diperoleh dari hasil praproses dihitung nilainya dengan menggunakan metode TextRank. Hasil perhitungan metode TextRank diperoleh dengan persamaan 1:

$$\boldsymbol{S(V_e) = (1-d) + d * (S(V_a) + \frac{1}{2} S(V_b))} \quad \text{(1)}$$

dimana $S(V_a)$ dan V_b masing-masing merupakan bobot dari $term_a$ dan $term_b$, dan d merupakan koeffisien *damping factor* (jika tidak ada link keluar). Pada metode ini, koefisien *damping factor* diinisialisasi senilai 0.85.

2.5 TF-IDF

Metode TF-IDF diterapkan setelah dilakukan teknik perangkingan dengan TextRank. Hasil dari kata kunci yang telah diekstraksi dikalikan dengan hasil TF-IDF kebutuhan perangkat lunak untuk memberikan bobot setiap kata kunci yang telah diekstraksi[9]. TF-IDF merupakan sebuah evaluasi seberapa penting suatu kata tersebut dalam sebuah kalimat atau dokumen. Rumus yang digunakan dalam perhitungan TF-IDF dapat dilihat pada persamaan 2.

$$\boldsymbol{Wi,j = tfi,j \; x \; log\left(\frac{N}{dfi}\right)} \quad \text{(2)}$$

dimana Wi,j adalah bobot untuk istilah I dalam dokumen j, N adalah jumlah dokumen dalam koleksi, tfi,j adalah frekuensi istilah dari istilah i dalam dokumen j dan dfi adalah frekuensi dokumen istilah i dalam koleksi.

2.6 Metode Evaluasi Klasifikasi

Setelah melakukan ekstraksi kata kunci dengan menggunakan metode TextRank serta perkalian dengan bobot untuk masing-masing *term*nya, maka pada tahap ini dilakukan metode klasifikasi dengan menggunakan klasifier *Support Vector Machines* (SVM).

Konsep SVM dapat dijelaskan secara sederhana dengan mencoba menemukan *hyperlane* terbaik yang bertindak sebagai pemisah antara dua *class* di ruang *input*. Secara konseptual, SVM adalah pengklasifikasi linier, tetapi SVM dapat dimodifikasi dengan trik kernel sehingga dapat digunakan untuk menyelesaikan permasalahan non-linier. Walaupun masih tergolong metode baru, SVM menawarkan kinerja yang lebih baik dibandingkan metode *machine learning* lainnya seperti *Naive Bayesian*, *Decision Tree* atau *Artificial Neural Network* (ANN)[10].

Kernel yang digunakan pada penelitian ini adalah kernel *polynomial*. Kernel *polynomial* menggunakan transformasi polinomial untuk memetakan data ke ruang fitur yang lebih tinggi. *Degree* dan *coefficient* mengatur kompleksitas transformasi, sehingga SVM dapat memisahkan kelas yang tidak linier terpisah. Dengan demikian, kernel ini mampu menangani masalah klasifikasi kompleks yang tidak dapat dipisahkan secara linear[11].

III. HASIL DAN PEMBAHASAN

Pada penelitian ini, dilakukan klasifikasi kebutuhan fungsional dan non fungsional berdasarkan ISO/IEC 25000. Data kebutuhan yang telah dilabeli secara manual kemudian dilakukan praproses teks. Setelah itu dilakukan ekstraksi kata kunci dengan menggunakan metode TextRank untuk mencari kata kunci yang mewakili suatu kebutuhan berdasarkan kelasnya, dalam hal ini yaitu ISO/IEC 25000. Setelah itu, hasil ekstraksi kata kunci disempurnakan dengan melakukan perkalian dengan bobot dari masing-masing kata kunci. Hal ini dikarenakan terdapat beberapa *term* dari hasil ekstraksi kata kunci dengan TextRank tidak dianggap sebagai kata kunci. Namun, *term* tersebut memperoleh nilai tertinggi dengan metode TextRank. Maka dari itu, nilai masing-masing *term* yang diperoleh dari metode TextRank dikalikan dengan bobot TF-IDF dari masing-masing kata kunci. Dengan demikian nilai *term* yang dikalikan dengan bobot TF-IDF, lebih merepresentasikan sebagai sebuah kata kunci dari kebutuhan terkait. Model dari ekstraksi kata kunci kemudian digunakan untuk melakukan klasifikasi kebutuhan berdasarkan ISO/IEC 25000.

