https://doi.org/10.31449/inf.v49i19.9271

Text Segmentation in Online Health Consultation Using Multi Layer Perceptron with Sentence Embedding and Sentence Features

Yunianita Rahmawati, Daniel Siahaan*, Diana Purwitasari
Department of Informatics, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember (ITS), Surabaya 60111, Indonesia
E-mail: 7025211024@student.its.ac.id, daniel@if.its.ac.id, diana@its.ac.id
*Corresponding author

Keywords: aspects of doctor-patient communication, OHC, MLPSentFeat, sentence feature, text segmentation

Received: September 3, 2025

The six aspects of doctor-patient communication are universal and relevant to various medical contexts and are the basis for structuring and analyzing doctors' answer texts in online health care. Identifying each aspect of communication in the doctor's answers is important to ensure effective communication. There are three critical challenges in segmenting the doctor's answers, i.e. limited public dataset, extreme imbalance data, and implicit semantic variations between aspects. This study proposed a novel approach to text segmentation, focusing on communication aspects, particularly doctor-patient interactions. Unlike classical segmentation methods, which relied on topic similarity, the proposed method segmented text based on the communicative function of each sentence, offering a more accurate reflection of the interaction structure in Online Health Consultation (OHC). The model developed, MLPSentFeat, integrated sentence features directly into its architecture and demonstrated effectiveness in handling data with highly imbalanced label distributions without the need for synthetic data or complex balancing techniques.

Sentence features were formed using a combined approach: the Likelihood Ratio (LR) algorithm filtered words relevant to a label, and Information Gain (InfoGain) selected the most informative features. Experimental results showed that integrating these sentence features significantly enhanced the model's sensitivity to variations in linguistic structure, especially in recognizing non-standard sentence types, such as questions, which were prevalent in the information-gathering aspect of medical communication.

The best model produced, MLPSentFeat+DS2+Features3, was optimized to achieve the lowest segment error percentage of 8.18%, despite a slight performance decline in some labels after optimization. The use of text normalization, along with appropriate data size, cleanliness, and alignment of sentence features with the sentences' semantic structure, proved crucial in preventing overfitting. The MLPSentFeat model was successfully applied across various domains, demonstrating cross-domain adaptability, including in doctor's answer in online medical consultation systems, with potential for further development to identify questions with diverse sentence structures.

Povzetek: Študija predstavlja nov pristop k segmentaciji besedil zdravnikovih odgovorov v spletnih zdravstvenih posvetih, ki temelji na komunikacijski funkciji stavkov in z modelom MLPSentFeat učinkovito obravnava neuravnotežene podatke ter izboljšuje prepoznavanje različnih vidikov komunikacije zdravnik–pacient.

1 Introduction

The public increasingly used online health consultation (OHC) services as a source of fast and practical health information [1][2]. The main attraction lay in the interaction between the doctor and the patient, primarily through the patient's questions and answers. The consultation text included the user's question, the doctor's answer, the consultation title, the doctor's name, and the time. The OHC service classified these consultation texts based on disease topics. Doctors' answers were prepared based on personal preferences, intuition, and the context of the questions received, and without standard structural guidance, giving rise to variations in content and appearance (layout). Doctors' answers were not structured in terms of text content; for example, they repeated discussions of a topic in different parts or conveyed topics that were not always relevant to the patient's questions and disease topics [3]. Sometimes, the medical communication delivered was also less effective [4]. Irregularities in layout also occurred, such as using paragraph breaks unrelated to a particular topic or combining several topics in one paragraph, which caused the main explanation to lack focus. OHC doctors' answers tended to be short, with an average of only 20 to 25 sentences [5], causing unclear topic boundaries, which indicated weaknesses in the text structure [6]. For communication to be more effective, the doctor's answers needed to be structured following aspects of doctor-patient communication that had been proven to support the therapeutic relationship and increase patient satisfaction and compliance [7][8]. Furthermore, to ensure that the doctor's responses were well-structured according to the existing layout, these responses should have been divided into smaller segments. This segmentation aimed to make the text more readable and comprehensible [6], as well as to enable readers to more easily locate relevant topics without having to read the entire text [9]. Segmentation of doctors' answers was carried out at the sentence level because the text length was relatively short, so that topic changes could occur at the sentence level. Acceptance of text-based communication differed from oral communication, so users might have experienced difficulties in interpreting written text that was not well structured [10].

This research used the six aspects of doctor-patient communication formulated by Ann King [11]: fostering the relationship, gathering information, providing information, making decisions, responding to emotions, and enabling disease- and treatment-related behavior. These six aspects were universal and relevant to various medical contexts, thus serving as the basis for structuring and analyzing doctors' answer texts in OHC. Identifying each aspect of communication in the doctor's answer was important to ensure effective communication, where patients felt respected and understood [12], and played an essential role in health services [13][14]. The information-gathering aspect helped patients reflect on their medical problems, even though doctors did not always receive an immediate response. One form of reflection was asking questions so the patient could understand their condition more deeply [15]. Providing information was related to patient satisfaction with communication, which increased when doctors were willing to discuss the patient's medical condition openly [16]. Responding to emotions allowed doctors to convey information with empathy and provide hope [16]. Fostering the relationship built trust between patients and doctors [16][17]. Decision-making enabled doctors to recommend medical treatment to patients, especially in cases requiring immediate care [18]. The enabling disease- and treatment-related behavior aspect provided practical understanding to patients regarding how to alleviate or cure the disease they were experiencing [19]. Even though not all aspects appeared equally in every doctor's answer, this research still identified all of these aspects to evaluate the quality of communication in OHC.

This research faced several challenges in segmenting doctors' answer texts. First, although the number of labeled sentences in the dataset was relatively large, the total amount of medical consultation data available in each category was relatively limited. This limitation in the amount of medical consultation data resulted in a restricted variation in the information that could be learned by the model. Doctors' answer responses tended to follow similar patterns, which restricted the diversity of information in the data, causing the model to focus on highly similar patterns within the response texts. Therefore, despite the large number of labeled sentences, the limited variation in the medical consultation data became the primary factor leading to overfitting in the classification model. Second, the distribution of doctors' answer sentences was highly unbalanced, where aspects such as gathering information and responding to emotion (especially expressions of sympathy and empathy) were minimal, while the providing information aspect dominated. The OHC service did not provide an interface for users to answer follow-up questions from doctors in consultation sessions, so doctors limited the collection of information from users. In addition, doctors rarely expressed sympathy and empathy due to their difficulty in dealing with uncertainty in understanding patients' emotions and expressing empathy [10]. Third, this research formulated text segmentation targeting six aspects, each of which reflected an aspect of doctor-patient communication. In addition, existing text segmentation methods generally did not provide semantic labels, so the coverage of communication aspects in each segment was unclear [3][20][21]. Fourth, the structure of neural network models, such as MLP, had several hidden layers and neurons that needed to be optimized using a metaheuristic optimization algorithm so that classification performance improved [22][23][24][25]. Variations in wording and sentence forms in doctors' answer texts, as well as similarities between classes, reduced the effectiveness of simple models. Therefore, proper parameter tuning became crucial to improve the accuracy, generalization, and efficiency of model training.

In classical segmentation, one segment contained sentences whose topics were consistent. However, in segmentation based on communication aspects, the similarity of segments was determined not by the topic, but by the communicative function of the sentence. Therefore, even though the topic changed (for example, from 'itch' to 'flu'), if both sentences functioned as aspects of providing information, they were still considered to be in one segment.

The models that addressed the text segmentation task consisted of unsupervised and supervised learning. Unsupervised models included clustering, similarity-based, and probabilistic generative models [26]. Text segmentation used probabilistic generative models such as topic modeling [20][27][28][29][30], which allowed the identification of "hot topics" in documents [27]. This study formed segments based on doctor-patient communication aspects, which were not discussed explicitly in the doctor's answers. Text segmentation could also be performed using clustering models, in which segments or segment topics were formed based on the words discussed in the text [21][31][32][3]. However, since the six aspects of doctor–patient communication were not explicitly addressed in the doctors’ responses, clustering models were less suitable for segmenting the text based on these aspects. Text segmentation using similarity-based methods such as TextTiling [33], C99 [34], GraphSeg [35][36][3], Content Vector Segmentation (CVS) [37][38][3], allowed the formation of segments by measuring the similarity between text units and was applied to long data [33]. The text content of a doctor's answer in a communication aspect could cover various topics, and differences could be observed between sentences. In addition, doctors' response texts tended to be shorter, consisting of only 20 to 25 sentences [5] or one or two paragraphs.

Text segmentation used supervised learning based on certain aspects, for example, the sentiment analysis aspect with SEGBOT [26] and the doctor-patient communication aspect using the Hierarchical Temporal Convolutional Network (HTCN) [39]. The SEGBOT model was not designed to segment based on specific aspects of communication, such as doctor-patient communication, but rather to detect natural boundaries in text discourse structures based on semantic transitions between sentences. The HTCN model allowed for the segmentation of doctors' answer topics based on communication aspects and was applied to perform sentence-level segmentation. However, each aspect required many sentences. In addition, Wiguna's research only discussed sentence classification based on communication aspects but did not regroup dialogue into complete segments based on communication flow. This research shared a similarity with Wiguna's research in that it segmented the text of doctors' answers based on communication aspects. At the same time, the difference lay in the imbalance of sentence distribution, especially in the aspects of gathering information and responding to emotions (specifically sympathy and empathy). This difference represented an analysis gap that could be exploited to improve segmentation performance in this research.

The solution proposed in this research was to develop a text segmentation model using MLPSentFeat. This research aimed to form small segments of doctors' answers in OHC services and provide segment information. This model was designed through four main stages. First, at the pseudo-labeling stage, the training data were expanded by applying pseudo-labeling techniques based on voting classifiers from various machine learning algorithms and using the k-fold cross-validation algorithm to divide the training and testing data. Second, in the classification stage, MLPSentFeat was applied to the doctor's answers at the sentence level. Third, in the text segmentation stage, segments of doctors' answers were formed based on the classification results, labels, and annotations. Fourth, in the MLP optimization stage, the best MLP configuration was obtained, including the optimal number of hidden layers and neurons. This process was conducted by comparing various metaheuristic optimization algorithms and adjusting the MLP parameters: hidden layers, neuron range, population size, and repetition size. The best optimization model was then applied to various datasets.

The novelty of this study encompassed several methodological and empirical contributions. First, this research introduced a text segmentation approach based on specific communicative aspects—particularly those found in doctor–patient interactions—where segment similarity was determined not by topical relevance, but by the communicative function of each sentence. This departed from traditional segmentation methods, which typically constructed segments based on thematic coherence. Second, the MLPSentFeat model demonstrated the capability to classify data with highly imbalanced label distributions, where certain labels contained only a few dozen samples while others contained thousands, without the need to augment the data with synthetic (dummy) samples.

Moreover, the MLPSentFeat model operated efficiently and did not require high computational resources. Third, the model offered a novel architectural combination by integrating an MLP with explicitly constructed sentence features, enabling it to capture sentence patterns from labels with limited training instances. Fourth, sentence feature construction was achieved through a hybrid algorithm that combined LR and InfoGain, where LR filtered label-specific relevant words and InfoGain selected the most informative features. Fifth, the explicit integration of sentence features into the MLPSentFeat architecture was shown to improve the model's sensitivity to linguistic structural variation, particularly in recognizing non-standard sentence forms such as interrogative expressions, which were dominant in the gathering information aspect of doctor–patient communication. Sixth, as a minor novelty, this study also identified sentence forms in doctors' answers to selectively exclude certain words from the stopword list, thereby treating them as important features during the preprocessing stage, particularly in the stopword removal process.

This research focused on segmenting doctors' answers in OHC services based on aspects of doctor-patient communication. This segmentation not only contributed to improving users' understanding and access to information but also could be used by NLP developers for chatbot development, by OHC services to standardize answers, and by doctors to improve the quality of interactions with patients. This research contributed as follows:

- This research expanded the training data with pseudo-labeling using a voting classifier from several machine learning methods.
- This research proposed a text segmentation model using an MLP-based sentence embedding and sentence features for segmenting doctors' answers and annotating segments based on aspects of doctor-patient communication.
- The authors optimized the model by tuning its parameters to determine the optimal number of hidden layers and neurons for the MLP method after conducting an experiment using the initial parameters.

The structure of the remaining sections was organized as follows: Section II described relevant scientific work. Section III outlined the methodology used. Section IV presented the results and discussion. Section V summarized the conclusions of the research.

2 Related work

The related work provided a brief review of the research that was used as a reference in this study. This short review was divided into several subchapters, including the application of sentence forms in doctors' answers, the identification of aspects of doctor-patient communication, text segmentation, pseudo-labeling methods, and optimization methods.

Table 1: The form of the doctor's answer sentence is based on the doctor-patient communication.

Communication Aspect	Sentence Form	Example of a doctor's answers sentences	Role, Responsibilities, and Abilities of Doctors
Fostering the relationship (F1)	Phatic sentence [119]	'*Halo* (Hello)'. '*Sekian informasi dari saya* (That was the information from me)'.	The doctor greeted the patient and closed the consultation.
		'*Mohon maaf atas keterlambatan respon kami* (We apologized for our late response)'	The doctor admitted the error and expressed regret.
		'*Pertama-tama, saya hendak mengonfirmasi terlebih dahulu* (First of all, I wanted to confirm first)'	Doctors developed open communication skills.
		'*Jika sulit menghubungi dokter, silakan gunakan layanan chat di Play Store* (If it was difficult to contact a doctor directly, use the chat service in the Play Store)'	Doctors were involved in building partnerships.
Gathering information (F2)	Ordinary interrogative sentences [120][121]	'***Apakah** anda tidak mengalami menstruasi selama 3 bulan **?*** (Had you not had your period for 3 months?)'	Doctors asked open-ended questions to patients.
Providing information (F3)	Declarative sentences [122]	'*Bila hasilnya positif, maka keterlambatan menstruasi anda disebabkan oleh kehamilan* (If the result was positive, then your delayed menstruation was caused by pregnancy)'	Doctors shared patient information.
	Rhetorical interrogative sentences [120]	'*Baca laman artikel ini mengenai **Kenapa Bisa Terjadi Sakit Perut Sebelah Kanan ?*** (Read this article about **Why Can Right Side Abdominal Pain Occur ?**)'	
	Incomplete sentence [123]	'*Infeksi saluran kemih* (Urinary tract infection)'	
Decision-making (F4)	Command sentence (imperative form) [121]	'*Anda harus melakukan pemeriksaan ke dokter kandungan* (You had to have a check-up with a gynecologist)'	Doctors created collaborative action plans.
	Negative imperative sentences [124]	'*Bila keluhan berlanjut **jangan** ragu memeriksakan diri ke dokter* (If the complaint persisted, **do not** hesitate to see a doctor))'	
	Rhetorical interrogative sentences [120]	'*Dokter akan mengetahui **apakah** anda menderita penyakit ini berdasarkan hasil pemeriksaan* (The doctor would know whether you had this disease based on examination results)'	
	Declarative sentences [122]	'*Anda dapat menggunakan asuransi umum sehingga meringankan biaya prosedur tersebut* (You could use general insurance that covered the cost of the procedure)'	Doctors agreed with patients.
Responding to emotions (F5)	Empathy-expression sentence [125]	'*Saya mengerti kekhawatiran yang Anda rasakan* (I understood the concern you had. You were not alone. We felt the same)'	Doctors expressed empathy toward patients.
	Sympathy-expression sentence [126]	'*Sebelumnya kami turut prihatin terhadap kondisi yang menimpa ibu Anda* (We expressed our condolences regarding the condition that befell your mother)'	Doctors expressed sympathy for patients.
	Wish-expression sentence [121]	'***Semoga** ibu Anda lekas sembuh* (I hoped you got better soon)'	Doctors provided emotional support.
Enabling disease- and treatment-related behavior (F6)	Command sentence [121]	'*Perbanyak minum air untuk mencegah dehidrasi* (Drink more water to have prevented dehydration)'	Doctors changed patients' health behaviors or provided treatment-related advice.
	Negative imperative sentences [121]	'***Jangan** berikan jus dan minuman bersoda* ((**Don't** give juice and fizzy drinks)'	
	Rhetorical interrogative sentences [120]	'*Amati **apakah** diare tetap berlangsung lebih dari 1 minggu* (Observe whether diarrhea persisted for more than 1 week)'	

Table 2: Review of the methodology for identifying aspects of doctor-patient communication regarding the doctor's answers.

Study	Model	Embedding	Dataset	Research Gap	Contribution
Rahmawati et al. [5]	Multiclass classification with one-hot encoding. Compares LR, MLP, GPT-2, GRUs, FNN, LSTM, RNN, CNN, and BiLSTM.	Sentence embedding (SBERT)	Doctor's answers from the Alodokter website were from 502 data points with 11223 sentences. It was in Indonesian.	Used a small amount of data. Sentence features were extracted by selecting words with high frequency in each class.	Proposed MLPSentFeat. Alodokter dataset (13799 data). Sentence features were extracted from the keywords in each communication aspects. Feature reductions.
Zhafirah et al. [36]	Semantic Relatedness Graph (SRG).	Word embedding (IndoBERT)	Doctor's answers from the Alodokter website was 15,000. It was in Indonesian.	Segmentation results were not as good.	Segmenting doctor's answers using classification techniques.
Wiguna et al. [39]	HTCN, BiLSTM, LSTM, and CNN.	Word embedding (Word2Vec, FastText, GloVe)	The number of doctor's answers from the SehatQ website (KlikDokter, Alodokter) was 15,000. It was in Indonesian.	It had not been tested using sentence embedding. The amount of labeled data is small.	Using Sbert. Added 2000 labeled data from 42,226 sentences.
Rahmawati et al. [3]	Modeling of LDA topics, clustering (kMeans), and text segmentation (Graphseg, CVS).	Word embedding (TF-IDF)	Doctor's answers from the Alodokter website includes 500 data points with 8576 sentences. It was in Indonesian.	Used a small amount of data and had poor segmentation.	Use classification techniques based on aspects of doctor–patient communication to form segment and annotations.
Juanita [87]	Multi-label Relevance and Label Powerset classification by RF, Adaboost, KNN, and MLP.	Word embedding (Word2Vec, TF-IDF)	Doctor's answers from the Alodokter website includes 500 data points, and Steadyhealth includes 500 data points. They were in Indonesian and English.	Used a small amount of data. It had not been tested using sentence embedding.	Added labeled data by 2000 with 42,226 sentences. Using SBERT.

2.1. Use of sentence forms in doctors' answers

This section described the sentence forms in the doctor's answer for each aspect of doctor-patient communication, which were used as the basis in designing the model, such as preprocessing settings and sentence features. Identifying the doctor's answer sentence for the aspect of doctor-patient communication referred to the doctor's role, responsibility, and ability in each communication aspect [11]. The sentence form referred to the types of sentences used in each aspect of communication. The results of identifying the form of the

Placed the question word in the middle of the sentence and could end with or without a question mark. Declarative sentences conveyed statements from the doctor, while incomplete sentences represented structurally incomplete sentence forms. Command sentences were used to direct patients to perform certain actions, whereas negative imperative sentences were intended to prevent specific actions, typically indicated by the use of the word '*jangan* (do not)'. In addition, doctors

Doctor's answer sentence based on the elements of doctor-patient communication were shown in Table 1. Doctor-patient communication used different sentence forms, adjusted to the doctor's roles, responsibilities, and abilities. In addition, the writing style of doctors in the OHC service tended to use the same sentence form when explaining information, namely declarative sentences and rhetorical interrogatives.

Phatic sentences were used to maintain social relations between doctors and patients, as illustrated in Table 1. Ordinary interrogative sentences were written with a question word at the beginning and ended with a question mark—for example, starting with '*apakah* (had)' and ending with '?'. Rhetorical interrogative sentences also expressed sympathy, empathy, and prayers for the patient.

Some sentence forms were only used for one aspect of communication. Here are some examples. Phatic sentence forms (sentences used to build communication) were only used in the "fostering the relationship" aspect because doctors needed to develop relationships with patients. Sympathy, empathy, and wish sentence forms (sentences expressing hope or prayer) were only used in the "responding to emotion" aspect because doctors needed to acknowledge the patient's emotions (sympathy), understand the patient's feelings (empathy), and give hope for healing or pray for the patient (wish). Incomplete sentences were only used in the "providing information" aspect, which meant that only this aspect contained such sentence forms.

Other sentence forms were used in several aspects of communication. 1) Various forms of interrogative sentences with various uses in the aspects of gathering information, providing information, decision-making, and enabling disease- and treatment-related behavior. This sentence structure was frequently used in various aspects of communication, indicating that physicians often used interrogative sentences. Question sentences were used to obtain answers or confirm previously known information. Interrogative sentences generally required a question word or a question mark (?), although the mark might not have been used in some types of interrogative sentences. Interrogative sentences in the aspect of gathering information were used to obtain information from patients, while in other aspects were used to confirm information. 2) Various forms of command sentences were used in the aspects of decision-making and enabling disease- and treatment-related behavior. Doctors instructed patients to take action related to treatment in both aspects. 3) News sentences were used to provide information and make decisions. Doctors provided health information in the form of news sentences in both aspects.

Doctors provided general information in the aspect of providing information, while doctors conveyed information about medical procedures in the decision-making aspect.

2.2. Identify aspects of doctor-patient communication

Research on identifying aspects of communication between doctors and patients from the doctor's answers (Table 2) had evolved through various approaches, including text classification and segmentation methods. Multiple models were explored, ranging from multilabel and multiclass classification to machine learning and deep learning techniques. These methods had been extensively applied to enhance text analysis in health communication. Additionally, while most previous studies employed word embedding, a few utilized sentences embedding. The datasets used in several prior studies originated from the Alodokter platform, which provided a relatively small volume of data.

2.3. Text Segmentation method

Text segmentation referred to the process of dividing documents into smaller units, known as segments. These segments could consist of topics, words, sentences, lines, passages, or characters. The type of segment was typically determined by the analysis goal and target specification. Document segmentation supported text analysis by offering: 1) reduced and more coherent data chunks than complete documents, and 2) well-defined units for analysis and data access [40]. Text segmentation was commonly applied for purposes such as: 1) topic segmentation, which extracted topically coherent segments, and 2) discourse segmentation, which extracted Elementary Discourse Units (EDUs) and other discourse structures—covering both topical and linear segmentation [41]. Segment boundaries were typically marked by paragraph breaks [33].

Previous studies performing aspect-based segmentation included Wiguna, who segmented doctor's answers based on aspects of doctor–patient communication using the HTCN method. The HTCN method segmented texts by leveraging dilated causal convolution, allowing the model to capture temporal sentence relationships accurately, maintain sequential dependencies, and manage long-range dependencies without gradient vanishing issues. Li addressed segmentation of topics and discourse in sentiment analysis of film reviews using the SEGBOT method. SEGBOT used BiRNN for contextual understanding, RNN for decoding, and a pointer network to determine segment boundaries based on end-position probabilities. Zhu [42] applied aspect-based segmentation to Chinese restaurant reviews using the PRanking method, where segmentation occurred by scoring texts according to aspects and separating segments at score transitions. Hananto [20] conducted segmentation of online customer reviews using TopicDiff-LDA for automatic annotation based on tourism aspects. This method detected topic distribution changes using LDA to determine segment boundaries. These prior studies served as references for aspect-based segmentation research.

Research on segmentation in the health domain included Chowdhury [43], who developed the MedTextSeg algorithm for segmenting sections in medical reports. Eisenstein [44] proposed a Bayesian algorithm based on lexical cohesion, which did not require labeled data, and applied it to medical textbooks. Ganesan [45] segmented clinical texts by identifying headers, footers, and key sections such as allergies or chief complaints using Logistic Regression (LR). Apostolova [46] employed SVM to automatically classify report content into semantic sections. Edinger [47] implemented rule-based approaches to search specific segments in clinical texts, with compatibility for engines like Lucene and Essie. Sadoughi [48] utilized LSTM for detecting section boundaries in clinical dictations. Ginter [49] combined HMM and LSA to segment and annotate electronic patient records. These efforts provided a foundation for health-related text segmentation studies.

Applications of text segmentation spanned several domains, such as chatbot Q&A in manufacturing consulting services [50], prediction of workplace accident severity [51], and automated assessment in Chinese medical education [52]. Other applications included analysis of "infarction" in Japanese medical literature [53], construction of Chinese EMR dictionaries [54], model extraction from knowledge graphs for manufacturing planning [55], self-learning evaluation in science education [56], structured extraction of medical documentation [57], negation detection in French clinical records [58], comparison of medical corpora [59], and scene segmentation in narrative fiction [60].

2.4. Pseudo-labeling method

This study employed a classification model to automatically label unlabeled data. Pseudo-labeling used a voting classifier algorithm derived from multiple classification models, namely, LR, SVM, Decision Tree (DT), K-Nearest Neighbor (KNN), Random Forest (RF), Naïve Bayes (NB), Gradient Boosting Machine (GBM), Extreme Gradient Boosting (XGBoost), Light GBM (LGBM), and Categorical Gradient Boosting (CatBoost).

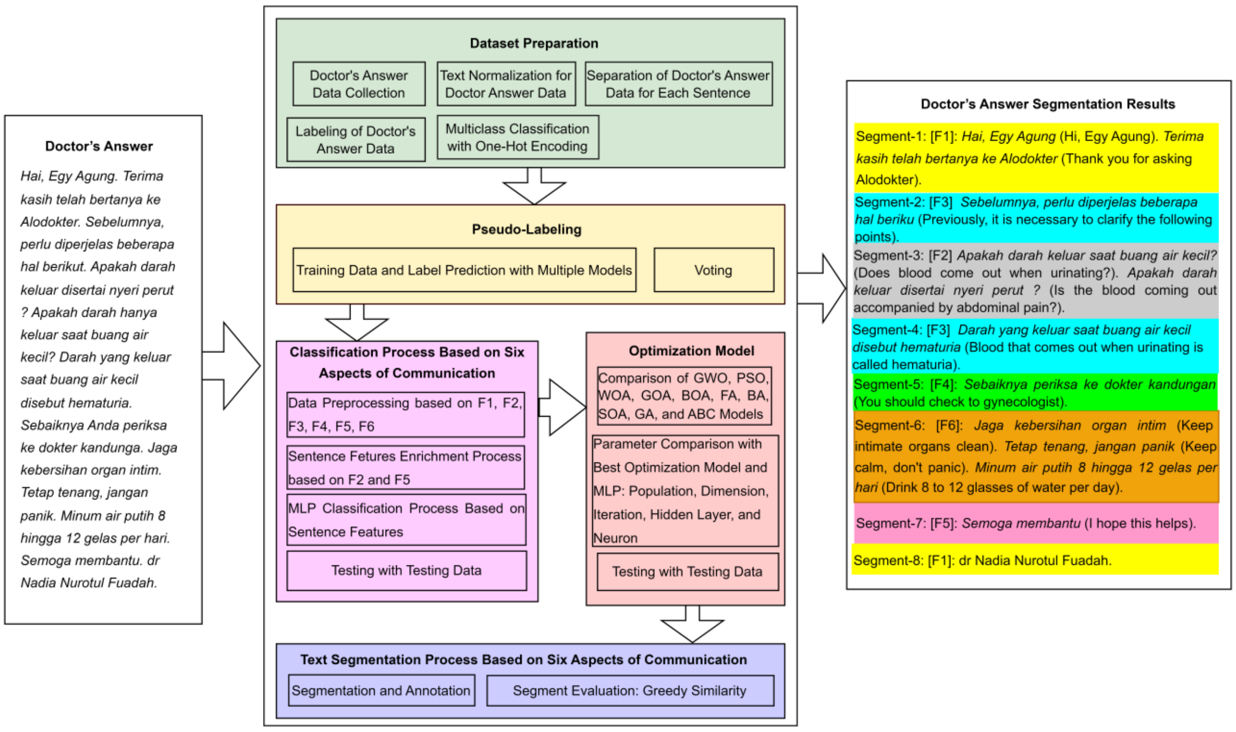
The architectures of XGBoost, CatBoost, DT, LR, and RF referred to Abbas's study [61], which proposed the EnsCL-CatBoost model to improve accuracy in software requirements classification using unlabeled data. The EnsCL-CatBoost model was compared with XGBoost, CatBoost, DT, LR, and RF, and yielded the best results. XGBoost was an efficient and fast model for building gradient-boosted decision trees using parallel tree boosting techniques, which improved accuracy in regression, classification, and ranking tasks. CatBoost handled categorical features without One-Hot Encoding, managed missing data, reduced overfitting, and achieved high prediction accuracy with a fast-training process.

DT handled nonlinear relationships, was easy to interpret, and was effective for feature selection without requiring data distribution assumptions. LR performed well in classifying software requirements by handling binary and multi-class problems and providing measurable prediction probabilities. RF was suitable for handling complex software data, preventing overfitting, and improving accuracy through bagging techniques and out-of-bag error validation.

The architectures of LGBM, SVM, and NB referred to Alzamzami's research [62], which developed a sophisticated LGBM model for general sentiment analysis on informal short texts. It comprehensively evaluated various classification algorithms and proposed features. LGBM demonstrated the ability to manage high-dimensional data and address data imbalance issues. NB served as an effective probabilistic classifier, particularly in word frequency-based sentiment analysis. SVM was effective in text classification by selecting an optimal hyperplane that maximized the margin between classes. In multi-class scenarios, a one-versus-all approach with a linear kernel was employed to obtain the best results.

The GBM architecture referred to He [63], who conducted aspect-based sentiment analysis using meta-based self-training methods (MSM). The GBM-BERT method utilized unlabeled data. GBM enabled each part of the task-specific tower to filter irrelevant information and combine helpful information from other components, thereby improving the model's overall performance.

The KNN architecture was based on Li's study [64], which proposed the augmented retrieval K-nearest neighbors (KRA) method for a deep learning-based text classification model.


Dataset Preparation
Doctor's Answer Data Collection
Text Normalization for Doctor Answer Data
Separation of Doctor's Answer Data for Each Sentence
Labeling of Doctor's Answer Data
Multiclass Classification with One-Hot Encoding
Pseudo-Labeling
Training Data and Label Prediction with Multiple Models
Voting
Classification Process Based on Six Aspects of Communication
Data Preprocessing based on F1, F2, F3, F4, F5, F6
Sentence Fetures Enrichment Process based on F2 and F5
MLP Classification Process Based on Sentence Features
Testing with Testing Data
Optimization Model
Comparison of GWO, PSO, WOA, GOA, BOA, FA, BA, SOA, GA, and ABC Models
Parameter Comparison with Best Optimization Model and MLP: Population, Dimension, Iteration, Hidden Layer, and Neuron
Testing with Testing Data
Text Segmentation Process Based on Six Aspects of Communication
Segmentation and Annotation
Segment Evaluation: Greedy Similarity
Doctor's Answer
Doctor's Answer Segmentation Results

Hai, Egy Agung. Terima kasih telah bertanya ke Alodokter. Sebelumnya, perlu diperjelas beberapa hal berikut. Apakah darah keluar disertai nyeri perut ? Apakah darah hanya keluar saat buang air kecil? Darah yang keluar saat buang air kecil disebut hematuria. Sebaiknya Anda periksa ke dokter kandunga. Jaga kebersihan organ intim. Tetap tenang, jangan panik. Minum air putih 8 hingga 12 gelas per hari. Semoga membantu. dr Nadia Nurotul Fuadah.

Segment-1: [F1]: *Hai, Egy Agung* (Hi, Egy Agung). *Terima kasih telah bertanya ke Alodokter* (Thank you for asking Alodokter).

Segment-2: [F3] *Sebelumnya, perlu diperjelas beberapa hal beriku* (Previously, it is necessary to clarify the following points).

Segment-3: [F2] *Apakah darah keluar saat buang air kecil?* (Does blood come out when urinating?). *Apakah darah keluar disertai nyeri perut ?* (Is the blood coming out accompanied by abdominal pain?).

Segment-4: [F3] *Darah yang keluar saat buang air kecil disebut hematuria* (Blood that comes out when urinating is called hematuria).

Segment-5: [F4]: *Sebaiknya periksa ke dokter kandungan* (You should check to gynecologist).

Segment-6: [F6]: *Jaga kebersihan organ intim* (Keep intimate organs clean). *Tetap tenang, jangan panik* (Keep calm, don't panic). *Minum air putih 8 hingga 12 gelas per hari* (Drink 8 to 12 glasses of water per day).

Segment-7: [F5]: *Semoga membantu* (I hope this helps).

Segment-8: [F1]: dr Nadia Nurotul Fuadah.

Figure 1: General architecture of MLPSentFeat method.

KNN leveraged data similarity to enhance prediction performance and helped language models understand context by referencing similar training examples.

These models provided multiple advantages, several of which had been applied to unlabeled and short text data. Therefore, the authors used these references as a foundation for the pseudo-labeling process on doctor's answer data, which typically consisted of longer sentences.

The classifier voting process referred to Elsaeed's research [65], which detected fake news on social media by combining the classification results of multiple models using voting classifiers. Hard voting improved prediction accuracy by aggregating the outputs of several models and selecting the final label based on the majority vote per instance. The authors adopted this voting classifier approach to integrate the results from multiple models and determine the best label for each sentence. This ensemble learning strategy, also highlighted in Elsaeed’s work, aimed to improve predictive performance. However, prior studies had not combined these classification methods for labeling unlabeled data, suggesting a novel contribution in up-to-date research modeling.

2.5. Optimization methods

Previous studies modified various optimization algorithms to optimize the weights and biases of MLP methods across different domains [66][67][23][68][69]. In addition, the number of hidden layers and neurons in MLP could be optimized using data mining approaches [70]. The references indicated that no prior study compared optimization algorithms specifically for tuning the number of hidden layers and neurons in the MLP method.

Studies also compared optimization algorithms in various domains [71][72][73][74][[75]. However, these studies did not explore optimization algorithms for MLP-based optimization.

This study performed optimization by tuning the number of hidden layers and neurons in the MLP model through the comparison of several metaheuristic algorithms—aimed at identifying optimal parameters through objective evaluation [76]. The compared algorithms included Grey Wolf Optimizer (GWO), Particle Swarm Optimization (PSO), Whale Optimization Algorithm (WOA), Grasshopper Optimization Algorithm (GOA), Butterfly Optimization Algorithm (BOA), Firefly Algorithm (FA), Bat Algorithm (BA), Seagull Optimization Algorithm (SOA), Genetic Algorithm (GA), and Artificial Bee Colony (ABC). The architectural foundation of each algorithm referred to previously established research.

The GWO architecture referred to [77], which mimicked the leadership hierarchy and hunting behavior of gray wolves (Canis lupus). PSO architecture referred to [78], which was inspired by the flocking behavior of birds and fish. The model integrated artificial life (A-life), group behavior theory, and evolutionary computing. WOA architecture, based on [79], simulated the bubble-net hunting strategies of humpback whales. GOA architecture, based on [80], mathematically modeled locust herd movement to solve optimization problems. BOA architecture referred to [81], which emulated butterflies’ mating and foraging behavior using olfactory-based guidance for global optimization. FA architecture, based on [82], was inspired by the flickering patterns of fireflies and employed rules of attraction based on brightness. BA architecture, based on [83], was inspired by the echolocation behavior of microbats to detect prey and navigate in darkness. SOA architecture referred to [84], which simulated seagull migration and attack behaviors for improved search space exploration. GA architecture, based on [85], modeled natural evolution through selection, crossover, and mutation with solutions represented as binary gene chromosomes. ABC architecture referred to [86], which modeled the division of roles in artificial bee colonies, where each worker bee represented a nectar source.

3 MLPSentFeat on text segmentation

This section explains the MLPSentFeat model, which has five stages shown in Figure 1, i.e., doctor’s answer dataset preparation, pseudo-labeling, classification, model optimization, and text segmentation. The pseudo-labeling stage is for labeling new data with several machine learning methods, the classification stage is for testing model performance on new data, the text segmentation stage is for forming segments from the classification results, and the MLP optimization model is for optimizing the number of neurons and hidden layers of the MLP. The input for the MLPSentFeat model is the doctor's answer data, while the output is the doctor's answer data, segmented based on aspects of doctor communication. In general, the flow of this research is as follows: first, the doctor's answer data is processed into the pseudo-labeling stage, where there is no testing of the pseudo-labeling results. Second, the pseudo-labeling results are processed into the classification stage. Third, the classification results are processed into the text segmentation stage. Fourth, the classification results can also be processed into the model optimization stage to obtain the optimal number of neurons and hidden layers. Then, the optimal number of neurons and hidden layers is applied to all the doctor's answer data through the classification stage. Next, it can be processed into the text segmentation stage again.

Some terms to pay attention to include sentence, data, and segment. Sentences consist of one or more words. The data consists of several sentences. For example, the input data for the doctor's answers in Figure 1 is referred to as 1 data. Segments are small groups that are formed from the doctor's answer data. One dataset can contain one or more segments. One segment can contain one or more sentences. These segments are formed based on six aspects of doctor-patient communication.

3.1. Doctor’s answer dataset preparation

Data from the doctor's answers was prepared using several steps, shown in Figure 2. This section explains the preparation of the dataset, starting from data collection until the data is ready to be processed using a method. This data preparation has five processes: doctor's answer data collection, text normalization for doctor's answer data, separation of doctor's answer data for each sentence, labeling of doctor's answer data, and multiclass classification with one-hot encoding.

A. Dataset collection

This research uses three datasets of doctors' answers from various OHC platforms, namely dataset 1 (DS1), dataset 2 (DS2), and dataset 3 (DS3). DS1 was taken from the Alodokter platform, adopted data published by Juanita [87] spanning December 8, 2014, to February 28, 2021, can be accessed at

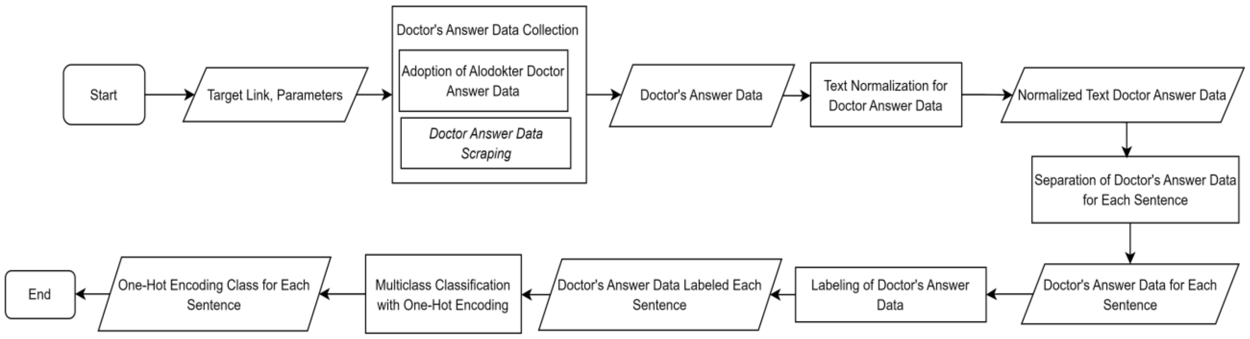

Start
Target Link, Parameters
Doctor's Answer Data Collection
Adoption of Alodokter Doctor Answer Data
Doctor Answer Data Scraping
Doctor's Answer Data
Text Normalization for Doctor Answer Data
Normalized Text Doctor Answer Data
Separation of Doctor's Answer Data for Each Sentence
Doctor's Answer Data for Each Sentence
Labeling of Doctor's Answer Data
Doctor's Answer Data Labeled Each Sentence
Multiclass Classification with One-Hot Encoding
One-Hot Encoding Class for Each Sentence
End

Figure 2: Doctors' answer dataset preparation.

https://data.mendeley.com/datasets/p8d5bynh3m/1, file name Indo-Online Health Consultation-Multilabel-Raw.csv. DS2 and DS3 were taken by scraping data from the OHC platform. DS2 is taken from the Alodokter platform, https://www.alodokter.com/komunitas/diskusi/penyakit, from February 29, 2021, to November 2024. DS3 is taken from several OHC platforms, namely HelloSehat (https://hellosehat.com/community/), detikHealth (https://health.detik.com/konsultasi), and Mayapada Hospital (https://mayapadahospital.com/). The three OHC platforms have a small number of doctors' answers, so the three doctors' answers are combined from January 2024 to December 2024. DS1 has 15441 data with 306955 sentences, DS2 has 36247 data with 495478 sentences, and DS3 has 3111 data with 53063 sentences. All doctors' answers are in Indonesian. A doctor's complete answer text consists of several sentences that provide medical information to the reader regarding the user's condition. This complete doctor's answer text is used as input in the MLPSentFeat model. The doctor's answer data is stored in CSV format, where each doctor's answer text is placed in a single cell containing multiple sentences.

The gathering aspect of doctors' answers on various platforms is very small; the Alodokter platform has a slightly higher number than other platforms. The response to the emotional aspect has three types of sentences: sympathy, empathy, and hope. This type of sympathy and empathy sentence has a small number of sentences, while the type of sentence of hope has a large number. Responding to emotions in the doctor's answer data from other platforms is more varied than Alodokter's. The aspect of responding to emotions, especially the sentences of sympathy and empathy, is of particular concern to the writer, because different annotators give this aspect different labels. First, the doctor's answer data is annotated by the midwife. This type of sentence of hope is labeled as fostering the relationship. Second, the doctor's answer data is annotated by the doctor. This type of sentence of hope is labeled as responding to the emotion. So, initially, the response to the emotion aspect has a small number, then it increases to a large number after getting additional types of hope sentences. Here, the MLPSentFeat model still pays attention to the types of empathy and sympathy sentences, so sentence features are also formed from the response to the emotion aspect.

Doctors rarely reveal the information-gathering aspect because this consultation is asynchronous. So, doctors do not want to dig up information from patients. Apart from that, some platforms do not provide a place to reply to doctors' questions. Doctors rarely express sympathy and empathy because patients do not explain their illnesses in detail. In addition, it is not easy to understand other people's emotions through text [5].

B. Text normalization for doctor answer data

Text normalization is an essential step in text standardization, reducing variation and ensuring consistency in format before being processed by the model. In this study, the doctor's answer data is normalized into two categories: general normalization and specific normalization. These text normalization techniques are performed using simple text normalization methods.

General normalization focuses on changes made to the basic structure of the text to improve consistency and uniformity. Examples of general text normalization applied to the doctor's answer data include the separation of hyphens (-), slashes (/), question marks (?), and exclamation marks (!) from the words they follow. This step separates the words before and after these marks during the tokenization process. If this step is not performed, the words before and after these marks would merge into one word, potentially altering the meaning of the word. For example, 'rata-rata' is changed to 'rata - rata', 'posisi tidur/horizontal' is changed to 'posisi tidur / horizontal'. Additionally, periods (.) following numbers, acronyms, or titles are removed. This step aims to place the period at the end of a sentence, as the initial input of this study is the intact doctor's answer data, which is separated by sentences by identifying the period as one of the sentence-ending markers. For example, *'1. Minum air putih* (1. Drink water)' is changed to *'1 Minum air putih'*, '10.000' is changed to '10000', *'dr. Syifa'* is changed to 'dr Syifa' (example of writing a doctor's title), and *'Spesialis Penyakit Dalam (Sp.PD)* (Internal Medicine Specialist)' is changed to *'Spesialis Penyakit Dalam (Sp PD)*' (example of writing an acronym). These steps ensure that the text is more consistent and can be processed more effectively by the model. In addition, there is also a step for correcting spelling errors. This step aims to ensure that a word is recognized as the same word, rather than being treated as a different one by the system. For example, a misspelled word such as '*inum'* is corrected to '*minum* (drink)'. If such errors are not corrected, the system may interpret them as two distinct entities, which can lead to inaccurate analysis.

Specific normalization focuses on more specific changes related to the context or information present in the text, such as the processing of certain sentences or words to improve analysis accuracy. The goal of this specific normalization is to standardize words in the doctor's answers so that they can be classified correctly according to their communication aspects. Additionally, this specific text normalization aims to standardize the way steps in various instructions or tips are written. In some cases, these steps are often written in one long sentence, which can make it difficult for the model to recognize and classify sentences according to the relevant communication aspect. Therefore, these steps are organized into separate sentences to make it easier for the model to clearly recognize each step and identify the intended communication aspect. As a result, the model can classify the sentences more accurately and efficiently. In the initial experiment, when the model used very limited data, without specific normalization, the model struggled with classification. This normalization helps reduce data variation, allowing the model to classify more accurately despite the limited data available.

Steps in specific normalization include, first, shortening long sentences by separating clauses in those sentences into shorter, more manageable sentences. Additionally, this step is crucial because a sentence often contains several communication aspects, whereas each sentence should ideally contain one clear communication aspect. This shortening is performed carefully to ensure that the meaning of the sentence remains intact. For example, *'Beberapa Tips yang Anda dapat lakukan untuk mencegah terjadinya ISK atau infeksi pada saluran kemih yaitu Minum air putih minimal 2-liter perhari Jangan menahan keinginan BAK atau Buang Air Kecil* (Some tips that you can do to prevent UTI or urinary tract infections are: Drink at least 2 liters of water per day Don't hold back the urge to urinate)'. This sentence, which is about self-medication tips and consists of three clauses, is split into three sentences. Inaddition, this sentence also has two communication aspects: providing information and enabling disease and treatment-related behavior, thus it must be separated into several shorter sentences according to the clauses. The sentence is changed to, Sentence 1: *'Beberapa Tips yang Anda dapat lakukan untuk mencegah terjadinya ISK atau infeksi pada saluran kemih yaitu.*', sentence 2: '*Minum air putih minimal 2-liter perhari.*', sentence 3: '*Jangan menahan keinginan BAK atau Buang Air Kecil.*' Sentence 1 includes the information-giving aspect, while sentences 2 and 3 include the aspects of enabling the occurrence of disease and behavior related to treatment.

The second step is the process of synonym standardization to ensure consistent word usage in the text. For example, in the aforementioned sentence, acronyms ISK and BAK are added. This step aims to reduce the model's tendency to associate acronyms from one language with English, considering that some word embedding, classification, or segmentation methods often use English corpora in their training process. Furthermore, this step aims to reduce ambiguity in acronyms, improve classification and segmentation accuracy, and ensure consistency in the interpretation of terms used. Thus, the model can be trained more effectively and accurately in understanding the appropriate context.

The third step is the process of standardizing words for different synonyms. The goal of synonym standardization is to reduce variation in the data and ensure consistency in text processing, allowing the model or language processing system to more easily understand the same concept even when expressed with different words. For example, the word '*menstruasi* (menstruation)' has the synonym '*haid* (menstruation)'. Both words are standardized to 'menstruasi'. The choice of the word 'menstruasi' as the standardized form, compared to 'haid', is based on several reasons. The word '*menstruasi'* is more commonly used in formal and technical contexts, particularly in medical and academic literature, while '*haid'* is more frequently used in everyday conversation. Thus, using 'menstruasi' ensures consistency in the use of standardized terminology, especially in scientific and medical writing, which is crucial for maintaining the quality and clarity of information. By standardizing synonyms, such as equating 'menstruasi' and '*haid'*, the model will not consider them as two different entities.

The fourth step is the process of standardizing the writing of specific words. For example, the word '*terimakasih* (thank you)' is often written as a single word, but some doctors write it as two separate words, '*terima kasih'*. The standardized form of this word is the one written separately, so in this process, the writer standardizes the word by writing it separately. This step ensures that the model does not perceive it as two different entities, which could lead to confusion in understanding the context. Additionally, it helps to improve consistency in the data and facilitates the tokenization and text analysis process, allowing the model to recognize the word more accurately.

The fifth step is the process of using consistent acronyms for the same term. For example, the word '*Tuberkulosis* (Tuberculosis)' can be abbreviated as 'TBC' or 'TB'. The author chooses the acronym 'TBC' for this standardization because 'TBC' is more commonly used in formal writing in Indonesia, even in everyday conversation. The goal of this step is to ensure consistency in terminology usage, so the model does not consider it as two different entities.

The sixth step is the process of separating combined question sentences into individual question sentences. For example, the sentence 'apakah ada demam, terasa mual, muntah (Is there a fever, nausea, vomiting)' consists of one sentence containing three symptoms that need to be identified. The sentence is then converted into three separate sentences: Sentence 1: '*Apakah ada demam ?* (Is there a fever ?)', Sentence 2: '*Apakah terasa mual ?* (Do you feel nauseous ?)', and Sentence 3: '*Apakah ada muntah ?* (Is there vomiting?)'. This step is performed to increase the number of sentences with the gathering information aspect, considering that this aspect has relatively few sentences.

The seventh step is the process of adding question words and question marks (?) to question sentences. For example, a sentence that initially does not have a question mark, such as '*Apakah anda menjadi tidak bisa belajar atau bekerja* (Are you unable to study or work)', is changed into a sentence with a question mark, '*Apakah anda menjadi tidak bisa belajar atau bekerja ?'*. This step is intended to standardize question sentences, where a question sentence is generally initiated by a question word and ends with a question mark (?). Additionally, this step aims to facilitate the classification of the gathering information aspect, considering that this aspect has a limited number of sentences and a variety of question sentence forms. Without standardizing the question sentence forms, the model would struggle to classify the doctor's answer sentences into this aspect.

The eighth step is paraphrasing sentences. For example, the sentence '*Bau bagaimana yang Anda maksudkan ?'* is revised to '*Bagaimana bau yang Anda maksud ?* (What smell do you mean?)'. The original sentence is an interrogative sentence; however, it does not adhere to the standard rules of interrogative sentence construction, which require that a question begin with a question word. Such a sentence is more appropriate in informal or semi-formal spoken contexts. Therefore, the original sentence is reformulated to align with the conventions of formal written language. The purpose of this step is to improve sentence structure in accordance with standard writing conventions, enhance readability, and ensure that the meaning of the sentence is preserved despite changes in form. Paraphrasing is also performed to adjust the language style from informal or spoken forms to more formal and standardized written forms.