Sebelum dilakukannya klasifikasi, dilakukan *balancing* data pada data model. Hal ini dikarenakan jumlah data pada model tersebut sedikit, sehingga model yang terbentuk masih kurang merepresentasikan kelas dari model data. Hal ini dapat diatasi dengan menggunakan SMOTE, sehingga data dari masing-masing kelas memiliki jumlah yang seimbang.

Pada penelitian ini, diperoleh hasil klasifikasi dengan presisi sebesar 83% dan *recall* sebesar 80.3% dengan menggunakan klasifier SVM dan 10-*cross validation*. Detail hasil klasifikasi kebutuhan ditunjukkan pada Tabel 1. Pada kelas *Performance Efficiency* mendapatkan hasil presisi sebesar 66.7% dan *recall* sebesar 61.9%. Pada kelas *Functional Suitability* mendapatkan hasil presisi sebesar 51.8% dan *recall* sebesar 74.6%. Pada kelas *Reliability* mendapatkan hasil presisi sebesar 83.8% dan *recall* sebesar 73.8%. Pada kelas *Compatibility* mendapatkan hasil presisi sebesar 100% dan *recall* sebesar 95.8%. Pada kelas *Portability* mendapatkan hasil presisi sebesar 96.9% dan *recall* sebesar 96.9%. Pada kelas *Security* mendapatkan hasil presisi sebesar 89.7% dan *recall* sebesar 79.5%. Pada kelas *Maintainability* mendapatkan hasil presisi sebesar 97.8% dan *recall* sebesar 80.4%.. Serta pada kelas *Usability* mendapatkan hasil presisi sebesar 85.1% dan *recall* sebesar 83.3%.

Tabel 1. Hasil Klasifikasi *Support Vector Machine* (SVM)

Kelas	Presisi	*Recall*	F-*Measure*
Performance Efficiency	66.7%	61.9%	64.2%
Functional Suitability	51.8%	74.6%	61.1%
Reliability	83.8%	73.8%	78.5%
Compatibility	100%	95.8%	97.6%
Portability	96.9%	96.9%	96.9%
Security	89.7%	79.5%	84.3%
Maintainability	97.8%	80.4%	88.2%
Usability	85.1%	83.3%	84.2%
Weighted Average	83%	80.3%	81.1%

Nilai presisi terbaik diperoleh pada kelas *Compability*, nilai *recall* terbaik diperoleh pada kelas *Portability*, dan nilai F-Measure diperoleh pada kelas *Compability*. Nilai presisi terbaik diperoleh pada kelas *Compability*, dikarenakan pada saat sebelum dilakukannya *balancing* data, data dari kelas *Compability* memiliki data yang sedikit. Dengan adanya *balancing* data, data yang sedikit tersebut mengalami replikasi secara berulang sehingga saat diklasifikasi sangat besar peluang dari data tersebut muncul. Sehingga kelas *Compability* memiliki presisi yang paling tinggi. Untuk mengatasi hal tersebut, maka diperlukannya ekspansi jumlah data sehingga tidak terdapat kasus replikasi data jika dilakukannya *balancing* data pada kelas yang memiliki jumlah data yang sedikit.

Dari hasil presisi dan *recall* yang di dapat dari masing-masing kelas, rata-rata presisi yang di dapatkan pada klasifikasi ini yaitu sebesar 83% dan *recall* sebesar 80.3%. Hasil presisi dan *recall* yang didapatkan dari klasifikasi merupakan hasil yang baik. Hal tersebut terjadi karena pada penelitian ini menambahkan model ekstraksi kata kunci dari perangkat lunak dengan metode TextRank dan menggunakan algoritma SVM untuk melakukan klasifikasi. Penambahan model kata kunci pada klasifikasi dapat membantu memperjelas klasifikasi, karena model kata kunci yang diberikan memiliki arti yang sama dengan data kebutuhan perangkat lunak. Selain itu, pada klasifikasi menggunakan algoritma SVM yang terbukti handal dalam melakukan klasifikasi teks. Dengan menggunakan algoritma SVM, klasifikasi dilakukan dengan menggunakan kernel yang efektif dalam pengklasifikasian.