The ninth step involves adding punctuation marks to sentences that lack them or correcting the placement of punctuation that is inaccurately positioned. This step is intended to improve readability, clarify sentence structure, eliminate ambiguity, accurately mark sentence boundaries, and ensure compliance with standard Indonesian writing conventions. As an example, the sentence used in Step One can be referenced. The sentence consists of several phrases, but there are no punctuation marks separating them. Therefore, a comma can be inserted to distinguish the individual phrases. Additionally, the sentence does not end with a period, so a full stop should be added to mark the end of the sentence. The revised version is as follows: '*Beberapa tips yang Anda dapat lakukan untuk mencegah terjadinya ISK atau infeksi pada saluran kemih yaitu minum air putih minimal 2 liter per hari, jangan menahan keinginan BAK atau Buang Air Kecil.*' (Some tips that you can do to prevent UTI or urinary tract infections are: drink at least 2 liters of water per day, don't hold back the urge to urinate.).

These normalization steps—both general and domain-specific—are essential to ensure that the doctor's response data can be easily interpreted by both the model and human readers. The purpose of this normalization is not only to facilitate the model's understanding of the text and its ability to perform accurate classification, but also to ensure that the data can be comprehended by medical annotators who are responsible for labeling the data. Data that is not clearly readable can hinder the labeling process by medical annotators. By ensuring that the data is easily understandable by readers, normalization also supports the model in comprehending the meaning of the text and classifying the data more accurately.

General text normalization is applied to the data from DS1, DS2, and DS3. The basic structure of the text must be consistent across these datasets, as this consistency enables the model to recognize relevant patterns and features more effectively, increases accuracy in classification and analysis, and facilitates the tokenization and processing of the data. Moreover, this consistency reduces the potential for the model's confusion in identifying the true meaning, thereby improving the overall performance of the model. Specific text normalization is applied to the DS1 data because it contains many spelling errors and structural inconsistencies. Therefore, specific normalization is necessary to correct these spelling errors and improve the consistency of the text, allowing the model to process the data more effectively. This step also allows for assessing the impact of normalization on the model's performance in classification and analysis.

C. Separation of doctor's answer data for each sentence

The process of separating the doctor's answer data per sentence aims to divide the doctor's answer text into individual sentences. It means that each doctor's answer, initially located in a single cell, will be separated into one sentence per cell. This process is carried out because Python (the programming language used to build this model) reads the content in a single cell as one sentence, even though the cell contains multiple sentences. Since this study performs text segmentation at the sentence level, it is necessary to separate the sentences to process each individually.

The text segmentation process requires separating multiple sentences to be processed effectively. Working with just one sentence is insufficient, as these sentences will be divided into smaller segments. Therefore, to meet the segmentation processing requirements involving multiple sentences and Python's reading of cells, the author separates each sentence in the doctor's answer data into separate cells originally stored in one cell. Sentence identification is done by detecting period (.), question mark (?), and exclamation mark (!) as indicators of sentence boundaries [88]. This separation process is carried out automatically by identifying these three markers, and each sentence is stored in a separate cell. The doctor's answer data separation per sentence is applied to the entire DS1, DS2, and DS3 data.

Table 3: Distribution of doctor response data based on aspects of communication.

Data	Labelled Data								Unlabelled Data	
	Total Data	Total Sentences	F1	F2	F3	F4	F5	F6	Total Data	Total Sentences
DS1	5410	106231	16147	977	55824	9270	5446	18567	10031	200724
DS2	8000	110562	16555	274	48142	8246	8000	29345	28247	384916
DS3	1000	23280	1987	73	14460	1818	753	4189	2111	29783

Thus, the doctor's answer text, initially stored as a single unit in one cell, is now separated into multiple cells, each containing one sentence. This sentence segmentation is crucial, as the relatively short length of the doctor's answer text often leads to changes in communication aspects at the sentence level. By separating the doctor's answers based on sentences, the model can more easily recognize different communication aspects in each sentence, which would not be possible if the text were left as one long sentence. For example, changes in communication aspects at the sentence level can be observed in the previous explanation (point B), where in the example, the first sentence falls under the aspect of providing information. In contrast, the second and third sentences are categorized as enabling disease and treatment-related behavior. Therefore, by dividing the text into separate sentences in different cells, each sentence can be processed as a smaller and independent unit. This action allows the model to recognize and analyze each sentence in the doctor's answer text more accurately.

D. Labeling of doctor's answer data

The data labeling process aims to label each sentence in the doctor's response. Each sentence within the text is analyzed to identify the relevant aspect of communication. This labeling procedure determines that each sentence is assigned only one label for the communication aspect. The single-label-per-sentence approach is applied to avoid interpretative conflicts among annotators. This approach is essential to ensure data consistency and reliability, particularly when a sentence contains multiple meanings that each annotator may interpret differently. Moreover, manual labeling requires intensive cognitive processing. Annotators must comprehend the meaning of each sentence, recall the criteria for each communication aspect, and determine the most appropriate label. If annotators are required to assign more than one label, their cognitive load increases significantly, potentially affecting the accuracy and consistency of the labeling outcomes. Equally important, this approach also considers time efficiency, considering that the annotators are practitioners such as doctors with limited time availability. Restricting labeling to a single dominant aspect per sentence allows the annotation process to proceed more quickly and in a more focused manner while still maintaining the quality of the results.

The determination of these communication aspects is thoroughly explained in Subsection 2.1, which outlines the criteria for categorizing sentences under the appropriate communication aspect. Once the corresponding aspect is identified, each sentence is labeled F1 through F6, representing the various categories of communication aspects described in Subsection 2.1. This labeling is performed at the sentence level across the entire doctor's response. For example, Sentence 1 in Figure 1 is a greeting by the doctor directed at the patient or user. This greeting corresponds to the “fostering the relationship” aspect, wherein greeting the patient is one of the doctor’s responsibilities; therefore, Sentence 1 is labeled as F1. Sentence 4 is a question posed by the doctor to the user, aligning with the “gathering information” aspect, as this category involves the doctor asking open-ended questions; hence, it is labeled as F2. Sentence 6 is an informational statement from the doctor to the user, representing the “providing information” aspect, as this includes the doctor’s role in sharing relevant information with the patient; thus, it is labeled F3. Sentence 7 suggests that the user consult an obstetrician in person. This approach aligns with the “decision-making” aspect, where the doctor’s responsibility includes planning collaborative actions—such as referring patients to other doctors—thereby earning a label of F4. Sentence 8 is a self-care recommendation that fits the “enabling disease- and treatment-related behavior” aspect, where doctors advise actions supporting appropriate health-related behavior. Thus, it is labeled as F6. Sentence 11 expresses the doctor’s hope that the information provided will be helpful. Such expressions may help ease the patient’s emotions by making them feel acknowledged and are part of the “responding to emotions” aspect; accordingly, Sentence 11 is labeled as F5.

The doctor's responses were labeled by medical annotators, consisting of five general practitioners. Each physician annotated different subsets of the dataset during different periods, ensuring no annotator labeled the same data. This approach was necessary due to the large volume of unlabeled data requiring annotation. The process was conducted manually, with annotators carefully examining each sentence and assigning the most appropriate label based on the contextual communication content. The labeling results were manually verified by comparing them against predefined labeling guidelines for each category. In addition, the labeled data were re-reviewed by a more experienced annotator to ensure that all labels adhered to the defined standards.

Based on the labeling outcomes, the occurrence of communication aspects within a single doctor response does not follow a fixed order from F1 to F6 (i.e., from fostering the relationship to responding to emotions), and multiple aspects may appear more than once within a single response. Moreover, not all responses encompass all six communication aspects. In many cases, only a subset of the aspects is present. This variation is likely influenced by the individual doctor’s writing style,

$$LR\ of\ Feature\ X = \frac{Frequency\ of\ X\ in\ CorpusL}{(Frequency\ of\ X\ in\ CorpusL\ +\ Frequency\ of\ X\ in\ CorpusNon)} \quad (1)$$

$$Baseline\ LR = \frac{Number\ of\ Words\ in\ CorpusL}{(Number\ of\ Words\ in\ CorpusL\ +\ Number\ of\ Words\ in\ CorpusNon)} \quad (2)$$

$$G(D,t) = -\sum_{i=1}^{m} P(Ci)logP(Ci) + P(t)\sum_{i=1}^{m} P(Ci|t)logP(Ci|t) + P(\bar{t})\sum_{i=1}^{m} P(Ci|\bar{t})logP(Ci|\bar{t}) \quad (3)$$

intuition, and clinical judgment. For example, in dataset DS1, only 294 out of 5,410 responses contained all six aspects, meaning that approximately 94.56% of the data did not cover the full range of communication categories. This dataset indicates that only a few responses are comprehensive and aligned with all six communication aspects. These findings reflect the contextual and selective nature of doctor-patient communication in clinical practice, which often does not address every ideal aspect of communication. A similar pattern was observed in DS2 and DS3. In DS2, only 130 out of 8,000 responses (1.63%) included all six aspects. In DS3, only 2 out of 1,000 responses (0.2%) met this complete communication criteria. These results consistently demonstrate that doctor communication across all three datasets tends to be partial, varied, and seldom incorporates all predefined communication aspects.

The total number of labeled and unlabeled responses and the distribution of sentences across labeled communication aspects are presented in Table 3. Although the dataset contains a relatively large number of sentences, segmentation requires complete doctor responses. As shown in Table 3, the distribution of labeled data across DS1, DS2, and DS3 indicates a significant imbalance among communication aspects. The "providing information" aspect (F3) consistently dominates all datasets, with 55,824 sentences in DS1 and 48,142 in DS2—far exceeding other aspects. In contrast, the "gathering information" (F2) and "responding to emotions" (F5) aspects are underrepresented, particularly in DS2 (only 274 and 8,000 sentences, respectively), and even fewer in DS3. This imbalance suggests that doctors' responses are more focused on delivering information than on elicitation or emotional engagement.

In terms of volume, DS2 contains the largest number of labeled and unlabeled data, indicating its strategic role in training and evaluation. Although DS1 is smaller, it remains relevant due to its balanced ratio between labeled and unlabeled data. Even though DS3 has the most limited size, it is valuable for use as a separate evaluation set to test model generalizability on smaller-scale data.

Initially, the "responding to emotions" aspect of the dataset primarily included empathetic and sympathetic sentences [5]. However, upon further review, additional sentences in the form of "wish" expressions were added to this category. Since doctors frequently included wishes in their responses, these expressions were initially mislabeled as "fostering the relationship." However, they were later reclassified under "responding to emotions" to more accurately reflect their communicative function.

E. Multiclass representation using one-hot encoding

This process aims to convert class labels in a dataset containing more than two classes (multiclass classification) into a binary vector format using the one-hot encoding technique. As described, each sentence in the doctor's response is assigned a single communication aspect label. The classification involves six distinct communication aspects, symbolized as F1 through F6 (further explanation can be found in Subsection 2.1). Accordingly, each sentence is categorized into one of these six aspects. This type of classification is known as multiclass classification, where each sample (in this case, a sentence) is assigned to one class out of several available options [89]. In other words, once a sentence is assigned to one label, it cannot simultaneously belong to another. For example, if sentence-1 in Figure 1 is labeled as F1, it will not be assigned to any remaining labels (F2 to F6).

The one-hot encoding technique transforms the assigned labels for each sentence in the doctor's response into binary vectors. One-hot encoding represents each class in a multiclass classification task as a binary vector, where the value 1 indicates the selected class and the value 0 indicates the remaining classes [90][91]. One-hot encoding was chosen because it allows categorical labels to be numerically represented without assuming any ordinal relationship or hierarchical weight among the classes. An important consideration in multiclass classification tasks involving mutually exclusive categories, such as the six communication aspects (F1 through F6), which are conceptually equal and independent. Moreover, this representation enables direct comparison between predicted and actual labels, thereby supporting model performance evaluation based on class-specific metrics such as accuracy, precision, recall, and F1-score.

In this representation, each communication aspect class (F1 to F6) is mapped to a column, with a value of 1 assigned to the column corresponding to the true label and a value of 0 assigned to all others. For instance, if Sentence 1 in Figure 1 is labeled as F1, the F1 column will have a value of 1, while the columns for F2 to F6 will have a value of 0, resulting in a one-hot vector of [1, 0, 0, 0, 0, 0]. Similarly, if Sentence 6 is labeled as F3, its one-hot representation will be [0, 0, 1, 0, 0, 0], where the F3 column holds the value 1 and the remaining columns (F1, F2, F4, F5, and F6) each hold the value 0.

The one-hot encoding scheme represented all sentences in the DS1, DS2, and DS3 datasets. This format was consistently applied throughout the classification model's training and evaluation stages. The one-hot representation facilitated direct comparison between predicted and actual labels and supported the computation of performance metrics such as accuracy, F1-score, precision, and recall for each individual class.

3.2. Pseudo-labeling

This stage was designed to expand the data by utilizing new data that has not been labeled. This data labeling was done by pseudo-labeling using a voting classifier from several machine learning models. This data addition was carried out to overcome overfitting, the first research problem. Using pseudo-labeling is necessary because manual labeling by annotators requires significant time, costs, and effort [92][93]. Using multiple methods for pseudo-labeling can reduce labeling errors by exploiting the advantages of each method to increase correct labels [93]. The pseudo-labeling stage in Figure 3 has four processes: separating the doctor's answer data labeled by the doctor with k-fold, preprocessing training data, training data and label prediction, and voting.

The input for the pseudo-labeling stage, namely labeled data and unlabeled data, can be seen in Table 3. The output for the pseudo-labeling stage is labeled unlabeled data.

3.2.1. Separating doctor's answer data labeled with K-Fold

The labeled doctor's answer data was split into training and test data, 80:20, using k-fold cross validation, which was rotated five times. Training data is for testing the model, while testing data is for testing the model. Cross-validation can overcome overfitting because it divides data fairly between training and testing [65][94]. K-fold cross-validation refers to [65], where the testing set is usually smaller than the training set to get better performance. K-fold cross validation divides the data into k equal subsets (folds), the model is iteratively trained in k-1 folds and validated against the remaining folds [95]. The k-fold cross-validation method can help reduce uncertainty in data sharing and prevent overfitting (the model fits the training data too well, so it is less accurate on new data) [94].

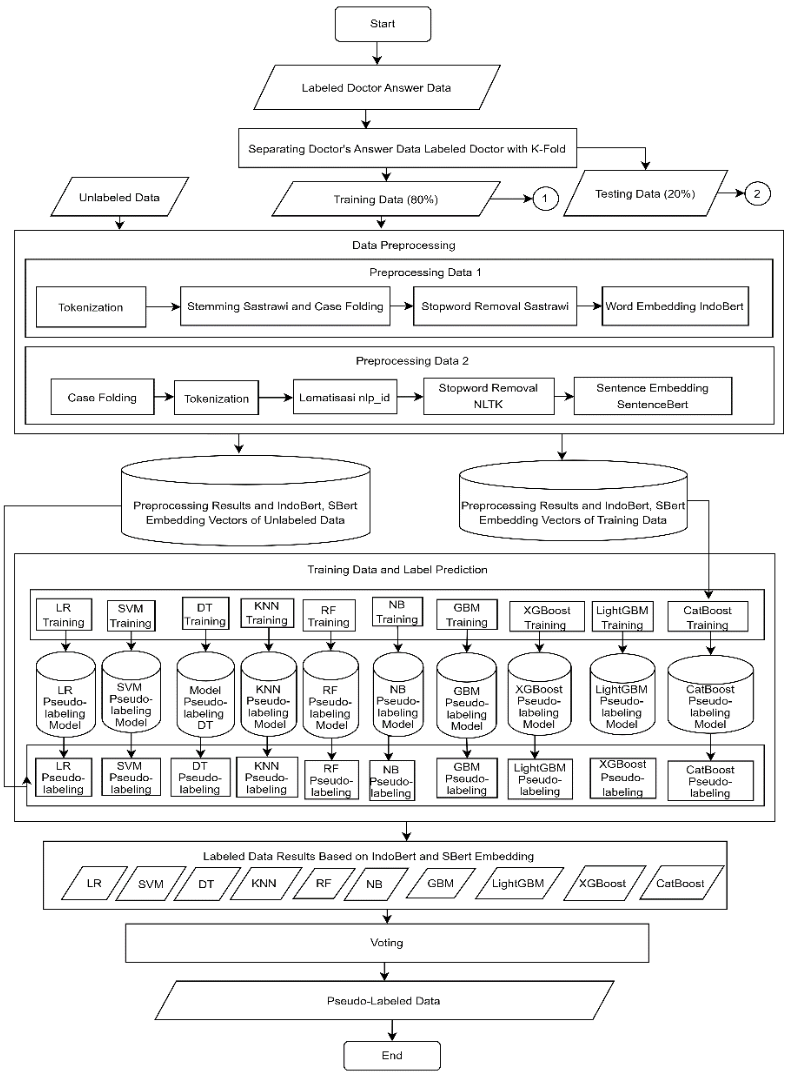


Figure 3: Pseudo-labeling architecture.

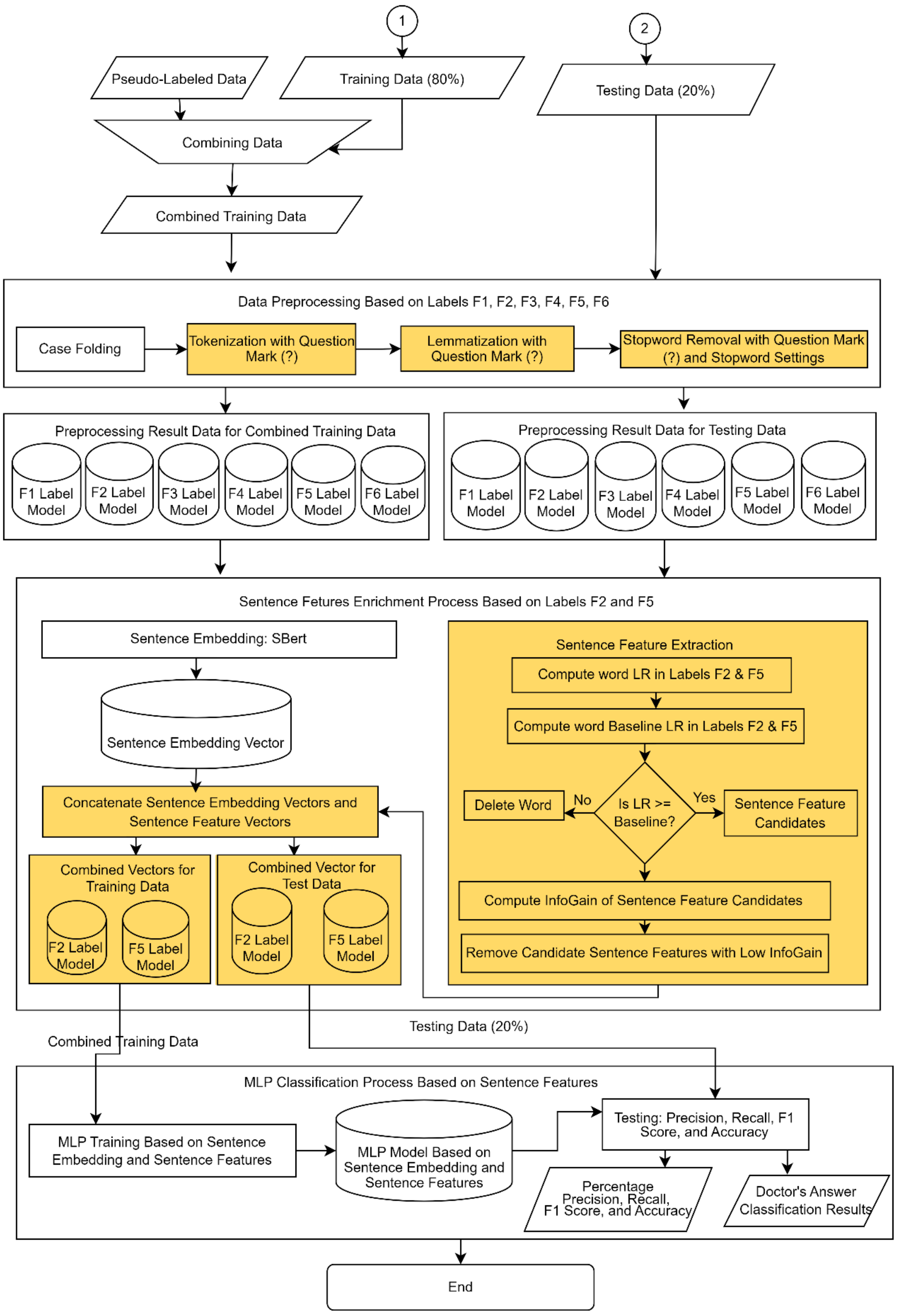

1
2
Pseudo-Labeled Data
Training Data (80%)
Testing Data (20%)
Combining Data
Combined Training Data
Data Preprocessing Based on Labels F1, F2, F3, F4, F5, F6
Case Folding
Tokenization with Question Mark (?)
Lemmatization with Question Mark (?)
Stopword Removal with Question Mark (?) and Stopword Settings
Preprocessing Result Data for Combined Training Data
Preprocessing Result Data for Testing Data
F1 Label Model
F2 Label Model
F3 Label Model
F4 Label Model
F5 Label Model
F6 Label Model
Sentence Fetures Enrichment Process Based on Labels F2 and F5
Sentence Embedding: SBert
Sentence Embedding Vector
Concatenate Sentence Embedding Vectors and Sentence Feature Vectors
Combined Vectors for Training Data
Combined Vector for Test Data
Sentence Feature Extraction
Compute word LR in Labels F2 & F5
Compute word Baseline LR in Labels F2 & F5
Delete Word
No
Is LR >= Baseline?
Yes
Sentence Feature Candidates
Compute InfoGain of Sentence Feature Candidates
Remove Candidate Sentence Features with Low InfoGain
Testing Data (20%)
Combined Training Data
MLP Classification Process Based on Sentence Features
MLP Training Based on Sentence Embedding and Sentence Features
MLP Model Based on Sentence Embedding and Sentence Features
Testing: Precision, Recall, F1 Score, and Accuracy
Percentage Precision, Recall, F1 Score, and Accuracy
Doctor's Answer Classification Results
End

Figure 4: MLPSentFeat architecture.

3.2.2. Data preprocessing for pseudo-labelling process

The training data is used to train the pseudo-labeling model. The training and unlabeled data undergo a pseudo-labeling process: tokenization, root word retrieval, stopword removal, and embedding. The output of this stage is the results of preprocessing and embedding vectors from training and unlabeled data. Preprocessing results have two types: based on IndoBert and SBERT. So, there are four results of this process, namely, preprocessing results and embedding vectors from IndoBert training data, preprocessing results and embedding vectors from SBERT training data, preprocessing results and embedding vectors from IndoBert unlabeled data, and preprocessing results and embedding vectors from SBERT unlabeled data. The training data's preprocessing results and embedding

vectors are used to train the machine learning model. In contrast, the preprocessing results and embedding vectors from unlabeled data are used for the pseudo-labeling process.

The embedding methods used are IndoBert [96] and SBERT, which are applied alternately. IndoBERT is a BERT variant optimized for Indonesian texts [97], providing accurate and efficient classification results [98]. These two embeddings are used because they can capture different sentence features [99].

3.2.3. Training data and label prediction

This training model classifies doctors' answers using multiclass single-label classification. This classification means that the target class is represented in one column with several possible labels, but each sample only has one label assigned [100][101]. So, the doctor's answer data has one target class, which contains six labels for aspects of doctor-patient communication, and each sentence of the doctor's answer is classified into one label.

This stage has two processes: first, training the training data with machine learning methods, LR, SVM, DT, KNN, RF, NB, GBM, Xgboost, LGBM, and CatBoost models with SBERT and IndoBert embedding. Second, the pseudo-labeling process uses the pseudo-labeling model. The result of this stage is labeled data with a total of 20 labels, obtained based on the training model, consisting of 10 labels with SBERT embedding and 10 labels with IndoBert embedding.

3.2.4. Voting

Determining the final label results for each doctor's answer sentence is done by voting based on the labeling results of each model. The pseudo-labeling results of 20 models were combined, then counted, and the majority vote was taken. The label with the most votes is the final label for the doctor's answer sentence. For example, a sentence labels F3 from 15 models and F4 from 5. Because label F3 has the most significant number, the final label chosen for the sentence is F3. This final label is the pseudo-labeling result used for the following model process.

3.3. Classification process based on six aspects of communication

This classification stage uses the MLPSentFeat model to see the model's performance in classifying doctors' answer data using communication aspects. The classification stage architecture is shown in Figure 4, which has three processes: data preprocessing, sentence features enrichment process, and text classification with the MLPSentFeat model. The colored part is the latest version of this research model, namely the preprocessing settings based on labels or communication aspects, adding sentence features based on labels or communication aspects, and combining sentence embedding vectors with sentence features. The input for the classification stage is labeled data from the pseudo-labeling process, training data is 80% of the labeled data, and test data is 20% of the labeled data. Labeled data from the pseudo-labeling process is combined with training data, as much as 80% of the labeled data, into training data, called combined training data.

Meanwhile, test data is used to test model performance. This testing model's output is the MLPSentFeat model's performance, and the doctor's answer data is classified. Research [102][103] also combines labeled data from pseudo-labeling results into training data.

3.3.1. Data Preprocessing based on labels

The doctor's answer data, combined with training and test data, underwent a preprocessing process, namely case folding, tokenization, lemmatization, and stopword removal. Several preprocessing processes are arranged based on specific label requirements (F1, F2, F3, F4, F5, and F6), namely the tokenization, lemmatization, and stopword removal processes. It is because the words or characters needed to recognize a label differ from one label to another. These preprocessing settings, especially stop words, are adjusted to the sentence structure and words used in a label by considering explanations from aspects of communication between doctors and patients. Therefore, it is important to understand the form of sentences and the use of words in the doctor's answer data according to the communication aspect. For example, the F2 label has an interrogative sentence form, so it requires a question mark (?), then tokenization, lemmatization, and stopword removal are processed, and a question mark (?) appears for class F2. By default, question marks (?) do not appear in preprocessing. In addition, stopword settings are based on specific label needs. For example, label F2 also requires a question word, so the question word is excluded from the stopword dictionary.

Another example is labels F4 and F6, prohibitive command sentences with negative words. Setting stopwords in labels F4 and F6 brings up negative words by removing them from the stopword dictionary. This also applies to other labels.

This preprocessing setting allows adding information in sentences after the preprocessing process, so the model can classify correctly because it can distinguish one sentence from another. In addition, additional information from the preprocessing setup process can act as additional data that can be utilized by the model in the process of adding sentence features. These two processes (preprocessing settings and sentence features) complement each other; the preprocessing settings process adds additional information that can be captured by adding sentence features, and vice versa. The output from this preprocessing stage is the preprocessing model for each label for the training and test data.

Algorithm 1: Sentence-fetures enrichment process.

```
Input: Sentence after preprocessing
Output: Sentence feature vector V
Procees:
Step 1: Calculate Likelihood Ratios (LR):
    For each word in labels F2 and F5, compute the LR using Equation 1.
Step 2: Calculate Baseline LR:
    For labels F2 and F5, compute the baseline LR using Equation 2.
Step 3: Select Significant Words Based on LR:
        - For each label (F2 and F5):
        - If a word's LR > Baseline LR, retain the word.
        - Otherwise, discard the word.
        - The retained words will proceed to the next step.
Step 4: Compute InfoGain:
        - For each retained word, calculate its InfoGain.
        - Test the MLPSentFeat model with all selected words.
Step 5: Filter Words with Low InfoGain:
    Remove words one by one starting with low InfoGain.
    For each removal:
        - Test the MLPSentFeat model with all selected words.
        -
        - Compare the accuracy of the model with the current features with the accuracy of the model with the overall features.
        - If a decrease in accuracy is found, stop removing features.
        - Make the word a boundary for selecting sentence features.
Step 6: Apply Sentence Features to Classification Model:
    For each sentence in the dataset:
        - Create a feature attribute for each selected word.
        - If a word from the sentence is in the feature set, assign 1 to the corresponding attribute; otherwise, assign 0.
        - Combine all attribute values into a sentence feature vector.
Step 7: Combine Sentence Feature Vector with Embedding Vector:
    - Merge the sentence feature vector with the corresponding embedding vector to form a combined vector.
Step 8: Feed Combined Vector into Classification Model:
    - Use the combined vector as input for the classification process.
```

3.3.2. Sentence vector enrichment process based on the labels

This sentence feature enrichment process adds sentence features to the MLPSentFeat model. Sentence features help the model capture additional information that cannot be captured with just the basic sentence embeddings. There are four processes in this stage: sentence embedding, sentence feature extraction, concatenating sentence embedding vectors and sentence feature vectors, and reducing sentence features. The sentence feature enrichment stage output is a model containing a combined vector of sentence embeddings and sentence features for each label, F1, F2, F3, F4, F5, and F6.

The sentence embedding process uses the SBERT method [104]. The result of this process is a sentence embedding vector. The length of the doctor's answer sentence vector is 384.

The sentence feature extraction process is carried out by adding sentence features to labels F2 and F5, considering that these two labels have a small number of sentences. The purpose of adding this sentence feature is to correctly classify labels with a small number by capturing certain information in the two labels.

The first step in determining sentence features is calculating the LR for each word in labels F2 and F5. The LR calculation formula can be seen in equation 1. Second, calculate the LR baseline at labels F2 and F5. The formula for calculating the baseline LR can be seen in equation 2. Third, choose words with an LR greater than the baseline in each class. A word with an LR greater than the baseline is retained, and vice versa. Words that have an LR greater than the baseline will be processed further. Fourth, calculate the InfoGain for each word. Fifth, remove words that have very small InfoGain.

The InfoGain calculation formula can be seen in equation 3. A small InfoGain indicates that the word does not significantly influence the data, so the word needs to be discarded. Then, the sentence features that have been formed are tested against the classification model. This word deletion stops when the accuracy resulting from the classification model is greater than the previous accuracy for the first time. Sixth, after the sentence features are formed, the sentence features are applied to the classification model. The application of sentence features is done by making sentence features into attributes, then filling in the attributes by counting the words that include sentence features for each sentence. If a word is a sentence feature, it will be given a value of 1 in the sentence feature attribute. Otherwise, it will be given a value of 0. Then, the values of each attribute are combined into a vector. Seventh, sentence feature vectors are combined with embedding vectors. The combination of these two vectors is called a combination vector. This combination vector is processed in the classification model. A complete explanation of adding sentence features is provided in Algorithm 1.

The LR calculation and LR baseline refer to [105][106], which calculates the frequency of word occurrences based on specific criteria, for example, sentence features are taken based on ambiguous or unambiguous sentences. The InfoGain method refers to [107], which performs feature selection using InfoGain and a genetic algorithm. InfoGain is an entropy-based feature selection method used in machine learning to measure how much information a feature provides about a text category. InfoGain helps determine the importance of a term in classification by calculating its contribution to differentiating information categories. The following are the equations for LR, baseline LR, and information gain.

Where, LR of Feature Frequency of Frequency of X in CorpusNon is the frequency of occurrence of words other than the labels being processed. For example, LR calculations are carried out on label F2, and CorpusNon is the frequency of word occurrences from all labels other than F2.

The number of Words in CorpusL is the number of words in a particular label, for example, label F2. The number of words in CorpusNon is the number of words other than processed labels. For example, LR calculations are carried out on label F2, and CorpusNon is the number of words from all labels other than F2.

Where C is a collection of documents with no feature t, the larger the value of G(D,t), the more useful t is for classification in C. This feature t should be selected. If a larger value of G(D,t) is desired, then the values of P(t) and $P(\bar{t})$ must be smaller.

3.3.3. MLP Classification based on Sentence Embedding and Sentence Features Model

The classification experiment used the MLPSentFeat model to test its performance based on the communication aspects of doctors' answer data. The MLP model is a classification model because MLPSentFeat can classify doctors' answer data well and even outperform other models [5].

This classification stage has two processes: the model training process using training data and the data testing process using test data. This model produces classification performance and classifies doctors' answer data.

Here are some initial parameters used in the MLP model. The input layer consisted of one layer with several neurons in the form of a combination vector. The length of this combination vector depended on the length of the sentence feature vector, while the length of the SBERT embedding vector was 384.

There was one hidden layer with a total of 100 neurons. The output layer consisted of one layer with six neurons, representing the number of communication aspect labels, namely F1, F2, F3, F4, F5, and F6. The MLP parameters used included max_iter = 1000, activation = 'relu', solver = 'adam', and random_state = 42. These parameters were set at this stage before the optimization process began.

Optimization was performed on the MLP model to improve classification results, focusing mainly on the number of hidden layers and neurons within them. The reason was that the hidden layer played an important role in the MLP model in capturing more complex and non-linear patterns in the data. Therefore, adjusting the number of neurons and hidden layers to optimal values could improve the model's capacity for better generalization. Meanwhile, the input and output layers were generally more structured and stable, so they were less frequently optimized than the hidden layer, which had a central role in information processing and decision-making.

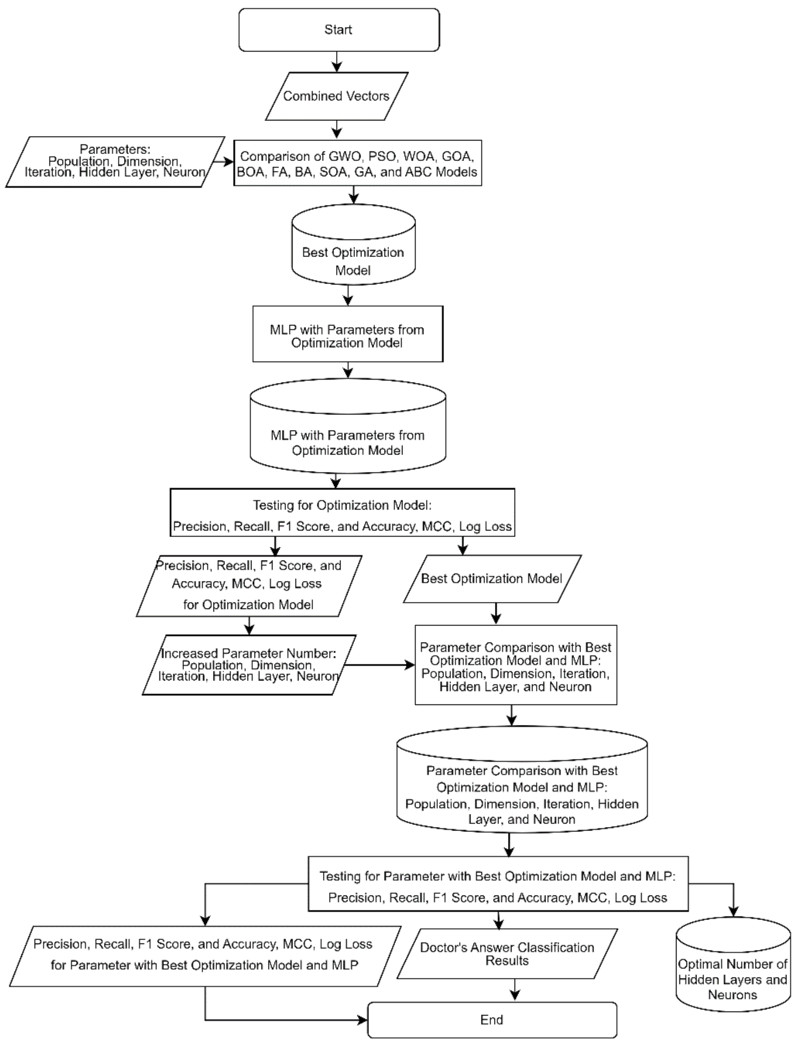

Start
Combined Vectors
Parameters: Population, Dimension, Iteration, Hidden Layer, Neuron
Comparison of GWO, PSO, WOA, GOA, BOA, FA, BA, SOA, GA, and ABC Models
Best Optimization Model
MLP with Parameters from Optimization Model
MLP with Parameters from Optimization Model
Testing for Optimization Model: Precision, Recall, F1 Score, and Accuracy, MCC, Log Loss
Precision, Recall, F1 Score, and Accuracy, MCC, Log Loss for Optimization Model
Best Optimization Model
Increased Parameter Number: Population, Dimension, Iteration, Hidden Layer, Neuron
Parameter Comparison with Best Optimization Model and MLP: Population, Dimension, Iteration, Hidden Layer, and Neuron
Parameter Comparison with Best Optimization Model and MLP: Population, Dimension, Iteration, Hidden Layer, and Neuron
Testing for Parameter with Best Optimization Model and MLP: Precision, Recall, F1 Score, and Accuracy, MCC, Log Loss
Precision, Recall, F1 Score, and Accuracy, MCC, Log Loss for Parameter with Best Optimization Model and MLP
Doctor's Answer Classification Results
Optimal Number of Hidden Layers and Neurons
End

Figure 5: Optimization architecture.

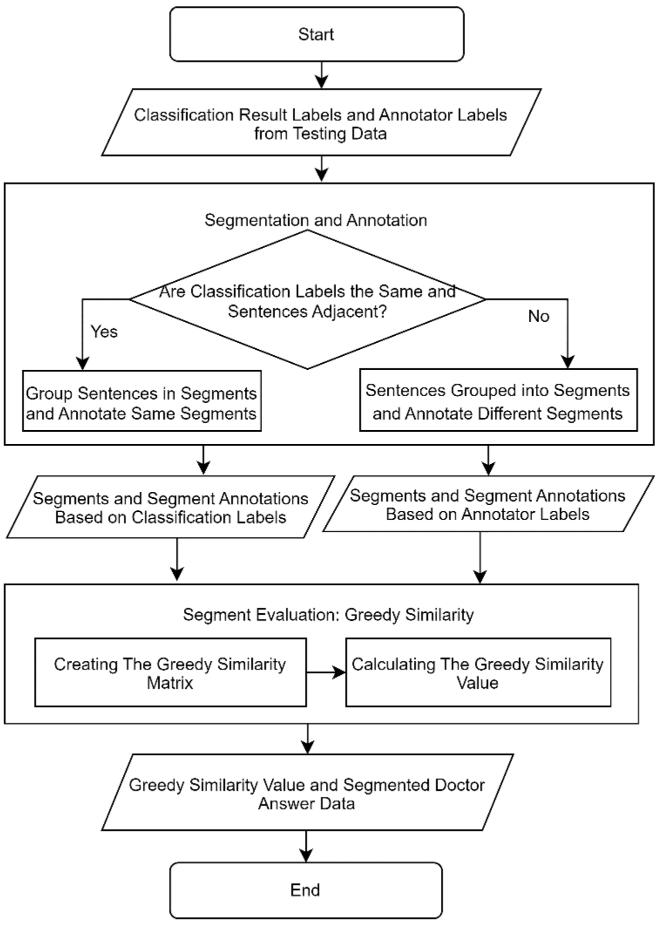

Start
Classification Result Labels and Annotator Labels from Testing Data
Segmentation and Annotation
Are Classification Labels the Same and Sentences Adjacent?
Yes
No
Group Sentences in Segments and Annotate Same Segments
Sentences Grouped into Segments and Annotate Different Segments
Segments and Segment Annotations Based on Classification Labels
Segments and Segment Annotations Based on Annotator Labels
Segment Evaluation: Greedy Similarity
Creating The Greedy Similarity Matrix
Calculating The Greedy Similarity Value
Greedy Similarity Value and Segmented Doctor Answer Data
End

Figure 6: Text segmentation architecture.

Various optimization methods were applied to obtain the optimal number of layers and neurons in the MLP model to complete the classification task on the doctor's answer data based on communication aspects. Additionally, previous studies that served as references showed that by optimizing the number of hidden layers and neurons using optimization methods, the model achieved better performance in recognizing complex patterns [70]. Therefore, optimizing these parameters was considered more appropriate than modifying other parameters.

Testing uses precision, recall, accuracy, and F1 score measuring tools, which refer to [65]. Precision measures how accurate or precise a prediction is compared to the actual results. The precision formula is shown in equation (4). Recall or sensitivity measures how many positive samples are correctly identified by the model. The recall formula is shown in equation (5). Accuracy shows how many predictions are correct compared to the total predictions made. The accuracy formula is shown in equation (6). F1 score is a measure of the balance between precision and recall, especially if the costs of false positives and false negatives are different. The F1 score is shown in equation (7).

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

$$Accuracy = \frac{TP\ TN}{TP +\ +TN + FP + FN} \tag{6}$$

$$\text{F1-Score} = 2 \text{ x } \frac{(\text{Recall x Precision})}{(\text{Recall+ Precision})} \tag{7}$$

Where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

3.4. Optimization process

This optimization model is for tuning parameters, namely optimizing the MLPSentFeat method to make it more optimal, by getting the optimal number of hidden layers and neurons. This optimization model optimizes the number of hidden layers and neurons using a metaheuristic optimization algorithm. Based on this optimization goal, the optimization stage was carried out after the initial classification experiment using MLP with the pre-established initial hidden layer parameters and neuron count to achieve more optimal results. Therefore, the number of hidden layers and neurons was parameter optimized to obtain optimal results with the optimal parameters. Several optimization algorithm parameters are set to obtain the optimal number of hidden layers and neurons, namely population number [108], dimensions [109], iterations [110], hidden layers, and neurons

[70][22]. The metaheuristic optimization models compared are GWO, PSO, CSA, WOA, GOA, BOA, BCO, HS, FA, HHO, CS, BA, SOA, GA, and ABC. Optimization model testing uses recall, accuracy, F1-Score, MCC (Matthew correlation coefficient), and log loss. MCC is suitable for measuring imbalanced and binary [111] and multiclass [112]. The MCC formula is shown in equation (8) [111].

$$MCC = \frac{(TP * TN) - (FP * FN)}{\sqrt{(TP + FP) * (TP + FN) * (TN + FP) * (TN + FN)}} \quad (8)$$

The architecture of the optimization model, which has two processes, is shown in Figure 5. First, metaheuristic optimization models are compared to find the best model with constant parameters. Second, parameter comparison by increasing the number of parameters in the best metaheuristic model. Optimization model input, testing data, and optimization model parameters, namely, the number of populations, dimensions, iterations, hidden layers, and neurons. The output of the optimization model is the optimal number of hidden layers and neurons. The model will then be applied to other data to test its performance on different data.

3.5. Text segmentation process based on six aspects of communication

The text segmentation stage aims to form segments and annotate segments of doctors' answers from the classification results. The formation of this segment is based on aspects of doctor-patient communication. The input for the text segmentation model is testing data of 20% of labeled data, which contains labels from the classification results and labels from the annotator. The flow of the text segmentation model is shown in Figure 6.

The formation of this segment is based on labels generated from the classification process and labels provided by the annotator. This step evaluates segments by comparing segments from classification labels and annotator labels. The segment formation and segment annotation process combine sentences with the same class next to each other. A situation like this will form a segment. If these conditions are not met, a new segment will be formed. For example, if several sentences have the same class but are not located next to each other, then these sentences are not included in the same segment. Apart from that, if two sentences are located next to each other but do not have the same class, these sentences are not included in the same segment. Algorithm 2 explains the segment formation process, segment annotation, greedy similarity matrix creation, and greedy similarity value calculation. This step was taken by Manchanda [113], who showed that sentences with the same topic are not always in one segment in the data.

Algorithm 2: Segment formation and evaluation.

```
Inputs:
D ← Set of sentences
C_cls ← Class labels obtained from the classification
C_annot ← Class labels from the annotator
Output:
S_cls ← Segments from classification
S_annot ← Segments from the annotator
G ← Greedy similarity matrix
GS ← Greedy similarity value
Step 1: Segment Formation
1. Initialize:
   For classification, set segment_id_cls = 0
   For annotation, set segment_id_annot = 0
   Create empty lists S_cls and S_annot
2. Create Segments (Same Process for Both Classification and Annotator):
   For each sentence i in D do:
     For Classification Segments:
       If i == 0, create a new segment S_cls(segment_id_cls)
       else
       If Ccls(i) == Ccls(i-1)
         Append sentence i to S_cls(segment_id_cls)
       else
         Increment segment_id_cls
         Create new segment
         S_cls(segment_id_cls) and add sentence-i
   For Annotator Segments:
     If i == 0, create a new segment
       S_annot(segment_id_annot)
     else
       If Ccls(i) == Ccls(i-1)
         Append sentence-i to
         S_annot(segment_id_annot)
     else
       Increment segment_id_annot
       Create new segment
       S_annot(segment_id_annot) and add sentence-i
Step 2: Assign Segment Annotations
1. For each segment Scls(i) in S_cls do:
   Assign segment_id_cls(i)
   Assign classification label Lb(i)
   Assign doctor-patient communication aspect
2. For each segment Sannot(i) in S_annot do:
   Assign segment_id_annot(i)
   Assign label annotator La(i)
   Assign doctor-patient communication aspect
Step 3: Segment Evaluation (Greedy Similarity Calculation)
1. Initialize Greedy Similarity Matrix:
   Create an empty matrix G with dimensions |Scls|×|Sannot|
   Set all elements G(i, j) = 0
2. Greedy Similarity Matrix:
   For each segment i in S_cls do:
     For each segment j in S_annot do:
       If Lb(i) ≠ La(j), then:
         G(i, j) = 0
       else:
         The number of identical sentences is calculated using equation 9.
Step 4: Compute Final Greedy Similarity Score
1. The highest values were selected from matrix G, ensuring that no two selected values share the same row or column.
2. Compute the final greedy similarity score using equation 10.
```

One data can consist of one or several segments, and a segment can consist of one or several sentences. This annotation is important because conveying an explanation in the doctor's answer includes certain aspects of communication. Segment formation and segment annotation are done on labels from text classification results, with labels from the annotator.

Segment evaluation aims to measure the similarity of the segments resulting from text classification with the annotator segments. This segment is evaluated using a greedy similarity model by comparing sentences in the classification result segment with sentences in the annotator segment. The greedy similarity model refers to [114] [3], which measures the similarity of two categories.

Table 4: Results of training and testing data, pseudo-labeling results.

Data Types	Total Data	Total Sentences	F1	F2	F3	F4	F5	F6
Training1 DS1	4328	85374	12947	799	44754	7431	4359	15084
Training2 DS1	4328	84907	12903	782	44622	7442	4363	14795
Training3 DS1	4328	84379	12887	781	44198	7376	4357	14780
Training4 DS1	4328	84786	12915	753	44688	7351	4351	14728
Training5 DS1	4328	85478	12936	793	45034	7480	4354	14881
Testing1 DS1	1082	20857	3200	178	11070	1839	1087	3483
Testing2 DS1	1082	21324	3244	195	11202	1828	1083	3772
Testing3 DS1	1082	21852	3260	196	11626	1894	1089	3787
Testing4 DS1	1082	21445	3232	224	11136	1919	1095	3839
Testing5 DS1	1082	20753	3211	184	10790	1790	1092	3686
Pseudo-Labeled DS1	10031	200724	30363	11	139285	9575	6324	15166
Training1 DS2	6403	88314	13251	236	38533	6562	6401	23331
Training2 DS2	6403	88470	13253	210	38609	6544	6398	23456
Training3 DS2	6400	88564	13228	210	38468	6648	6397	23613
Training4 DS2	6403	88569	13226	226	38591	6629	6406	23491
Training5 DS2	6403	88331	13262	214	38367	6601	6398	23489
Testing1 DS2	1600	22248	3304	38	9609	1684	1599	6014
Testing2 DS2	1600	22092	3302	64	9533	1702	1602	5889
Testing3 DS2	1600	21998	3327	64	9674	1598	1603	5732
Testing4 DS2	1600	21993	3329	48	9551	1617	1594	5854
Testing5 DS2	1600	22231	3293	60	9775	1645	1602	5856
Pseudo-Labeled DS2	28247	384916	48834	74	208943	26824	27066	73175
Training1 DS3	799	18918	1592	52	11887	1443	592	3352
Training2 DS3	799	18412	1586	65	11327	1458	595	3381
Training3 DS3	800	18683	1581	58	11693	1474	591	3286
Training4 DS3	799	18637	1584	56	11511	1476	619	3391
Training5 DS3	799	18418	1601	61	11402	1413	611	3330
Testing1 DS3	200	4349	394	21	2568	373	160	833
Testing2 DS3	200	4855	400	8	3128	358	157	804
Testing3 DS3	199	4584	405	15	2762	342	161	899
Testing4 DS3	200	4630	402	17	2944	340	133	794
Testing5 DS3	200	4849	385	12	3053	403	141	855
Pseudo-Labeled DS3	2111	29783	2560	185	21338	2429	826	2445

The way to fill in the matrix elements is to compare sentences in segments between rows (containing sentences in the classification result segment) and columns (containing sentences in the annotator segment). While comparing these sentences, pay attention to the labels and information of the row sentences and the labels and information of the column sentences. Conditions for filling matrix elements are as follows: 1) If the row segment annotation and column segment annotation are not the same, then the matrix element is filled with a value of zero (0). 2) If the row and column segment annotations are the same, then pay attention to the sentences in the row and column segments. Count the total of sentences in the row and column simultaneously, which is called the correct sentences. A sentence in the classification result segment is said to be correct if it is located in the same segment as the annotator segment. Calculation of the greedy similarity matrix using the formula is shown in equation (9).

$$G(i,j) = \frac{2\ x\ C}{|i| + |j|} \tag{9}$$

Where G is the greedy similarity matrix, C is the number of correct sentences between rows and columns, namely the number of sentences of a segment in a row of the matrix, and j is the number of sentences of a segment in a column of the matrix.