VI. KESIMPULAN

Klasifikasi yang diusulkan pada penelitian ini yaitu klasifikasi terhadap standar kualitas ISO/IEC 25000 menggunakan kata kunci. Kata kunci di dapatkan dari ekstraksi kebutuhan perangkat lunak dengan menggunakan metode TextRank dan dilakukan pembobotan terhadap kata kunci menggunakan metode TF-

IDF. Setelah kata kunci didapatkan, kata kunci akan ditambahkan pada *dataset* untuk digunakan sebagai model. Kemudian dilakukan klasifikasi dengan menggunakan algoritma *Support Vector Machine* atau disebut SVM.

Pada penelitian ini yaitu klasifikasi kebutuhan perangkat lunak terhadap ISO/IEC 25000 dengan menggunakan metode SVM dan Textrank sebagai metode untuk mendapatkan model kebutuhan perangkat lunak didapatkan presisi sebesar 83% dan *recall* sebesar 80.3%.

REFERENSI

[1] T. Kh and I. Hamarash, "Model-based quality assessment of internet of things software applications: A systematic mapping study," 2020.

[2] I. P. Delgado-Solano, A. S. Núñez-Varela, and G. H. Pérez-González, "Keyword Extraction From Users' Requirements Using TextRank and Frequency Analysis, and Their Classification into ISO/IEC 25000 Quality Categories," in *2018 6th International Conference in Software Engineering Research and Innovation (CONISOFT)*, 2018, pp. 88–92.

[3] F. H. Wattiheluw, S. Rochimah, and C. Fatichah, "Klasifikasi Kualitas Perangkat Lunak Berdasarkan Iso/Iec 25010 Menggunakan Ahp Dan Fuzzy Mamdani Untuk Situs Web E-Commerce," *JUTI J. Ilm. Teknol. Inf.*, vol. 17, no. 1, p. 73, 2019.

[4] A. Stefani and M. Xenos, "E-commerce system quality assessment using a model based on ISO 9126 and Belief Networks," *Softw. Qual. J.*, vol. 16, no. 1, pp. 107–129, 2008.

[5] M. Lu and P. Liang, "Automatic classification of non-functional requirements from augmented app user reviews," in *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, 2017, pp. 344–353.

[6] N. Jha and A. Mahmoud, "Mining non-functional requirements from app store reviews," *Empir. Softw. Eng.*, vol. 24, pp. 3659–3695, 2019.

[7] A. Fernández, S. Garcia, F. Herrera, and N. V Chawla, "SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *J. Artif. Intell. Res.*, vol. 61, pp. 863–905, 2018.

[8] S. Rajput, A. Gahoi, M. Reddy, and D. M. Sharma, "N-Grams TextRank A Novel Domain Keyword Extraction Technique," in *Proceedings of the 17th International Conference on Natural Language Processing (ICON): TermTraction 2020 Shared Task*, 2020, pp. 9–12.

[9] S. Qaiser and R. Ali, "Text mining: use of TF-IDF to examine the relevance of words to documents," *Int. J. Comput. Appl.*, vol. 181, no. 1, pp. 25–29, 2018.

[10] S. Ghosh, A. Dasgupta, and A. Swetapadma, "A study on support vector machine based linear and non-linear pattern classification," in *2019 International Conference on Intelligent Sustainable Systems (ICISS)*, 2019, pp. 24–28.

[11] X. He, Z. Wang, C. Jin, Y. Zheng, and X. Xue, "A simplified multi-class support vector machine with reduced dual optimization," *Pattern Recognit. Lett.*, vol. 33, no. 1, pp. 71–82, 2012.