After all elements are filled in, the greedy similarity value can be calculated by 1 selecting the most significant values from the greedy similarity matrix. The most considerable value cannot be located in one row with the same column as another row. 2) Calculate the greedy similarity value using the formula shown in equation (10).

$$GS = \frac{2\ x\ \sum Gmax}{\sum|i| + \sum|j|} \tag{10}$$

Where GS is the greedy similarity value, Gmax is the most considerable, i is the number of sentences of a segment in the matrix row, and j is the number of sentences of a segment in the matrix column.

4 Result and analysis

This section explains the results and analysis of research on text segmentation in online health consultation using MLPSentFeat. These results were obtained based on the method that has been designed. The research results are the pseudo-labeling stage, classification stage, text segmentation stage, and optimization stage.

4.1. Pseudo-labeling

The pseudo-labeling stage aims to label new data using several machine learning models. The pseudo-labeling stage uses two inputs: unlabeled data and labeled data (see Table 3). The training process uses training data to train pseudo-labeling models, which include LR, SVC, DT, KNN, RF, NB, GBM, Xboost, LightGBM, and CatBoost. Each model runs two preprocessing processes: word embedding (IndoBERT) and sentence embedding (SBERT). So, for each data type (DS1, DS2, and DS3), this training model produces 20 labels from each pseudo-labeling model, based on the embedding model. Then, a classifier voting process was carried out on the 20 labels. This classifier voting process takes the highest number of labels to be used as the final label.

The output of the training model is shown in Table 4, which displays data divided using K-fold and rotated five times. This data division is carried out on training data, as much as 80% of the labeled data, and test data, as much as 20% of the labeled data. This division applies to every DS1, DS2, and DS3 dataset. The training and testing dataset amounts are divided almost equally in each dataset.

However, it has a different number of sentences in each fold, which also causes the distribution of sentences in each label to be different. The amount of data from pseudo-labeling results is the same as that of unlabeled data (see Table 3). The resulting pseudo-labeling data is used as training data in the testing model. The results of this pseudo-labeling stage are then used in the classification stage as training data combined with the previously existing training data.

Table 5: Examples of stopwords list for each aspect.

Label	List of Stopword
F1	-
F2	*Mengapa* (why), *berapa* (how many), *apakah* (how), *kapan* (when)
F3	*Berapa* (how many), *beberapa* (several), *berikut* (following)
F4	*Jangan* (do not), *dalam* (in), *umum* (general), *tanpa* (without)
F5	*Tidak* (do not), *jangan* (do not), *tanpa* (without), *ada* (there is)
F6	*Perlu* (need), *tidak* (do not), *jangan* (do not), *perlu* (need), *cukup* (enough)

Table 6: Results of InfoGain for F2 and F5.

Words on F2	InfoGain	Words on F5	InfoGain
Konsultasi (consultation)	0.0406	*Moga* (hopefully)	0.1765
Spesialis (specialist)	0.0279	*Manfaat* (benefits)	0.0755
Air (water)	0.0254	*Bantu* (help)	0.0382
?	0.0242	*Tidak* (no)	0.0324
Darah (blood)	0.0238	*Jangan* (don't)	0.0219
Keluh (complaints)	0.0208	*Ya* (yes)	0.0156
….	….	….	….
Begah (bloated)	0.0001	*Bingung* (confused)	0.0002
Inhaler (inhaler)	0.0001	*Masif* (massive)	0.0002
Bisul (boils)	0.0001	*Signifikan* (significant)	0.0002
Kakak (brother)	0.0001	*Bermafaat* (benefit)	0.0001

Table 7: Results of LR and baseline LR for F2 and F5.

Words on F2	LR	Words on F5	LR
Apakah (what)	1.0000	*Moga* (hopefully)	1.0000
Bagaimana (how)	1.0000	*Informasi* (information)	0.2149
Fresh (fresh)	0.5000	*Bantu* (help)	0.6023
Essence (essence)	0.5000	*Lekas* (soon)	0.7347
Sigmoid (sigmoid)	0.5000	*Sembuh* (get well)	0.1356
?	0.9818	*Paham* (understand)	0.4211
….	….	….	….
Paru(lungs)	0.0105	*Selesai* (finished)	0.02222
Suplemen (supplements)	0.0105	*Tangis* (crying)	0.02174
Sakit (pain)	0.0105	*Ayah* (father)	0.02128
Demam (fever)	0.0104	*Iring* (accompany)	0.02128
Baseline LR	**0.0103**	**Baseline LR**	**0.0205**

Table 8: InfoGain reduction results for F2, F5 label, and testing on MLPSentFeat model.

Ranking	Features F2	InfoGain	Accuracy	Ranking	Features F5	InfoGain	Accuracy
-	Full features	-	0.9906	-	Full features	-	0.9911
12	*Alami* (natural)	0.0103	0.9910	6	*Ya* (yes)	0.0156	0.9912
11	*Anak* (children)	0.0104	0.9909	5	*Jangan* (don't)	0.0219	0.9914
10	***Nyeri* (painful)**	**0.0106**	**0.9895**	**4**	***Tidak* (no)**	**0.0324**	**0.9910**
9	*Apakah* (is)	0.0187	0.9903	3	*Bantu* (help)	0.0382	0.9909
8	*Menstruasi* (menstruation)	0.0194	0.9891	2	*Manfaat* (benefits)	0.0755	0.9910
7	*Sakit* (pain)	0.0195	0.9908	1	*Moga* (hopefully)	0.1765	0.9906
6	*Keluh* (complaints)	0.0208	0.9903	-	-	-	-
5	*Darah* (blood)	0.0238	0.9896	-	-	-	-
4	?	0.0242	0.9881	-	-	-	-
3	*Air* (water)	0.0254	0.9906	-	-	-	-
2	*Spesialis* (specialist)	0.0279	0.9902	-	-	-	-
1	*Konsultasi* (consultation)	0.0406	0.9904	-	-	-	-

Table 9: Results of InfoGain calculations on Features1.

Feature	InfoGain	Feature	InfoGain
Dokter (doctor)	0.1773	Obat (medicine)	0.0180
Hindar (avoid)	0.0493	Gejala (symptoms)	0.0136
Makan (eat)	0.0415	Ganggu (disturb)	0.0130
Tidak (not)	0.0324	Nyeri (pain)	0.0106
Minum (drink)	0.0255	Buang (throw)	0.0079
?	0.0243	Perlu (need)	0.0049
Konsumsi (consume)	0.0190	Khawatir (worry)	0.0038
Apakah (what)	0.0187	-	-

Table 10: InfoGain reduction results on Features1 and testing on MLPSentFeat model.

Rank	Features	InfoGain	Accuracy
-	Full features	-	0.9898
15	*Khawatir* (worry)	0.0038	0.9902
14	*Perlu* (need)	0.0049	0.9906
13	***Buang* (throw)**	**0.0079**	**0.9888**
12	*Nyeri* (pain)	0.0106	0.9904
11	*Ganggu* (disturb)	0.0130	0.9893
10	*Gejala* (symptoms)	0.0136	0.9900
9	*Obat* (medicine)	0.0180	0.9885
8	*Apakah* (what)	0.0187	0.9895
7	*Konsumsi* (consume)	0.0190	0.9901
6	?	0.0243	0.9899
5	*Minum* (drink)	0.0255	0.9910
4	*Tidak* (not)	0.0324	0.9908
3	*Makan* (eat)	0.0415	0.9891
2	*Hindar* (avoid)	0.0493	0.9885
1	*Dokter* (doctor)	0.1773	0.9899

4.2. Classification process based on six aspects of communication

This classification aims to classify doctors' answer data based on aspects of doctor-patient communication using the MLPSentFeat model. The classification stage uses input as training data, as much as 80% of labeled data, testing data, as much as 20% of labeled data, and pseudo-labeled data. Combined training data is used for model training. Test data is used for model testing. This classification stage has a preprocessing model for each label, F1, F2, F3, F4, F5, and F6. This preprocessing model for each label assists the MLPSentFeat model in recognizing sentences in each label. This preprocessing model is obtained by arranging the displayed punctuation marks or the words removed from the stopword dictionary into a specific label. The punctuation used is a question mark (?). Examples of the use of stopwords for each label are shown in Table 5. It can be seen that the F1 label does not have a specific stopword, because the MLPSentFeat model can recognize the F1 label. This F1 label uses common stopwords in the stopword removal process. There is a stop word to identify several labels because this stop word is often used on both labels. For example, the word '*tidak'* (do not) refers to labels F5 and F6, and the word '*tanpa'* (without) refers to F4 and F5. Stop words can be determined by looking at words that often appear in a label or based on the type of sentence used in the label. For example, the labels F4, F5, and F6 use the command sentence prohibition, so the word prohibition is excluded from the stopword dictionary. Another example is the F3 label, which uses many of the word '*berikut'* (following), so this word was removed from the stopword dictionary.

4.2.1. Establishment of sentence features

The formation of sentence features was carried out on labels F2 and F5 because these two labels have a small number of sentences. This data condition applies to all DS1, DS2, and DS3 data. Formation of sentence features is done by calculating the LR for each word on the two labels, calculating the baseline LR on each label, selecting words that have a greater LR than the label, calculating the InfoGain for each selected word, and deleting words that have a low InfoGain (see Figure 4 and Algorithm 1). The results of the LR calculation for each word in labels F2 and F5, as well as the baseline LR in both labels, are shown in Table 6. Apart from that, Table 6 also displays words that exceed the baseline LR in labels F2 and F5. The LR value ranges between 0 and 1. A word with an LR value of 1 on a particular label will not appear on another label, and vice versa. For example, the words *'Apakah'* (what) and '*Bagaimana'* (how) have an LR of 1, so these words do not appear in other labels. The baseline LR for label F2 is 0.0103, while the baseline LR for label F5 is 0.0205. Label F2 has 451 unique words, and words with an LR exceeding the baseline LR of 298 words. Label F5 has 223 unique words, and words with an LR exceeding the baseline LR of 88 words. Words that have LR more than the baseline LR will be processed further. Then, these selected words are made into temporary sentence features (called full features) in the MLPSentFeat model to see their performance. After that, record the test results.

Next, InfoGain calculations are done on the selected words in labels F2 and F5. The InfoGain value has a range between 0 and 1. Examples of InfoGain calculation results for each word in labels F2 and F5 are shown in Table 7. Table 7 displays words with InfoGain sorted from largest to smallest value. The word with the most considerable InfoGain value is the most informative word on the label, and conversely, the lower the InfoGain value, the less informative the word is. For example, the word '*konsultasi'* (consultation) has the highest InfoGain of 0.0406 on label F2, indicating that this word is informative on label F2. The word '*moga'* (hopefully) has the highest InfoGain of 0.1765 on the F5 label, indicating that this word is informative in the F5 label.

InfoGain provides an initial picture of the relationship between sentence features and classification targets. Features with high InfoGain have a statistically significant relationship with the target, while features with low InfoGain provide less information. This analysis is important to understand the relevance of each feature in the context of the classification target. Further classification trials are needed to ascertain the contribution of each feature to the overall model performance.

Next, words with a very low InfoGain are deleted, and those with relatively high InfoGain are retained. There are special provisions for determining the limit for deleting these words. A word with a low InfoGain means that the word does not have much influence on the label, so it needs to be deleted. So, one of the tips to stop deleting this word is to see how influential the word is on a label. The words that are not deleted are used as temporary sentence features. Then, this temporary sentence feature is applied in the MLPSentFeat model to see its performance. After that, record the test results. Testing the MLPSentFeat model with features by removing features one by one. Compare the temporary sentence features with the accuracy of full sentence features and observe and find a lower accuracy value than full sentence features for the first time. Temporary sentence features with lower accuracy than full features are the limit for selecting fixed sentence features. So, the selection of fixed sentence features starts from words with the highest InfoGain in a particular label to words with lower accuracy than full sentence features. Please remember that the process of deleting and testing words in the MLPSentFeat model is carried out in stages (one by one) and starts with the word with the lowest InfoGain. Gradual deletion of words may result in a different performance than removing several words immediately, even if the deletion stops with the same word.

For example, deleting words with the lowest InfoGain and the results of testing the MLPSentFeat model with temporary features to obtain fixed sentence features are shown in Table 8. Table 8 shows the temporary sentence features sorted from smallest to largest InfoGain. Then, the accuracy will be observed until the accuracy drops for the first time compared to the model using full features. This process is carried out on labels F2 and F5. Deleting sentence features in label F2, starting from the word with the lowest InfoGain until after the word 'alami' (natural). This decision was taken because these words have a very low value, namely below 0.01 (keep in mind that there are no special provisions in this determination). So there are 12 words retained in the F2 label. This process is also applied to the F5 label. There are six words retained in the F5 label.

The accuracy of testing the MLPSentFeat model with full sentence features resulted in an accuracy of 0.9898 for label F2 and 0.9911 for label F5. After removing temporary sentence features is carried out in stages, testing with the MLPSentFeat model is carried out on labels F2 and F5, observing the accuracy with full sentence features until the first decrease in accuracy is found, marked in yellow. Label F2 found the first decrease in the word '*nyeri'* (painful), while label F5 found the first decrease in the word '*tidak'* (no). These two words are used as the limit for determining fixed sentence features. So, the fixed sentence features in the F2 label start from the fixed sentence features boundary to the feature with the highest InfoGain, starting with the word '*nyeri'* (painful), which is the feature selection boundary, to '*konsultasi'* (consultation), which has the highest InfoGain. The F2 label has 10 fixed sentence features.

Meanwhile, the fixed sentence features in the F5 label start with the word 'tidak' (no), which is the limit for feature selection, to 'moga' (hopefully), which has the highest InfoGain. The F5 label has four fixed sentence features. The sentence features formed consist of a collection of fixed sentence features labeled F2 and F5, totaling 14 features called Features2.

Features 2 is taken from the words in the labels F2 and F5. Features 2 contains '*nyeri'* (painful), '*apakah'* (is), '*menstruasi'* (menstruation), '*sakit'* (pain), '*keluh'* (complaints), '*darah'* (blood), ?, '*air'* (water), '*spesialis'* (specialist), '*konsultasi'* (consultation), '*tidak'* (no), '*bantu'* (help), '*manfaat'* (benefits), and '*moga'* (hopefully). Apart from Features2, there are also Features1 and Features3. Features1 contains 15 features: '*dokter'* (doctor), '*hindar'* (avoid), '*makan'* (eat), '*tidak'* (no), '*minum'* (drink), ?, '*konsumsi'* (consume), '*apakah'* (what), '*obat'* (medicine), '*gejala'* (symptoms), '*ganggu'* (disturb), '*nyeri'* (pain), '*buang'* (throw), '*perlu'* (need), and '*khawatir'* (worry). Features 1 takes words often appearing in the labels F2, F4, F5, and F6 [5]. The reason for taking sentence features in these four labels and not focusing on labels F2 and F5 is that the MLPSentFeat model is not optimal in classifying these four labels. Features3 is the result of reducing the features of Features1 by two features to 12 features. Features3 contains '*dokter'* (doctor), '*hindar'* (avoid), '*makan'* (eat), '*tidak'* (no), '*minum'* (drink), ?, '*konsumsi'* (consume), '*apakah'* (what), '*obat'* (medicine), '*gejala'* (symptoms), '*ganggu'* (disturb), and '*nyeri'* (pain).

Reducing Features1 to Features3 aims to obtain optimal sentence features.The process of reducing sentence features is the same as the previous process. The InfoGain calculation results for each word from Features1 are displayed in Table 9. Table 9 displays the InfoGain sorted from largest to smallest value. The '*dokter'* (dokter) feature has the highest information gain value of 0.1773, which indicates that this feature provides the most significant information contribution to the classification target.

Next, the MLPSentFeat model was tested by deleting sentence features individually, starting from the lowest InfoGain. Then the accuracy results for each feature are compared with the accuracy of the full features. If it is found that the accuracy of the feature has decreased for the first time, then stop. This feature is used as a limit for selecting full sentence features. The results of testing the MLPSentFeat model by deleting sentence features one by one are shown in Table 10. Full sentence features from Features3 are taken from this limit to the highest InfoGain, starting from the word through, which is the feature selection limit, to the word doctor, which has the highest InfoGain. So, subtracting sentence features from Features1 produces 13 features called Features3.

4.2.2. Experiment MLPSentFeat model

Testing of the MLPSentFeat model is carried out in several types, namely testing three data sets, DS1, DS2, and DS3, with three sentence features, Features1, Features2, and Features3. This test shows the performance of the three features against these three datasets. In addition, the MLPSentFeat model is compared with the MLP and HTCN models. Comparison of the MLPSentFeat model with the MLP because the basis of this research model uses the MLP model to see the performance of the two models in classifying doctors' answer data with certain aspects. A comparison of the MLPSentFeat model with HTCN [39] was carried out because this HTCN model also aims to classify doctors' answer data with certain aspects. So, both of the models have the same goal. The results of various MLPSentFeat tests are shown in Table 11. The MLP model parameters used are hidden_layer_sizes=100, max_iter=1000, activation='relu', solver='adam', random_state=42, warm_start=True, and learning_rate=0.001. The HTCN model parameters used are channels=[64, 64], kernel_size=2, dropout=0.2, learning_rate=0.001, epochs=10, and batch_size=1.

Researcher Juanita [87] also identified doctors' answer data from the Alodokter platform (DS1) using the Multi-Label Classification (MLC) model. This model was not tested in the study because it is for multi-label classification, while this study uses multiclass classification. The F1 score results of the MLC model with various combinations of methods are around 0.97. while the average F1 score results of the MLPSentFeat model for DS1 with various sentence features are between 0.98 and 0.99. The MLPSentFeat model can outperform the MLC model by 1 or 2 points.

Table 11 presents the performance of several models across different datasets. The comparative analysis focuses on the baseline model (MLP) and the proposed model (MLPSentFeat), particularly related to labels F2 and F5. These two labels are the focus of the study due to

Table 11: Experiment MLPSentFeat model.

Models	F1-Score							Accuracy
	F1	F2	F3	F4	F5	F6	AVG	
MLP+DS1	0.9919	0.8660	0.9826	0.9806	0.9931	0.9673	0.9636	0.9808
MLP+DS2	0.9946	0.8750	0.9785	0.9835	0.9991	0.9712	0.9670	0.9806
MLP+DS3	0.9819	0.9272	0.9834	0.9840	0.9656	0.9580	0.9667	0.9780
HTCN+DS1	0.9938	0.9657	0.9605	0.9020	0.9913	0.9189	0.9554	0.9552
HTCN+DS2	0.9936	0.8678	0.9438	0.9308	0.9978	0.9180	0.9420	0.9474
HTCN+DS3	0.9864	0.9605	0.9859	0.9813	0.9427	0.9631	0.9700	0.9800
MLPSentFeat+DS1+Features1	0.9954	0.9933	0.9875	0.9773	0.9967	0.9760	0.9877	0.9864
MLPSentFeat+DS1+Features2	0.9957	0.9964	0.9882	0.9848	0.9970	0.9741	0.9894	0.9872
MLPSentFeat+DS1+Features3	0.9952	0.9969	0.9928	0.9916	0.9920	0.9857	0.9924	0.9918
MLPSentFeat+DS2+Features1	0.9946	0.9833	0.9906	0.9918	0.9989	0.9890	0.9914	0.9914
MLPSentFeat+DS2+Features2	**0.9946**	**0.9871**	**0.9911**	**0.9928**	**0.9992**	**0.9899**	**0.9925**	**0.9920**
MLPSentFeat+DS2+Features3	0.9946	0.9757	0.9904	0.9885	0.9992	0.9898	0.9897	0.9913
MLPSentFeat+DS3+Features1	0.9903	0.9861	0.9905	0.9898	0.9717	0.9757	0.9840	0.9872
MLPSentFeat+DS3+Features2	0.9908	0.9861	0.9909	0.9890	0.9604	0.9782	0.9826	0.9875
MLPSentFeat+DS3+Features3	0.9903	0.9861	0.9911	0.9877	0.9672	0.9788	0.9835	0.9878

the relatively limited number of sentences representing these aspects. At the same time, the MLPSentFeat model was designed to classify such sentences despite the scarcity of data in those categories. The results show that the MLP model tends to produce lower F1-scores for label F2 compared to the MLPSentFeat model in several combinations of datasets and features. The MLPSentFeat model is more effective in classifying label F2, demonstrating better and more consistent outcomes. Conversely, for label F5, the MLP model performs better in some datasets, with very stable F1-scores. MLPSentFeat still shows competitive performance for this label, although it does not always outperform the MLP model.

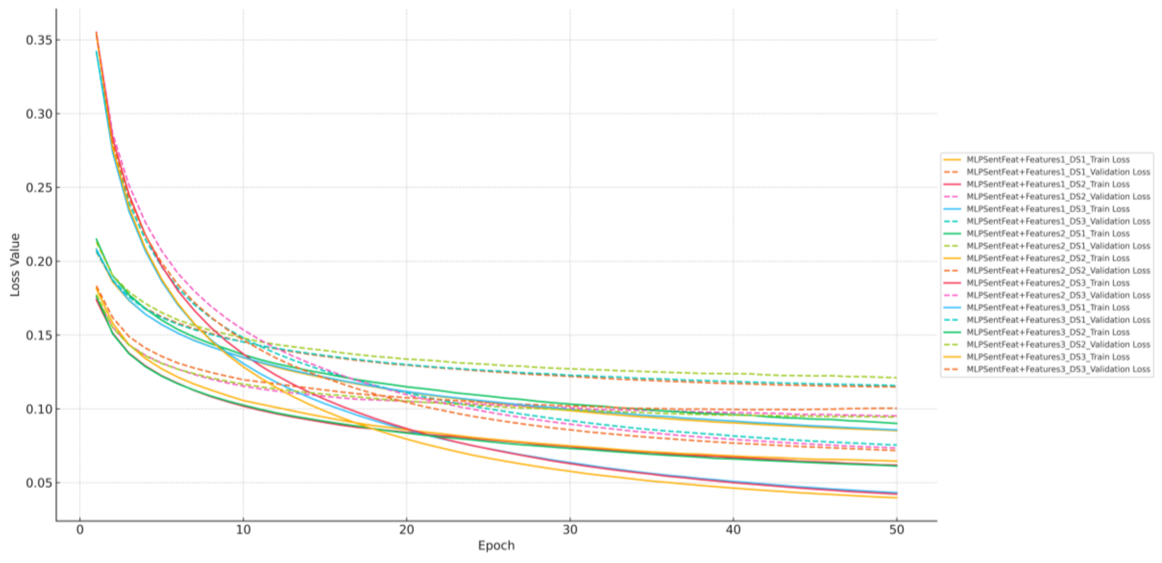
As previously described, the sentence features used in the model were extracted based on the lexical characteristics of labels F2 and F5. However, these features may also influence classification performance on other labels, as certain words appear in F2 and F5 and other communication-related categories. This overlap may result in side effects on cross-label classification performance. Overall, the MLPSentFeat model performs better in handling labels F2 and F5. More specifically, the combination of dataset DS2 with various sentence features in the MLPSentFeat model shows fluctuating F1-scores for label F2, compared to other data-feature combinations that tend to be more stable. For instance, the MLPSentFeat+DS2+Features3 combination yields the lowest F1-score for label F2 at 0.9757—lower than DS3, despite DS3 having significantly less data than DS2.

This finding suggests that the observed fluctuation is due to the quality of interrogative sentence construction in label F2 of dataset DS2. Many questions in this label are written in informal formats, often starting with interjections or introductory expressions commonly used in spoken communication. Such constructions can confuse the model, as the sentence structure resembles a statement rather than a question. In contrast, the F2 sentences in DS1 and DS3 are more structurally regular and conform to standard interrogative conventions, resulting in more stable model performance on these datasets.

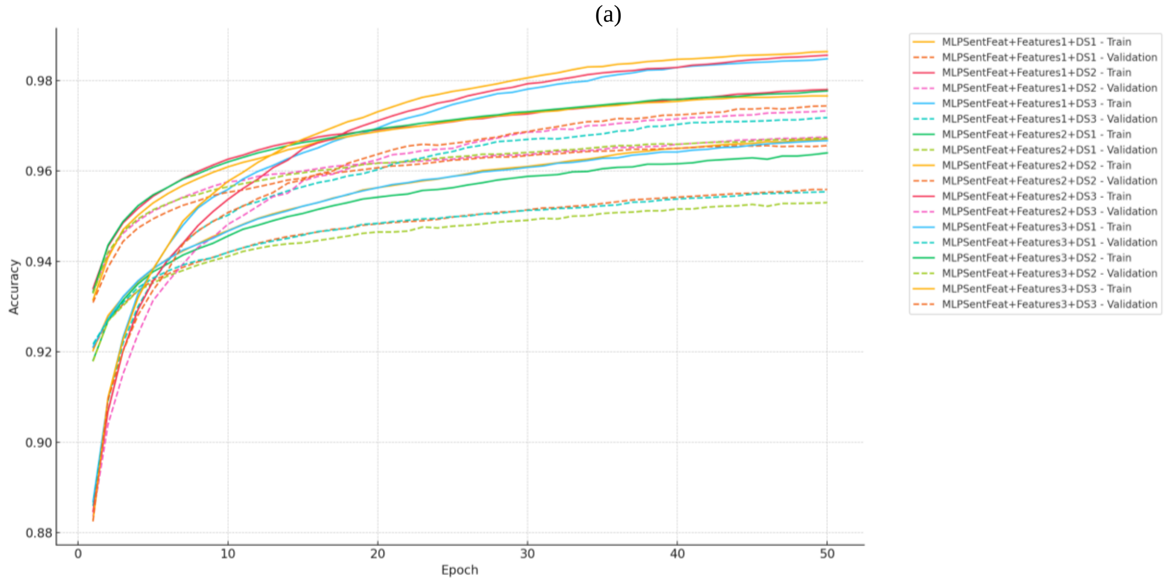
The MLPSentFeat model also shows stable performance on label F5, comprising sentences expressing hope or wishes. These sentences follow consistent lexical patterns that the model more easily recognizes. Furthermore, expressions of hope often carry positive or neutral sentiment, aligning well with the sentiment-based features employed in the model. On the other hand, interrogative sentences under label F2 are more difficult to identify due to their flexible and varied structure, which does not always begin with an explicit question word and may, in some cases, resemble declarative statements. Structural variation, the use of interjections at the beginning of sentences, and their emotionally neutral nature all present challenges for the model in consistently distinguishing questions.

The strength of the MLPSentFeat model in handling labels F2 and F5 demonstrates that the integration of explicitly designed sentence features within the model architecture significantly contributes to its sensitivity to linguistic structure variation. It indicates that a sentence feature–based approach can enhance the model's semantic representation capabilities, particularly for sentences that are uncommon or written in non-standard formats. Practically, the model's success in identifying labels F2 and F5 is highly significant, as these aspects play a crucial role in establishing effective two-way communication between doctors and patients—in the form of exploratory questions and empathetic expressions. Therefore, enhancing the model's ability to recognize these two aspects may serve as a foundation for developing quality evaluation systems for online medical consultations.

Although the MLPSentFeat model has demonstrated promising performance, it remains a challenge, especially in identifying label F2, encompassing interrogative sentences with highly variable structures that do not always adhere to formal writing conventions. This challenge opens avenues for further research, such as developing syntactic augmentation techniques, implementing sentence structure–based representations, and improving specialized text normalization procedures that can significantly affect model performance. At present, specialized text normalization has proven to make a meaningful contribution, particularly for label F2, as indicated by the stable F1-score in DS1 compared to DS2. Meanwhile, DS3, which did not undergo such



(a)



(b)

Figure 7: a) Training and validation loss and b) training and validation accuracy of the MLPSentFeat model on various datasets and sentence features.

normalization, still shows stable performance due to the inherently well-structured interrogative sentences in its F2 category, which adhere to standard linguistic conventions.

An analysis of Table 11 from the perspective of overall model comparison reveals that the MLPSentFeat model consistently demonstrates the best performance across all datasets and feature variations. The highest achievement is observed in the combination of MLPSentFeat+DS2+Features2, which yields an average F1-score of 0.9925—the highest score recorded in the entire set of experiments. This finding indicates that incorporating relevant sentence features can significantly enhance the model's ability to recognize key patterns involving doctors' responses based on communication aspects in the classification task. Furthermore, including specific feature variations, particularly Features2 and Features3, contributes substantially to improving model performance, as evidenced by consistently high F1-scores across various datasets. This inclusion suggests that the integration of appropriate sentence-level features enables the model to better adapt to differences in data characteristics. In contrast, the HTCN method demonstrates the lowest and least stable performance, with the lowest average F1-score of 0.9420 recorded in the HTCN+DS2 combination. It underscores that HTCN is less capable of adapting to the characteristics of the datasets used in this study, resulting in overall lower performance compared to the other models—especially when compared to MLPSentFeat, which excels in leveraging additional sentence-level features.

The analysis of Table 11 in terms of dataset size and text normalization indicates that the performance of text classification models is not solely determined by the volume of data, but is also significantly influenced by the text normalization processes applied to the data. The three datasets used—DS1, DS2, and DS3—exhibit different characteristics in terms of both data volume and the cleanliness and consistency of textual content. DS1, which contains a moderate amount of data and has undergone a specific text normalization procedure, demonstrates highly stable performance across all labels. The combination of the MLPSentFeat model with Features3 on DS1 yields an average F1-score of 0.9924, with F1-scores for all labels (F1–F6) consistently above 0.99. Notably, label F2—known to be sensitive to semantic variation—achieves the highest F1-score across all configurations, reaching 0.9969. This finding suggests that proper text normalization can enhance the consistency and quality of sentence feature representations, regardless of dataset size. In contrast, DS2 represents the largest dataset in terms of volume but does not undergo specific normalization. Its best-performing combination (MLPSentFeat+Features2) achieves the highest average F1-score of 0.9925. However, label-level stability is less consistent; the F1-score for label F2 in this combination is only 0.9871, which is lower than the corresponding score in DS1. It implies that while a large dataset can improve overall performance, without adequate handling of noise and linguistic variation, the model may remain vulnerable to decreased performance on certain labels. Meanwhile, DS3, the smallest dataset in data size, presents relatively clean text and does not require special normalization. Despite having a lower average F1-score than DS1 and DS2 (maximum 0.9835), the model maintains stable F1-scores across labels. The F1-score for label F2 in the MLPSentFeat+Features3 combination reaches 0.9861, indicating that well-structured text can contribute to stable model performance even with limited data. Overall, these experimental findings highlight that text normalization and structural data quality play a crucial role in improving the accuracy and stability of text classification models, and can compensate for limitations in data quantity. A larger dataset does not necessarily lead to superior performance if not accompanied by adequate text normalization, particularly in classification tasks that require sensitivity to semantic structure and variation in expressive sentence forms.

The analysis of Table 11 from the perspective of sentence feature utilization shows that three types of features—Features1, Features2, and Features3—were tested to evaluate their contributions to the performance of the MLPSentFeat-based classification model. The experimental results indicate that no single sentence feature consistently outperformed the others in all conditions, as the effectiveness of each feature is highly influenced by the characteristics of the data and the accompanying text processing applied. Features1, which was constructed based on the frequency of words predominantly appearing in labels F2 and F5, demonstrated relatively stable performance but did not dominate. This feature generally yielded good F1-scores across all three datasets but was never part of the combination that achieved the highest average F1-score. These findings suggest that although Features1 may contribute to sentence representation in general, its capacity to capture semantic variation and expressive structures remains limited. In contrast, Features2—generated through a LR approach and feature selection using Information Gain—achieved the highest average F1-score of 0.9925 when applied to the DS2 dataset. However, in this combination, the F1-score for label F2 decreased to 0.9871, indicating that while this feature effectively captures global semantic context, maintaining performance stability for expressive or minor labels remains challenging. This limitation is compounded by the nature of DS2, which contains a large volume of data but lacks text normalization, making the feature more susceptible to the effects of noise and imbalanced label distribution. On the other hand, Features3, which was derived by reducing Features1 using Information Gain, demonstrated the most stable performance across labels, particularly when combined with datasets DS1 and DS3. The combination of MLPSentFeat with DS1 and Features3 resulted in F1-scores exceeding 0.99 across all labels, including label F2, which achieved the highest score among all configurations at 0.9969. This feature was designed to capture sentence structures related to pragmatic functions, such as interrogative expressions, expressions of hope, and other communicative forms. The performance stability achieved through Features3 was especially evident in datasets that had undergone specific normalization (DS1) or those that naturally contained well-structured sentences (DS3), suggesting that the effectiveness of this feature strongly depends on the cleanliness and consistency of the data to reach optimal performance.

Figure 7 presents the training and validation loss curves (Figure 7a) and the training and validation accuracy curves (Figure 7b) for the MLPSentFeat model across various combinations of datasets and sentence features. These visualizations provide insights into the model's learning behavior throughout 50 training epochs and are used to identify convergence patterns, generalization capability, and potential overfitting. Each color in the graphs represents a unique combination of sentence feature type (Features1, Features2, or Features3) and dataset (DS1, DS2, or DS3), while line styles distinguish between training data (solid lines) and validation data (dashed lines). In general, most combinations exhibit a decreasing trend in loss and an increasing trend in accuracy as the number of epochs increases, indicating an effective learning process. However, the degree of alignment between the training and validation curves is a key indicator in assessing whether the model is overfitting.

The combinations that exhibited symptoms of overfitting were MLPSentFeat+DS1+Features2, MLPSentFeat+DS2+Features2, and MLPSentFeat+DS2 +Features3. These three configurations shared a common pattern: the training loss decreased sharply, while the validation loss remained stagnant or increased after several epochs. Additionally, a significant gap between training and validation accuracy was observed, indicating that the model had overly adapted to the training data but failed to generalize effectively to the validation set. In the case of MLPSentFeat+DS2+Features2, overfitting occurred despite the large dataset size and the highest average F1 score among all combinations. It was primarily due to the absence of specialized text normalization in DS2, which caused the model to learn from inconsistent

Table 12: Results of various classification experiments DataID #3422.

Order	**Sentences**	**1***	**2****	**3*****	**4******	**5*******	**6********
0	Halo (Hello)	F1	F1	F1	F1	F1	F1
1	Terima kasih telah menggunakan layanan konsultasi Alodokter (Thank you for using Alodokter consultation services)	F1	F1	F1	F1	F1	F1
….	….	….	….	….	….	….	….
13	Apakah terdapat keluhan lain yang menyertai? (Are there any other accompanying complaints?)	F2	F3	F2	F2	F2	F2
14	Apakah mual? (Is it nausea?)	F2	F3	F2	F2	F2	F2
15	Apakah muntah? (Is it vomiting?)	F2	F3	F2	F2	F2	F2
16	Apakah perut kembung? (Is your stomach bloated?)	F2	F3	F2	F2	F2	F2
17	Apakah diare? (What is diarrhea?)	F2	F3	F2	F2	F2	F2
18	Apakah konstipasi? (What is constipation?)	F2	F3	F2	F2	F2	F2
19	Apakah BAB (Buang Air Besar) berdarah? (Is your bowel movement bloody?)	F2	F3	F2	F2	F2	F2
….	….	….	….	….	….	….	….
24	Beri nutrisi yang bergizi dan sehat (Give it nutritious and healthy nutrition)	F6	F6	F3	F6	F6	F6
25	Konsumsi air putih (Drink water)	F6	F6	F6	F6	F6	F6
….	….	….	….	….	….	….	….
29	Semoga membantu (I hope this helps)	F5	F5	F5	F5	F5	F5

1: Annotator labels. 2: MLP labels. 3: MLPSentFeat+Features1 labels. 4: MLPSentFeat+Features2 labels. 5: MLPSentFeat+Features3 labels. 6: MLP Optimization Labels.

Table 13: Hyperparameter optimization with various optimization algorithms and MLP method.

Optimization	**F1 score**							**Neuron**	**Log Loss**	**MCC**	**Accuracy**
	F1	**F2**	**F3**	**F4**	**F5**	**F6**	**AVG**				
MLP+PSO	1.00	1.00	0.99	0.97	0.99	0.98	0.99	91	0.06	0.98	0.99
MLP+BOA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	100	0.07	0.98	0.99
MLP+GWO	**1.00**	**1.00**	**0.99**	**0.97**	**0.99**	**0.98**	**0.99**	**91**	**0.06**	**0.99**	**0.99**
MLP+SOA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	91	0.06	0.98	0.99
MLP+BA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	91	0.06	0.98	0.99
MLP+FA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	77	0.07	0.98	0.99
MLP+GA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	68	0.07	0.98	0.99
MLP+GOA	1.00	1.00	0.99	0.97	0.99	0.98	0.99	100	0.07	0.98	0.99
MLP+WOA	0.99	1.00	0.99	0.97	0.99	0.98	0.99	100	0.08	0.98	0.99
MLP+ABC	1.00	1.00	0.99	0.97	0.99	0.95	0.98	98	0.09	0.97	0.99

words and sentence structures. Features2, constructed using LR and feature selection based on InfoGain, tended to capture dominant word frequency patterns. In noisy data, this characteristic could lead to learning bias. This combination demonstrated the most pronounced gap between the training and validation curves, making it the most severe case of overfitting observed in the study. Meanwhile, the combination of MLPSentFeat+DS1+Features2 experiences overfitting because Features2 is more suitable for data with high variation and a large amount of data, as seen in the DS2 dataset. Given the features' highly selective nature, the model could not handle expressive variations present in the validation set. A similar situation was observed in MLPSentFeat+DS2+Features3, where the pragmatic sentence structures intended to be captured by the features did not align well with the varied and unnormalized nature of DS2. As a result, the model struggled to develop a generalized representation.

Conversely, the remaining six combinations did not exhibit signs of overfitting. The MLPSentFeat+DS1+Features3 configuration was the most stable, showing highly aligned training and validation loss and accuracy curves. This configuration indicates strong generalization ability, supported by the quality of the DS1 dataset, which underwent text normalization, and the high relevance of Features3 to the pragmatic structure of sentences. Other combinations, such as MLPSentFeat+DS3+Features3 and MLPSentFeat+DS3+Features2, also demonstrated stability despite being derived from a smaller dataset. It suggests that the cleanliness and consistency of sentence structures in DS3 contributed significantly to the model's stable performance. Combinations involving Features1 across all datasets also showed reasonably stable performance without overfitting, although they did not achieve top scores. It can be attributed to the general and less selective nature of Features1, which prevented the model from overfitting to specific patterns in the training

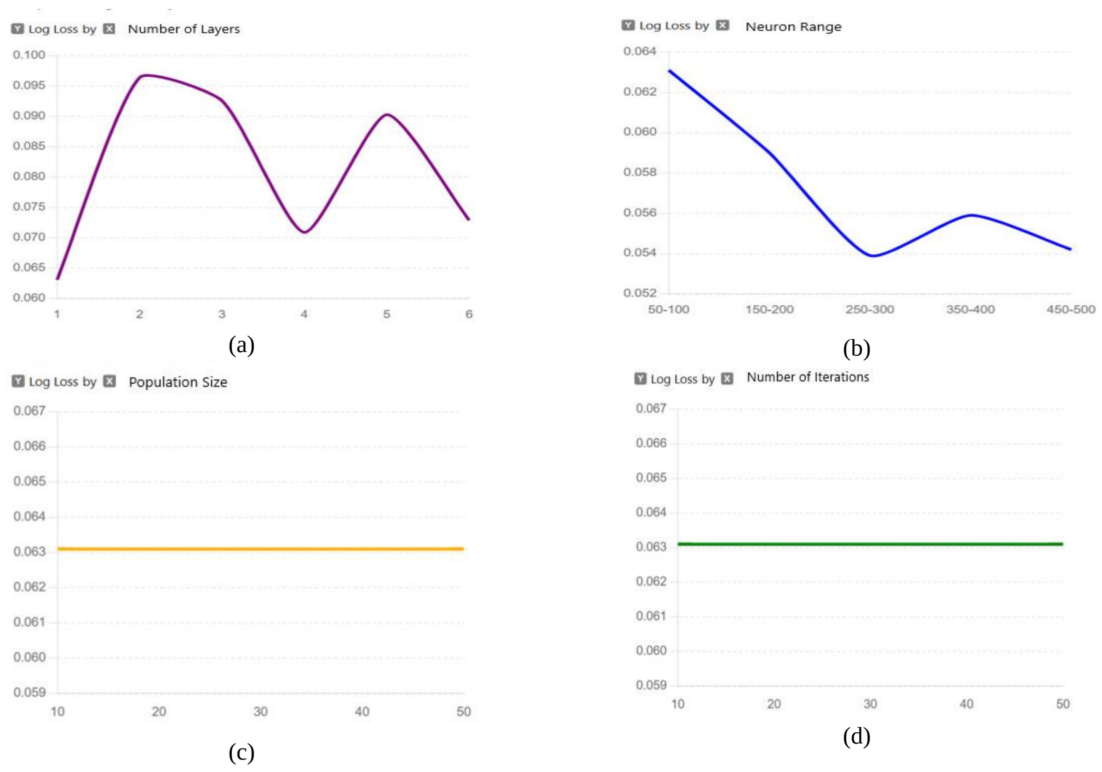


Figure 8: Log loss graph of GWO and MLP with various parameters. (a) Log loss parameter number of hidden layers. (b) Log loss parameter neuron range. (c) Log loss parameter number of populations. (d) Log loss parameter number of iterations.

Table 14: GWO and MLP parameter optimization.

Hyperparameter Setting	**Parameters**	**F1 score**							**Neurons**	**Log Loss**	**MCC**
		F1	**F2**	**F3**	**F4**	**F5**	**F6**	**AVG**			
Population=10, HL*=4, Iter**=10, Neurons= 50-100, dim***=4	Hidden Layer	1.00	1.00	0.99	0.98	1.00	0.98	0.99	76, 93, 69, 60	0.07	0.98
Population=10, HL=1, Iter=10, Neurons= 250-300, dim=1	**Neurons**	**0.99**	**1.00**	**0.99**	**0.99**	**0.99**	**0.99**	**0.99**	**283**	**0.05**	**0.99**
Population=50, HL=1, Iter=10, Neurons= 50-100, dim= 1	Population	1.00	1.00	0.99	0.97	0.99	0.98	0.99	91	0.06	0.98
Population=10, HL=1, Iter=50, Neurons= 50-100, dim= 1	Iteration	1.00	1.00	0.99	0.97	0.99	0.98	0.99	91	0.06	0.98

*HL=Hidden Layer. **Iter=Iteration. ***dim=Dimension.

data. These findings reinforce the notion that the presence or absence of overfitting is highly dependent on the interaction between dataset characteristics, the method of sentence feature extraction, and model capacity. Text normalization processes, dataset size and cleanliness, and the alignment between features and sentence semantics are critical factors in maintaining model stability and avoiding overfitting.

An example of the results of classifying doctors' answers based on communication aspects is shown in Table 12. The example shown uses data ID_Data 3422 from DS1. The models shown in this example are only the MLP and MLPSentFeat models because they are used to display the classification results of the MLP and MLPSentFeat models, considering that the performance of the MLPSentFeat model is increased from the MLP model. In addition, the MLPSentFeat model can classify F2 labels better than the MLP model (based on the accuracy in Table 11). These two classification results must be displayed because the MLPSentFeat model is a development of the MLP model.

The reference label used is the annotator label, which is given directly by the doctor. The colored sections in the table highlight errors in the classification results. The green color indicates an error in the classification results by the MLP model, so the MLP model has difficulty identifying sentences with label F2 in sentences 13-19. However, the MLPSentFeat model (with various sentence features and model optimization) can classify these sentences into label F2. The similarity of the annotator labels with the MLPSentFeat model labels and model optimization proves this. Meanwhile, the blue color (sentence 24) shows an error in the classification results by the MLPSentFeat model with Features1 (column 3), but the classification results are correct in other types of features. The results of this classification cannot ensure that the sentence features that cause errors in classification are bad. The results of this classification will be processed into the text segmentation and model optimization stages.

4.3. Optimization process

This optimization stage has two processes, namely hyperparameter optimization and application of the parameters that have been obtained (explained in Figure 6). Hyperparameter optimization to find the best configuration through trials using several optimization algorithms. The hyperparameters that have been formed are applied to various data (DS1, DS2, and DS3) and sentence features (Features1, Features2, and Features3). The classification results using optimal parameters were also analyzed in terms of the performance of the MLPSentFeat model, data overfitting, and the segments formed.

4.3.1. Hyperparameter optimization

The best configuration desired from this parameter optimization process is to obtain the optimal number of hidden layers and neurons for the MLP method. The process carried out in the hyperparameter optimization process first compares several metaheuristic optimization algorithms, including PSO, BOA, GWO, SOA, BA, FA, GA, GOA, WOA, and ABC. This optimization algorithm is run simultaneously with the MLP method, because this optimization algorithm functions to optimize the MLP method. This comparison of optimization algorithms is to obtain the best algorithm for optimizing the MLP method applied to doctors' answer data. The best optimization algorithm is determined by looking at the F1 score or the highest accuracy that exceeds the performance of the MLPSentFeat model. If several optimization algorithms have the same F1 score or accuracy value, look at the log loss value and choose the smallest value. If several optimization algorithms have the same log loss value, consider the MCC value and choose the highest value. Alternatively, directly choose the lowest log loss value. This provision was taken because this research used accuracy and F1 score metrics.

However, during optimization experiments, it turned out that these two metrics were not enough to determine the best algorithm. Second, the value of the initial parameters used in the best optimization algorithm should be increased. Then, the performance between parameter values is compared. The parameters set in the optimization algorithm to obtain the optimal number of hidden layers and neurons are the number of populations, dimensions, iterations, hidden layers, and neurons. The initial parameter settings are population=10, hidden layer=1, iteration=10, neurons=50-100, dimension=1. Each parameter value is increased 5 times. Each parameter value and type is compared, and the smallest log loss value is selected. A model with the smallest log loss value also has a high F1 score or accuracy. The parameter with the smallest log loss value is the optimal parameter, which is the parameter value used by that parameter.

The results of comparative experiments between optimization algorithms are shown in Table 13. The optimization model parameters include population=10, hidden layer=1, iteration=10, neurons=50-100, and dimension=1. MLP parameter settings include max_iter=1000, activation='relu', solver='adam', and random_state=42. Other MLP parameters have been determined: one input layer with 399 neurons, one hidden layer with 100 neurons, and one output layer with six neurons. Due to limited available resources, this hyperparameter optimization experiment was implemented on DS1 data, where training and test data had not been separated using K-fold (explained in Table 4).

The results of the optimization and MLP experiments in Figure 8 show that almost all optimization models produce an average F1 score and accuracy that is the same as the F1 score percentage of the MLPSentFeat model of 0.99, or even reduced. If only these two metrics were used, the best optimization model could not be chosen. Therefore, model selection requires metrics like log loss and MCC values. The first analysis looks at the smallest log loss value, namely 0.06, which is owned by three models, namely GWO, SOA, and BA. Because there are three of the best models, the analysis continues by paying attention to the MCC metrics of these three models. The best optimization model was chosen based on the highest MCC value, namely the GWO optimization model with 91 neurons 91.

After finding the best optimization algorithm, namely GWO, the algorithm was tested again by increasing the parameter values of each type of parameter. This parameter value was increased 5 times for each parameter. The experimental results of the performance of each parameter of GWO are shown in Figure 8. This figure shows the experimental results of the GWO algorithm, which displays log loss based on changes in the parameters of the number of hidden layers, neuron range, population size, and iteration size. An explanation of the initial GWO parameters has been provided previously. After that, each parameter value is increased gradually. The parameter for the number of hidden layers has the values 1, 2, 3, 4, 5, 6 (a). Setting the number of dimensions parameters is done together with the hidden layer parameters with values 1, 2, 3, 4, 5, and 6. The dimension parameters function to optimize the number of neurons in each layer [115]. This research uses dimension parameters to optimize the number of neurons in the hidden layer. The neuron range parameters are 50-100, 150-200, 250-300, 350-400, 450-500 (b). The population number parameter has the values 10, 20, 30, 40, 50 (c). The number of iterations parameter values 10, 20, 30, 40, 50 (d).

Figure 8(a) shows the relationship between the number of hidden layers and the log loss value. Log loss increases until it reaches a maximum value of 0.097 when two hidden layers exist. A significant decrease occurred when the number of hidden layers increased to four, with a minimum log loss value of 0.067. An increase occurred again when the hidden layers increased to five before experiencing a slight decrease when the hidden layers

Table 15: Experiment on applying MLP parameter optimization to the MLPSentFeat model.

Model	F1	F2	F3	F4	F5	F6	AVG	Accuracy
MLPSentFeat+DS1+Features1	**0.9954**	**0.9979**	**0.9923**	**0.9869**	**0.9966**	**0.9862**	**0.9926**	**0.9915**
MLPSentFeat+DS1+Features2	0.9953	0.9979	0.9895	0.9868	0.9901	0.9795	0.9899	0.9885
MLPSentFeat+DS1+Features3	0.9948	0.9985	0.9898	0.9899	0.9946	0.9779	0.9909	0.9888
MLPSentFeat+DS2+Features1	0.9946	0.9852	0.9918	0.9936	0.9992	0.9908	0.9925	0.9926
MLPSentFeat+DS2+Features2	0.9946	0.9776	0.9898	0.9901	0.9992	0.9884	0.9900	0.9908
MLPSentFeat+DS2+Features3	0.9946	0.9680	0.9921	0.9933	0.9992	0.9915	0.9898	0.9929
MLPSentFeat+DS3+Features1	0.9908	0.9861	0.9909	0.9863	0.9715	0.9771	0.9838	0.9874
MLPSentFeat+DS3+Features2	0.9903	0.9861	0.9919	0.9890	0.9748	0.9802	0.9854	0.9889
MLPSentFeat+DS3+Features3	0.9908	0.9861	0.9906	0.9884	0.9687	0.9762	0.9835	0.9871

numbered six. This pattern shows that a larger number of hidden layers does not continually improve model performance, but can cause overfitting or increased noise in learning. The best experimental results with parameter settings for the number of hidden layers are shown in Table 14.

Figure 8(b) illustrates the effect of the number of neurons on the log loss value. Log loss experienced a decreasing trend as the number of neurons increased until it reached the range of 250–300 neurons, with the lowest log loss value being 0.054. An increase in the number of neurons after passing this number causes slight fluctuations in the log loss. Too few neurons increase log loss, while too many neurons do not provide significant improvement. The 250–300 neurons range produces the lowest log loss and balances model complexity and good generalization. The best experimental results with parameter settings for the number of neurons are shown in Table 14.

Figure 8(c) visualizes the relationship between the optimization algorithm's population size and the log loss value. Varying population size, ranging from 10 to 50, did not significantly change log loss, which remained stable at around 0.063. Population size is not the main factor influencing model performance. Increasing the population size does not necessarily improve the optimization quality, as the model has reached stable performance with smaller population sizes. The best experimental results with population number parameter settings are shown in Table 14.

Figure 8(d) shows the relationship between the number of iterations and log loss. Increasing the number of iterations does not significantly impact log loss, remaining constant at around 0.063. The model has reached a convergence point before increasing the number of iterations. Additional iterations only increase computing time without providing significant performance improvements. The best experimental results with parameter settings for the number of iterations are shown in Table 14.

The results in Table 14 show that the best parameter to optimize is the number of neurons, with a configuration of 283 neurons producing the lowest log loss of 0.05. This value is smaller than the hidden layer, population, and other iteration parameters, which have a higher log loss. This configuration also produces an average F1 score of 0.99, showing that the model has excellent prediction performance and a minimal error rate.

4.3.2. Parameter application to the MLPSentFeat model

The hyperparameters used are the number of hidden layers of 1 and neurons of 283. These parameters are applied to the MLPSentFeat model with various datasets (DS1, DS2, and DS3) and sentence features (Features1, Features2, and Features3). The results of applying parameter optimization to the MLPSentFeat model with various datasets and sentence features are shown in Table 15.

The results of the experiment presented in Table 15 demonstrate that the application of parameter optimization to the MLP model positively impacts the MLPSentFeat model's performance in classifying doctor responses based on communication aspects. It is evident that the model optimization successfully increased the number of models with an average F1-score greater than 0.99, which doubled to four models achieving an average F1-score above 0.99, namely MLPSentFeat+DS1+Features1 (0.9926), MLPSentFeat+DS1+Features2 (0.9899), MLPSentFeat+ DS1+Features3 (0.9909), and MLPSentFeat+DS2+ Features1 (0.9925). This improvement indicates that the optimization enhances the model's accuracy and improves its performance's consistency and stability, as reflected in the stable F1-scores across most model combinations. However, despite performance improvements in some models, not all models showed an increase in average F1-score or accuracy after optimization. Sometimes, although optimization aims to improve overall performance, the process can lead to a decrease in performance for certain labels that are more varied or complex in structure. For example, in label F2 with the DS2 dataset, the F1-score decreased after optimization when combining Features2 and Features3. This decline can be explained by the model's inability to handle the diversity of sentence structures in label F2, which contains many interrogative sentences—this sentence type is more difficult for the model to recognize. While optimization improves overall performance, performance decreases for labels with higher expressive variation, such as label F2. It impacts the average F1-score of MLPSentFeat+DS2+Features2, the best model before optimization, with an average F1-score of 0.9925. However, after optimization, this model experienced a decrease in the F1-score for label F2 by 0.0095, decreasing the overall average F1-score of this model to 0.9900.

The performance of sentence features in the dataset after optimization remains consistent with the state before optimization, where Features2 is still suitable for data with

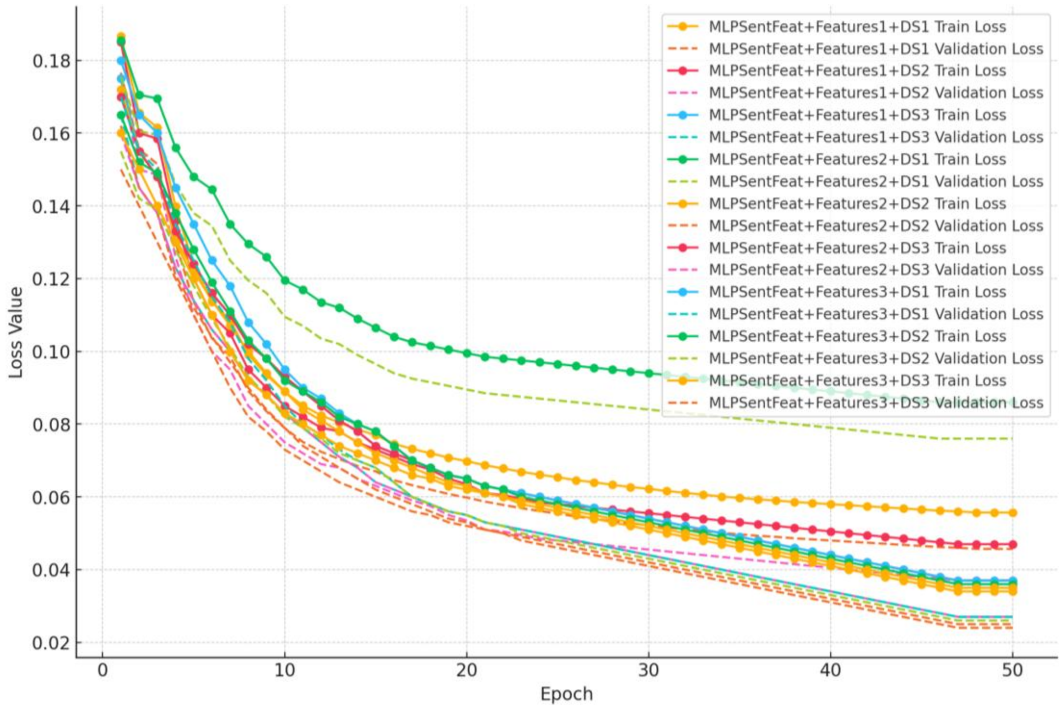
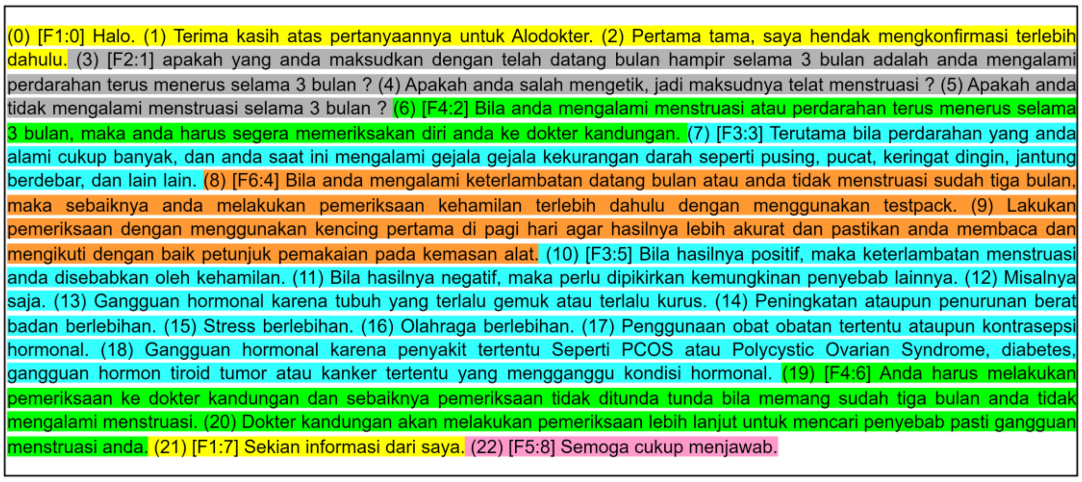


Figure 9: Training and validation loss from applying MLP parameter optimization to the MLPSentFeat model.

high variability (such as DS2), Features3 is more suitable for datasets that are well-structured and clean (such as DS1 and DS3), and Features1 shows better stability across the entire dataset. An important finding in this experiment is the improvement in Features1's performance after optimization. Features1, which initially did not provide the best performance, managed to become the best model in this experiment, with an average F1-score of 0.9926 in the combination of MLPSentFeat+DS1+Features1, which demonstrates excellent stability, as the F1-score across all labels tends to be more consistent compared to other features. Meanwhile, Features2 and Features3, which performed better before optimization, remained stable despite a slight decrease in F1-score after optimization.

Figure 9 illustrates the training and validation loss curves of the MLPSentFeat model after optimization, applied to various combinations of datasets and sentence features. Four models exhibited overfitting: MLPSentFeat+DS1+Features2, MLPSentFeat+DS2+Features2, MLPSentFeat+DS2+Features3, and MLPSentFeat+DS3+Features2. These models showed a sharp decline in training loss, while the validation loss stagnated or slightly increased post-optimization. The Features2 feature set experienced overfitting across multiple datasets due to its high selectivity in choosing the most informative features. As a result, the model became overly reliant on specific patterns (e.g., dominant words) and struggled to handle more diverse sentence structures. Although Features2 is well-suited for datasets with high frequency, such as DS2, it lacks the flexibility to manage more structured datasets, such as DS1 and DS3, leading to overfitting. In contrast, MLPSentFeat+DS2+Features3 continued to experience overfitting after optimization because the optimization did not fully address the structural issues in the data. Features3 could not adequately handle the highly variable data in DS2, which contained numerous sentences with diverse structures. Despite optimization improving certain aspects of the model (e.g., parameter tuning), if the data is not sufficiently normalized or exhibits high variability that the model cannot manage, optimization alone is insufficient to mitigate overfitting. The model tends to memorize patterns only relevant to the training data, failing to generalize to the more diverse validation data. The Features1 feature set, on the other hand, remained stable across all datasets and labels and did not experience overfitting, both before and after optimization. Furthermore, after optimization, the combination of Features1 achieved the highest average F1-score, particularly in the MLPSentFeat+DS1+Features1 model.

Models that did not experience overfitting after optimization—namely, MLPSentFeat+DS1+Features1, MLPSentFeat+DS2+Features1, MLPSentFeat+DS3+Features1, MLPSentFeat+DS1+Features3, and MLPSentFeat+DS3+Features3—demonstrated strong generalization capabilities. This can be attributed to the use of Features1 and Features3, both of which are more flexible in handling a variety of data types and sentence variations. Features1 tends to select more general features and is not overly focused on dominant words, enabling the model to adapt to variations in sentence structure, whether in structured data such as DS1 or more variable data like DS2. As such, Features1 helps the model remain stable in both training and validation datasets without over-relying on patterns that are specific to the training data. Features3, which focuses on sentence pragmatics, allows the model to capture more complex sentence contexts, such as expressions of sympathy or interrogative sentences. Although Features3 is more flexible in handling sentences with pragmatic variation, it still experiences overfitting on



(0) [F1:0] Halo. (1) Terima kasih atas pertanyaannya untuk Alodokter. (2) Pertama tama, saya hendak mengkonfirmasi terlebih dahulu. (3) [F2:1] apakah yang anda maksudkan dengan telah datang bulan hampir selama 3 bulan adalah anda mengalami perdarahan terus menerus selama 3 bulan ? (4) Apakah anda salah mengetik, jadi maksudnya telat menstruasi ? (5) Apakah anda tidak mengalami menstruasi selama 3 bulan ? (6) [F4:2] Bila anda mengalami menstruasi atau perdarahan terus menerus selama 3 bulan, maka anda harus segera memeriksakan diri anda ke dokter kandungan. (7) [F3:3] Terutama bila perdarahan yang anda alami cukup banyak, dan anda saat ini mengalami gejala gejala kekurangan darah seperti pusing, pucat, keringat dingin, jantung berdebar, dan lain lain. (8) [F6:4] Bila anda mengalami keterlambatan datang bulan atau anda tidak menstruasi sudah tiga bulan, maka sebaiknya anda melakukan pemeriksaan kehamilan terlebih dahulu dengan menggunakan testpack. (9) Lakukan pemeriksaan dengan menggunakan kencing pertama di pagi hari agar hasilnya lebih akurat dan pastikan anda membaca dan mengikuti dengan baik petunjuk pemakaian pada kemasan alat. (10) [F3:5] Bila hasilnya positif, maka keterlambatan menstruasi anda disebabkan oleh kehamilan. (11) Bila hasilnya negatif, maka perlu dipikirkan kemungkinan penyebab lainnya. (12) Misalnya saja. (13) Gangguan hormonal karena tubuh yang terlalu gemuk atau terlalu kurus. (14) Peningkatan ataupun penurunan berat badan berlebihan. (15) Stress berlebihan. (16) Olahraga berlebihan. (17) Penggunaan obat obatan tertentu ataupun kontrasepsi hormonal. (18) Gangguan hormonal karena penyakit tertentu Seperti PCOS atau Polycystic Ovarian Syndrome, diabetes, gangguan hormon tiroid tumor atau kanker tertentu yang mengganggu kondisi hormonal. (19) [F4:6] Anda harus melakukan pemeriksaan ke dokter kandungan dan sebaiknya pemeriksaan tidak ditunda tunda bila memang sudah tiga bulan anda tidak mengalami menstruasi. (20) Dokter kandungan akan melakukan pemeriksaan lebih lanjut untuk mencari penyebab pasti gangguan menstruasi anda. (21) [F1:7] Sekian informasi dari saya. (22) [F5:8] Semoga cukup menjawab.

Figure 10: Text segmentation results of DataID #222.

Table 16: Greedy data similarity matrix DataID #222.

Classification Segment	Annotator Segment								
	[F1:0]	[F2:1]	[F4:2]	[F3:3]	[F6:4]	[F3:5]	[F4:6]	[F1:7]	[F5:8]
[F1:0] (0,1,2)	■	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
[F2:1] (3,4,5)	0.00	■	0.00	0.00	0.00	0.00	0.00	0.00	0.00
[F4:2] (6)	0.00	0.00	■	0.00	0.00	0.00	0.00	0.00	0.00
[F3:3] (7)	0.00	0.00	0.00	■	0.00	0.00	0.00	0.00	0.00
[F6:4] (8,9)	0.00	0.00	0.00	0.00	■	0.00	0.00	0.00	0.00
[F3:5] (10,11, 12, 13,14,15,16, 17,18)	0.00	0.00	0.00	0.00	0.00	■	0.00	0.00	0.00
[F4:6] (19,20)	0.00	0.00	0.00	0.00	0.00	0.00	■	0.00	0.00
[F1:7] (21)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	■	0.00
[F5:8] (22)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	■

DS2, which contains a high level of variation in interrogative sentences. This is because Features3 struggles to handle the high variability in sentence structures in DS2, which are inconsistently structured. While optimization enhances certain aspects of the model, Features3 remains challenged in dealing with complex, unstructured interrogative sentences in DS2, leading to overfitting. On the other hand, in DS1 and DS3, which have more consistent sentence structures, Features3 is able to effectively handle pragmatic sentence structures, preventing overfitting and enabling better generalization on validation data.

Models that did not experience overfitting after optimization—namely, MLPSentFeat+DS1+Features1, MLPSentFeat+DS2+Features1, MLPSentFeat+DS3+ Features1, MLPSentFeat+DS1+Features3, and MLPSentFeat+DS3+Features3—demonstrated strong generalization capabilities. It can be attributed to Features1 and Features3, which are more flexible in handling various data types and sentence variations. Features1 tends to select more general features and is not overly focused on dominant words, enabling the model to adapt to variations in sentence structure, whether in structured data such as DS1 or more variable data like DS2. As such, Features1 helps the model remain stable in training and validation datasets without over-relying on patterns specific to the training data. Features3, which focuses on sentence pragmatics, allows the model to capture more complex sentence contexts, such as expressions of sympathy or interrogative sentences. Although Features3 is more flexible in handling sentences with pragmatic variation, it still experiences overfitting on DS2, which contains a high level of variation in interrogative sentences. It is because Features3 struggles to handle the high variability in sentence structures in DS2, which are inconsistently structured. While optimization enhances certain aspects of the model, Features3 remains challenged in dealing with complex, unstructured interrogative sentences in DS2, leading to overfitting. On the other hand, in DS1 and DS3, which have more consistent sentence structures, Features3 can effectively handle pragmatic sentence structures, preventing overfitting and enabling better generalization on validation data.

Based on the experiments conducted, both with and without optimization, it can be concluded that optimization provides a mathematical performance improvement, such as more stable training loss reduction and increased accuracy in some models. However, optimization also negatively affects certain model combinations, such as the sharp decline in the F1 score for label F2 in DS2, indicating that optimization caused overfitting for that label. Although the optimized models achieved higher average F1 scores in some combinations, optimization did not adequately address the variation in sentences present in DS2, especially the varying interrogative sentences, which led the model to become too focused on patterns in the training data and fail to generalize well to the validation or test data. Overall, although optimization improves certain aspects of the model, it is not always better than the model without optimization, especially when the validation data contains substantial variation. Optimization often exacerbates the overfitting issue, as seen with label F2 in DS2.

Table 17: Doctor's answer data segment evaluation results.

Model	**AVG**	**STD**	**MAX**	**MIN**	**Identical Data**	**Segment Error**
HTCN+DS1	0.9085	0.1074	1.0000	0.3309	2478	46.62%
HTCN+DS2	0.9081	0.1237	1.0000	0.2996	4508	43.65%
HTCN+DS3	0.9568	0.0850	1.0000	0.4444	706	29.33%
LDA+DS1	0.5963	0.0921	0.9041	-0.0249	0	100.00%
LDA+DS2	0.6209	0.0943	1.0000	0.0389	3	99.96%
LDA+DS3	0.4948	0.2022	1.0000	-1.0000	1	99.90%
CVS+DS1	0.7023	0.1061	0.9370	0.1060	0	100.00%
CVS+DS2	0.7022	0.0975	0.9723	0.0000	0	100.00%
CVS+DS3	0.6563	0.1377	1.0000	-0.1678	1	99.90%
K-means+DS1	0.7077	0.0997	1.0000	0.2000	8	99.85%
K-means+DS2	0.7343	0.0900	1.0000	0.1977	11	99.86%
K-means+DS3	0.6750	0.2055	1.0000	-0.3072	2	99.80%
MLPSentFeat+DS1+Features1	0.9729	0.0591	1.0000	0.5077	4203	22.31%
MLPSentFeat+DS1+Features2	0.9738	0.0593	1.0000	0.5077	4288	20.74%
MLPSentFeat+DS1+Features3	0.9821	0.5077	1.0000	0.0518	4669	13.69%
MLPSentFeat+DS2+Features1	0.9830	0.0582	1.0000	0.4440	7231	9.61%
MLPSentFeat+DS2+Features2	**0.9847**	**0.0547**	**1.0000**	**0.4945**	**7288**	**8.90%**
MLPSentFeat+DS2+Features3	0.9829	0.4306	1.0000	0.0583	7216	9.80%
MLPSentFeat+DS3+Features1	0.9754	0.5233	1.0000	0.0672	825	17.42%
MLPSentFeat+DS3+Features2	0.9763	0.5500	1.0000	0.0657	830	17.00%
MLPSentFeat+DS3+Features3	0.9782	0.5653	1.0000	0.0606	828	17.20%

To achieve a model with a high average value and no overfitting, the optimal combination is to use Features1, which is more flexible and general. Features1 can better adapt to the variations present in the data in a more generalizable way, without focusing too much on dominant features that could lead to overfitting. In datasets with high variation, such as DS2, models with Features1 are more flexible in handling varied sentences. Although optimization can improve some aspects, the Features1 combination remains more stable and does not lead to overfitting. Therefore, while optimization provides a mathematical improvement, combinations using Features1 provide more stable results and can achieve high values without sacrificing generalization ability.

4.4. Text segmentation process based on six aspects of communication

This segmentation aims to form small segments based on the classification results and labels. This text segmentation model has two processes: segment formation, annotation, and segment testing. The segment formation and segment annotation process aim to form segments and annotate segments based on aspects of doctor-patient communication. Segment testing aims to determine the performance of segments from the classification stage. The greedy similarity method is used as a text segmentation test. The data used in the text segmentation stage is test data of 20% of the labeled data; the distribution is shown in Table 4. The output of this model is the results of text segmentation of doctors' answers and segment annotations, as well as statistics on greedy similarity.

4.4.1. Segment formation and segment annotation

Several sentences have the same label but are not located next to each other, these sentences become different segments. Alternatively, if sentences are located next to each other but the labels are not the same, these sentences become different segments (Figure 5). This method applies to labels from text classification results and labels from annotators. The resulting segments and their annotations will be similar if the labels from the text classification results match the labels given by the annotator. One segment can contain one or more sentences. One dataset can contain several segments. This is formed based on aspects of doctor-patient communication, as in Figure 1. This segmented doctor's answer data is the final result of this research.

An example of segment formation and segment annotation from ID-222 data is shown in Figure 10. ID-222 data has 23 sentences, while the segmentation results of ID-222 data form 9 segments. Segment information is placed in square brackets [..] and at the sentence's beginning. The segment description contains the segment annotation and serial number, separated by a colon (:). For example, [F1:0], F1 means that the segment has an annotation in the form of F1 or fostering the relationship aspect, the number 0 means the segment serial number in the data, data ID-222. The number enclosed in regular brackets (...) and located in front of the segment or sentence description is the sentence serial number. The colors in Figure 10 are only to make distinguishing one segment from another easier. Segments that have the same segment annotation have the same color.

The method for forming segments based on ID-222 data is that sentences 0 to 2 have the same label in the form of F1, and the three sentences are located next to each other or in sequence so that they can form one segment, namely segment 0. Meanwhile, sentence 21 has the same label as sentences 0, 1, and 2, but the location of sentence 21 does not follow these three sentences, so sentence 21 cannot be a segment with segment 0. So, sentence 21 forms a new segment, namely segment 7. Another example, sentences 2 and 3 are located separately. These two sentences follow each other but have different labels, so the two sentences are not located in the same segment or form different segments. This model forms segments based on the similarity of labeling and sentence order.

The description of the nine ID-222 data segments, namely segment 0, has the annotation F1 (fostering the relationship), consisting of 3 sentences (sentences 0 to 2). Segment 1 has annotation F2 (gathering information), consisting of 3 sentences (sentences 3 and 5). Segment 2 has the annotation F4 (decision making), consisting of 1 sentence (sentence 6). Segment 3 has the annotation F3 (providing information), consisting of 1 sentence (7). Segment 4 has the annotation F6 (enabling disease and treatment-related behavior), consisting of 2 sentences (8 and 9). Segment 5 has the annotation F3, consisting of 9 sentences, 10 to 18. Segment 6 has annotation F4, consisting of 2 sentences (19 and 20). Segment 7 has annotation F1, consisting of 1 sentence (sentence 21). Segment 8 has the annotation F5 (responding to the emotion), consisting of 1 sentence (sentence 22).

4.4.2. Segment evaluation

Evaluation of this segment uses the greedy similarity algorithm, where the classification result segment is compared with the segment from the annotator. Greedy similarity calculates the similarity of sentences in the same segment between the classification segment and the annotator. Comparisons are made in detail based on the sentences in a segment, by comparing classification labels as rows and annotator labels as columns. Algorithm 2 shows the calculation steps for evaluating this segment. The greedy similarity calculation process forms a matrix for each dataset and calculates the greedy similarity value for each dataset. The range of greedy similarity values is between 0 and 1. A value of 1 indicates that the segment of the classification label is the same as the label of the annotator; this situation is called correct data. On the other hand, a greedy similarity value of less than 1 indicates a segment in the classification label that does not match the annotator's label, which is called wrong data.

Example for filling in the matrix elements of data using the greedy similarity algorithm calculation. This matrix filling process applies to all doctors' answer data. An example of filling in the ID-222 data matrix is shown in Table 16. ID-222 data has the same classification and annotator labels, so the segment results are the same. First, the greedy similarity matrix is formed (can be seen in Table 16). The matrix's rows and columns consist of the label's name from the classification process or annotator label, the sequence number of the segment, and the sequence number of the sentences in that segment. For example, row and column 1 contain [F1:0] (0,1,2), meaning that the row and column include segment 0, the annotation label is F1, either the label from the classification process or the annotator label, as well as sentences 0, 1, and 2.

Matrix elements are filled in by comparing rows and columns, such as labels and sentences in the segments. For example, element (0,0) is obtained by comparing segment 0 and segment 0, element (0,1) is obtained by comparing segment 0 and segment 1, and so on.

When comparing rows and columns, pay attention to the labels and sentences in the segments. Conditions for filling in matrix elements are as follows: 1) If the label annotations in the row and column are different, then the matrix element is filled with the number 0. For example, element (0,1), row 1 ([F1:0] (0,1,2)), and column 1 ([F2:1] (3,4,5)) contain 0 because the label for row 1 (F1) is different from the label for column 2 (F2). 2) If the segment annotations in the row and column are the same, then pay attention to the sentences in the row and column segments. For example, elements in row 1 (F1:0] (0,1,2)) and column 1 ([F1:0] (0,1,2)) have the same label, namely F1, so pay attention to the sentences in the row and column segments. Row segment 1 has sentences 0, 1, and 2. Column segment 1 also has sentences 0, 1, and 2. Count the total number of correct sentences, namely, sentences that are the same and are in the classification results segment and the annotator segment simultaneously. For example, in element (0,7), row 1, and column 8, these two segments have the same label annotation, namely F1, but the sentences in these segments are different. The segment in row 1 has three sentences, while the segment in column 8 has one. None of the sentences in the two segments are the same, so the element (0.7) is filled with the value 0.00. Another example is element (0,0), row 1, and column 1. These two segments have three sentences located simultaneously in the row and column, so the correct sentence from this element is 3. Then it is calculated using the formula, namely 2 * the number of correct sentences / (the number of row sentences in a segment+the number of column sentences in a segment). So the elements of row 1 and column 1 are calculated by 2 * 3 / (the number of sentences in row 1 is 3+the number of sentences in column 1 is 3) = 6 / 6 = 1, then the elements of row 1 and column 1 contain 1.00.

Next, the final greedy similarity value is calculated after the ID-222 data matrix elements are filled in. Calculate the final greedy similarity value for all doctors' answer data. The following is an example of calculating the final greedy similarity value for ID-222 data by 1 selecting several of the largest values from the greedy similarity matrix. The largest value cannot be located in one row with the same column as another row. The yellow greedy similarity matrix elements (Table 16) indicate that the greedy similarity matrix has the largest values. The values appear not to be located in the same row and column. For example, the largest value 1 is 1, is in the elements of row 1 and column 1. The largest value 2 is 1, is in the elements of row 2 and column 2. 2) Calculate the greedy similarity value with the formula, namely 2 * the total largest value / (number of matrix row sentences+number of matrix column sentences). The calculation of the ID-222 greedy similairty value is 2 * (1+1+1+1+1+1+1+1+1) / (9+9) = 18 / 18 = 1. So the ID-

222 greedy similarity value is 1. It means that the ID-222 data has identical segments between the segments from the classification process and the segments from the annotator.

Thus, the greater the final greedy similarity value of the data, the higher the level of similarity between the segments resulting from the classification process and the segments from the annotator, and it will even be identical if the final greedy similarity value reaches 1. On the other hand, if the final greedy similarity value is close to 0, the segments formed by the classification process are very different from those formed by the annotator.

4.4.3. Text segmentation experiment

After calculating the final greedy similarity value for all the doctors' response data, statistical analysis was performed on these values by calculating the average (AVG), standard deviation (STD), minimum value (MIN), maximum value (MAX), the number of identical data points, and the segment error percentage. The AVG metric measures the average similarity between the automatic segmentation (classification) results and the annotator across the entire dataset. This metric provides an overall view of the method's performance, where a higher AVG value indicates that the segmentation produced is more relevant and closer to the desired result. The STD metric helps assess the method's reliability under different conditions by measuring the variation or spread of the segmentation results produced by the model. A lower STD value indicates that the model's segmentation results are more consistent, with fewer fluctuations between the segments produced, and vice versa. The MIN metric is used to identify the worst-case performance of the method, which may reveal specific data or conditions that cause the method to fail or significantly differ from the annotator. The MAX metric is used to identify the best-case scenario where the method performs most optimally. If many values approach the MAX, the method often performs very well. If the MAX is far from the AVG, it suggests that only a few cases are optimal. The number of identical data points refers to data that produces the same segments between the classification process results and the annotator's segments. A higher identical data value indicates that the model is more effective in producing segments that match those of the system (classification) and the annotator, and vice versa. The segment error percentage, calculated based on the number of incorrect data points compared to the total data points, is performed because the data types vary. A lower segment error percentage indicates that the model is more accurate in dividing the data into relevant segments with fewer errors, and vice versa. The experimental results of text segmentation formation on the doctor's response data based on six communication aspects are presented in Table 17.

Statistical analysis of the final greedy similarity value for all doctors' answer data using the MLPSentFeat model with various DS1, DS2, and DS3 data, and various feature sentences Features1, Features2, and Features3, is shown in Table 17. Apart from that, segment formation using the HTCN, CVS [38], LDA [116], and K-means [117]. Previous researchers have used these methods to form text segments, so the author used this method to segment the doctor's answer data based on six aspects. The parameters used by the LDA method to form segments are the topics displayed based on the number of unique labels for each data (n_components=Label value), max_iter=20, random state=0, doc_topic_prior=0.1, and topic_word prior=0.1. The main parameter in the K-Means method for segment formation is the number of clusters (k), which is determined by the number of communication aspects presented in each doctor's response. Although there are six communication aspects, not all responses contain all six aspects. The number of clusters can be optimized using the elbow method to achieve more optimal segmentation results [118]. However, in this study, the number of clusters could not be optimized, as the value of k was predetermined based on the number of communication aspects identified in each response. Thus, the number of clusters directly represents the number of segments formed for each response. The other parameters used in the K-Means method were: init='k-means++', max_iter=300, n_init=100, tol=0.0001, and random_state=seed. The parameters used by the CVS method in forming segments are the minimum sentence limit for data, namely 4, penalty = get_penalty([sentence_vectors], segment_len), and sentence_vectors = vecr.transform(sentence_text).dot(wrdvecs).

Table 17 presents the results of the segmentation evaluation of doctors' answer data using various methods based on five metrics. The MLPSentFeat method performs better with the highest AVG and lowest STD values, while the LDA method records the lowest performance. The LDA+DS3 method has the lowest AVG of 0.4948 with the highest STD among all methods of 0.2022, as well as an extreme MIN value of -1.0000, which indicates instability and low segmentation accuracy. In contrast, the HTCN and K-means methods show moderate performance with AVG values ranging from 0.6750 to 0.9568, depending on the dataset used. The CVS model also shows moderate performance with the highest AVG of 0.7023 on the DS1 combination. The overall data shows that the combination of model and dataset greatly influences the stability and accuracy of segmentation results.

More specifically, the MLPSentFeat model, combined with various datasets and sentence features, demonstrates the most consistent and superior performance. Based on the evaluation results conducted on two segmentation models, MLPSentFeat+DS2+ Features2 and MLPSentFeat+DS1+ Features3, several significant differences are observed in terms of segmentation quality, consistency, and accuracy. First, regarding the AVG metric, the MLPSentFeat+DS2+ Features2 model shows a slightly higher value (0.9847) than the MLPSentFeat+DS1+Features3 model (0.9821). This AVG metric indicates that MLPSentFeat+DS2+ Features2 provides slightly better overall segmentation quality, although the difference is minimal. Second, in the STD metric, MLPSentFeat+DS2+Features2 has a very low STD value (0.0547), indicating that this model is more

Table 18: Data segment evaluation results from optimized classification.

Model	AVG	STD	MAX	MIN	Identical Data	Segment Error
MLPSentFeat+DS1+Features1	0.9822	0.0495	1.0000	0.5077	4616	14.68%
MLPSentFeat+DS1+Features2	0.9760	0.0578	1.0000	0.5000	4384	18.96%
MLPSentFeat+DS1+Features3	0.9768	0.5077	1.0000	0.0562	4422	18.26%
MLPSentFeat+DS2+Features1	0.9863	0.0508	1.0000	0.4840	7329	8.39%
MLPSentFeat+DS2+Features2	0.9825	0.0574	1.0000	0.4494	7171	10.36%
MLPSentFeat+DS2+Features3	**0.9867**	**0.5091**	**1.0000**	**0.0503**	**7346**	**8.18%**
MLPSentFeat+DS3+Features1	0.9752	0.0678	1.0000	0.5505	823	17.70%
MLPSentFeat+DS3+Features2	0.9783	0.0627	1.0000	0.5505	843	15.70%
MLPSentFeat+DS3+Features3	0.9768	0.5527	1.0000	0.0640	826	17.4%

consistent in producing stable segments without large fluctuations between the generated segments. In contrast, the MLPSentFeat+DS1+Features3 model has a higher STD value (0.5077), which indicates greater variation in the segmentation results. However, this does not mean the model is unstable. The low STD in MLPSentFeat+DS2+Features2 shows better stability, which is crucial for applications requiring consistency in segmentation across varying data. Next, in the Segment Error Percentage metric, MLPSentFeat+ DS1+Features3 records a lower value (6.62%) than MLPSentFeat+DS2+Features2 (8.90%). This finding indicates that MLPSentFeat+DS1+Features3 is more effective in separating the data into relevant segments, with fewer segmentation errors. This metric suggests that MLPSentFeat+DS1+Features3 has higher segmentation accuracy, although MLPSentFeat+DS2+Features2 demonstrates better consistency. Although MLPSentFeat+DS2+Features2 is slightly superior in terms of segmentation quality and consistency, MLPSentFeat+DS1+Features3 excels in segmentation accuracy, with a lower Segment Error Percentage. Therefore, if segmentation accuracy is the top priority, MLPSentFeat+DS1+Features3 can be considered the best model. However, if consistency and stability in segmentation results are prioritized, MLPSentFeat+DS2+ Features2 would be the more suitable choice.

Based on the segmentation evaluation results from the optimized classification presented in Table 18, the MLPSentFeat+DS2+Features3 model demonstrates the best overall segmentation performance. Two primary metrics indicate this conclusion: the lowest segment error percentage of 8.18% and the highest number of identical data points, totaling 7346. These results imply that this model achieves the most accurate segmentation alignment with the manual annotations while producing the fewest segmentation errors among all evaluated models. However, when considering performance stability across data samples (as reflected by the STD), the MLPSentFeat+DS1+Features1 model yields the lowest STD value of 0.0495, indicating the most consistent performance. Despite this stability, the model shows a relatively high segment error percentage of 14.68%, suggesting that it is less accurate in segmenting the data, even though it maintains a stable output. In contrast, the MLPSentFeat+DS2+Features1 model offers a well-balanced compromise, with the second-lowest segment error percentage (8.39%) and the second-lowest STD (0.0508). These findings demonstrate that the model is relatively accurate and reasonably stable. Furthermore, its average similarity score (AVG) is 0.9863, close to the highest observed value (0.9867).

In summary, if the primary objective is to maximize segmentation accuracy, the MLPSentFeat+DS2+ Features3 model is the most suitable choice, despite its relatively lower stability (STD = 0.5091). On the other hand, if a more stable yet sufficiently accurate model is preferred, then MLPSentFeat+DS2+Features1 represents a more balanced alternative. Conversely, although MLPSentFeat+DS1+Features1 demonstrates the highest stability, it is not recommended when segmentation accuracy is the main priority, due to its relatively high error rate.

In comparing the results from Tables 17 and 18, which presented the segment evaluation of the MLPSentFeat model across different feature sets and datasets, notable improvements and variations in performance were observed. In Table 17, the MLPSentFeat+DS2+Features2 model achieved the best performance, with an average of 0.9847 and a segment error percentage of 8.90%. The worst-performing model in this table was MLPSentFeat+DS1+Features3, which showed a segment error percentage of 13.69%. Upon optimization, as shown in Table 18, the same model, MLPSentFeat+DS2+ Features2, demonstrated an improvement in performance, achieving a segment error percentage of 8.39%, thus confirming the positive impact of the optimization process. However, the worst-performing model in the optimized classification, MLPSentFeat+DS3+Features1, showed a significant increase in segment error to 17.70%, highlighting the negative effect of the optimization on this particular variation. Overall, while the optimization process resulted in slight improvements in the performance of the best-performing models, it also led to a deterioration in the performance of certain configurations, particularly for those utilizing DS3. These findings underscored the necessity of careful model selection and an understanding of dataset characteristics to achieve optimal segmentation performance.

5 Conclusions

This study proposes a novel approach to text segmentation focused on communication aspects, specifically doctor-patient communication. Unlike classical segmentation methods that rely on topic similarity, this approach segments text based on the communicative function of each sentence, thus more accurately reflecting the structure of interactions in a medical context. The developed model, MLPSentFeat, explicitly integrates sentence features into its architecture, proving effective in handling data with highly imbalanced label distributions—such as cases where one label has only a few dozen data points, while others have thousands—without requiring synthetic data or complex balancing techniques.

The MLPSentFeat model consists of four main stages: pseudo-labeling, classification, text segmentation, and optimization. In the classification stage, sentence features are formed using a combined approach, with the LR algorithm filtering relevant words for each label and InfoGain selecting the most informative features. The results of the classification experiments show that integrating these sentence features significantly improves the model's sensitivity to variations in linguistic structure, particularly in recognizing non-standard sentences, such as questions commonly found in the information-gathering aspect. It is evident from the experiment, which shows a sharper decline in the F1-score for F2 (gathering information) compared to other labels, both before and after optimization. This decline is attributed to the high variability in question sentence structures in the DS2 dataset. The best classification model before optimization was MLPSentFeat+DS2+Features2, with an F1-score of 0.9925.

In the text segmentation stage, the MLPSentFeat+DS2+Features2 model also yielded the best results, with an average F1-score of 0.9847 and the lowest segment error percentage of 8.90%, indicating that the model performs well in segmenting text. After parameter optimization, the model remained stable, although it did not achieve the highest average F1-score. However, this challenge led to the model experiencing overfitting.

The use of text normalization, data size, cleanliness, and the alignment of sentence features with the semantic structure of sentences are key factors in preventing overfitting. Features2 has been proven well-suited for high-variability data, such as DS2, despite fluctuations in label F2, as seen in the MLPSentFeat+DS2+Features2 model. On the other hand, Features3 is better suited for well-structured datasets like DS1 and DS3, where the model is more effective in capturing pragmatic patterns, especially in sentences with a more formal structure. It is evident from the combination of Features3 with DS1 and DS3 in label F2, which shows F1-score stability, with the Features3 and DS3 combination achieving the highest F1-score for label F2, both before and after optimization. Features1 exhibited better stability across the entire dataset, meaning it functions well with both structured and varied data, though its results were more generic and less in-depth compared to Features2 or Features3. It is reflected in the stable average F1-score seen in combinations of Features1 with various datasets. The Features1+DS1 combination was the best after optimization, with an average F1-score of 0.9926. Moreover, this model was stable across all labels and did not experience overfitting. However, this model showed a relatively high segment error percentage after text segmentation, indicating that MLPSentFeat+DS1+ Features1 was not yet optimal in completing the text segmentation task.

No model was dominant in the experiment results, though several models exhibited distinct advantages and stability. Based on the goal of segmenting text based on communication aspects, the best model produced was MLPSentFeat+DS2+Features3 after optimization. This model completed the text segmentation task with the lowest segment error percentage of 8.18%. Furthermore, optimizing the model's parameters had a positive impact, as evidenced by the number of models achieving an average F1-score above 0.99 and the improved classification consistency in most model combinations. Nevertheless, not all models experienced an increase in average F1-score or accuracy after optimization. This finding is due to the optimization's inability to handle data diversity, even though mathematically, the optimization improved the model's performance. The MLPSentFeat model has been trained and tested with various health domains, showing that the model can adapt well to cross-domain applications.

Future work includes applying the MLPSentFeat model to doctor responses in different languages or other domains. Additionally, efforts will further improve the model's ability to identify questions with varying sentence structures, not limited to formal question structures.

References

[1] X. Kou, “Personalized Doctor Recommendation Method Based on Patient Online Consultation Text,” *2021 IEEE Int. Conf. Comput. Sci. Electron. Inf. Eng. Intell. Control Technol. CEI 2021*, pp. 31–34, Sep. 2021, doi: 10.1109/CEI52496.2021.9574585.

[2] M. Tanis, T. Hartmann, and F. te Poel, “Online Health Anxiety and Consultation Satisfaction: A Quantitative Exploratory Study on Their Relations,” *Patient Educ. Couns.*, vol. 99, no. 7, pp. 1227–1232, Jul. 2016, doi: 10.1016/J.PEC.2016.01.021.

[3] Y. Rahmawati, D. Siahaan, and D. Purwitasari, “Text Segmentation Methods for Annotation on eHealth Consultation with Interview Function Labels: A Comparative Study,” *8th Int. Conf. Softw. Eng. Comput. Syst. ICSECS 2023*, pp. 72–77, 2023, doi: 10.1109/ICSECS58457.2023.10256340.

[4] S. Hobma, P. Ram, A. Muijtjens, C. van der Vleuten, and R. Grol, “Effective Improvement of Doctor-Patient Communication: A Randomised Controlled Trial,” *Br. J. Gen. Pract.*, vol. 56, no.

529, pp. 580–586, 2006.

[5] Y. Rahmawati, D. Siahaan, D. Purwitasari, R. A. Wijayanti, and Violita Elsha Salshabillah, "Text Classification based on Sentence Features and Preprocessing Settings for Labeling eHealth Consultation Answer," in *7th International Seminar on Research of Information Technology and Intelligent Systems*, Yogyakarta: IEEE, 2024, pp. 1065–1070. doi: 10.1109/ISRITI64779.2024.10963536.

[6] M. Hadryan and J. Rutecka-Góra, "Readability and Clarity of Individual Pension Product Contracts," *Int. J. Semiot. Law*, vol. 36, no. 4, pp. 1749–1777, Aug. 2023, doi: 10.1007/S11196-023-09997-8/TABLES/12.

[7] A. van den Brink-Muinen *et al.*, "Doctor–patient communication in different European health care systems:: Relevance and performance from the patients' perspective," *Patient Educ. Couns.*, vol. 39, no. 1, pp. 115–127, Jan. 2000, doi: 10.1016/S0738-3991(99)00098-1.

[8] J. W. Y. Kee, H. S. Khoo, I. Lim, and M. Y. H. Koh, "Communication Skills in Patient-Doctor Interactions: Learning from Patient Complaints," *Heal. Prof. Educ.*, vol. 4, no. 2, pp. 97–106, Jun. 2018, doi: 10.1016/J.HPE.2017.03.006.

[9] R. Ramdani, A. F. Huda, and M. A. Bijaksana, "Text Segmentation Based on Word Embedding on Indonesian Quran Translation by Greedy with Window Approaching," *Int. Conf. Inf. Commun. Technol.*, vol. 7, Jul. 2019, doi: 10.1109/ICOICT.2019.8835338.

[10] S. Martikainen *et al.*, "Text-based Patient – Doctor Discourse Online And Patients' Experiences of Empathy," *Comput. Support. Coop. Work CSCW An Int. J.*, vol. 33, no. 4, pp. 1151–1175, Dec. 2024, doi: 10.1007/s10606-023-09481-8.

[11] A. King and R. B. Hoppe, "'Best Practice' for Patient-Centered Communication: A Narrative Review," *J. Grad. Med. Educ.*, vol. 5, no. 3, pp. 385–393, Sep. 2013, doi: 10.4300/JGME-D-13-00072.1.

[12] M. Mostafapour, J. D. Smith, J. H. Fortier, and G. E. Garber, "Beyond Medical Errors: Exploring the Interpersonal Dynamics in Physician-Patient Relationships Linked to Medico-Legal Complaints," *BMC Health Serv. Res.*, vol. 24, no. 1, pp. 1–11, Dec. 2024, doi: 10.1186/S12913-024-11457-3/TABLES/2.

[13] N. A. Mohd Salim, N. S. Roslan, R. Hod, S. F. Zakaria, and S. K. Adam, "Exploring Critical Components of Physician-Patient Communication: A Qualitative Study of Lay and Professional Perspectives," *Healthcare*, vol. 11, no. 2, p. 162, Jan. 2023, doi: 10.3390/HEALTHCARE11020162.

[14] J. F. Ha and N. Longnecker, "Doctor-Patient Communication: A Review," *Ochsner J.*, vol. 10, no. 1, p. 38, 2010, doi: 10.3329/jbcps.v32i2.26036.

[15] E. Bindels, C. Verberg, A. Scherpbier, S. Heeneman, and K. Lombarts, "Reflection revisited: How physicians conceptualize and experience reflection in professional practice - A qualitative study," *BMC Med. Educ.*, vol. 18, no. 1, pp. 1–10, May 2018, doi: 10.1186/S12909-018-1218-Y/TABLES/1.

[16] F. B. Romeiro, V. G. Pais, G. Humphris, and M. Figueiredo-Braga, "Patients' Emotional Expressions and Clinicians' Responses in Oncology – From Recognition to Exploration of Concerns," *PEC Innov.*, vol. 6, p. 100374, Jun. 2025, doi: 10.1016/J.PECINN.2025.100374.

[17] Y. Luo *et al.*, "How about Trust in Physician-Patient Relationship? A Concept Analysis of Physicians' Perspectives," *Patient Educ. Couns.*, vol. 112, p. 107709, Jul. 2023, doi: 10.1016/J.PEC.2023.107709.

[18] W. Wallace *et al.*, "The diagnostic and triage accuracy of digital and online symptom checker tools: a systematic review," 2022. doi: 10.1038/s41746-022-00667-w.

[19] W. Wallace *et al.*, "The Diagnostic and Triage Accuracy of Digital and Online Symptom Checker Tools: a Systematic Review", doi: 10.1038/s41746-022-00667-w.

[20] V. R. Hananto, U. Serdült, and V. Kryssanov, "A Text Segmentation Approach for Automated Annotation of Online Customer Reviews, Based on Topic Modeling," *Appl. Sci. 2022, Vol. 12, Page 3412*, vol. 12, no. 7, p. 3412, Mar. 2022, doi: 10.3390/APP12073412.

[21] A. Tagarelli and G. Karypis, "A Segment-based Approach to Clustering Multi-Topic Documents," *Knowl. Inf. Syst.*, vol. 34, no. 3, pp. 563–595, 2013, doi: 10.1007/s10115-012-0556-z.

[22] B. Qolomany, M. Maabreh, A. Al-Fuqaha, A. Gupta, and D. Benhaddou, "Parameters Optimization of Deep Learning Models using Particle Swarm Optimization," *2017 13th Int. Wirel. Commun. Mob. Comput. Conf. IWCMC 2017*, pp. 1285–1290, Jul. 2017, doi: 10.1109/IWCMC.2017.7986470.

[23] C. Bi *et al.*, "Optimizing a Multi-Layer Perceptron Based on an Improved Gray Wolf Algorithm to Identify Plant Diseases," *Math. 2023, Vol. 11, Page 3312*, vol. 11, no. 15, p. 3312, Jul. 2023, doi: 10.3390/MATH11153312.

[24] F. Z. El-Hassani, M. Amri, N. E. Joudar, and K. Haddouch, "A New Optimization Model for MLP Hyperparameter Tuning: Modeling and Resolution by Real-Coded Genetic Algorithm," *Neural Process. Lett.*, vol. 56, no. 2, pp. 1–31, Apr. 2024, doi: 10.1007/S11063-024-11578-0/FIGURES/13.

[25] R. Anggoro, W. Suadi, A. M. Shiddiqi, R. Ijtihadie, S. Lili, and D. W. L. Pamungkas, "The Development of Blackhole Attack in AODV Routing Protocol," *CENIM 2020 - Proceeding Int. Conf. Comput. Eng. Network, Intell. Multimed. 2020*, no. Cenim, pp. 309–314, 2020, doi:

10.1109/CENIM51130.2020.9297962.

[26] J. Li, B. Chiu, S. Shang, and L. Shao, "Neural Text Segmentation and its Application to Sentiment Analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 828–842, Feb. 2022, doi: 10.1109/TKDE.2020.2983360.

[27] H. Misra, F. Yvon, O. Cappé, and J. Jose, "Text Segmentation: A Topic Modeling Perspective," *Inf. Process. Manag.*, vol. 47, no. 4, pp. 528–544, Jul. 2011, doi: 10.1016/J.IPM.2010.11.008.

[28] T. Brants, F. Chen, and I. Tsochantaridis, "Topic-Based Document Segmentation with Probabilistic Latent Semantic Analysis," in *International Conference on Information and Knowledge Management, Proceedings*, Association for Computing Machinery (ACM), 2002, pp. 211–218. doi: 10.1145/584792.584829.

[29] F. Y. Y. Y. Choi, P. Wiemer-Hastings, J. Moore, F. Y. Y. Choi., P. Wiemer-Hastings, and J. Moore, "Latent Semantic Analysis for Text Segmentation," *Proc. 2001 Conf. Empir. Methods Nat. Lang. Process.*, vol. 102, no. 7, pp. 109–117, 2001.

[30] S. R. Joty, G. Carenini, and R. T. Ng, "Topic Segmentation and Labeling in Asynchronous Conversations," *J. Artif. Intell. Res.*, vol. 47, pp. 521–573, Feb. 2014, doi: 10.1613/jair.3940.

[31] M. Bayomi and S. Lawless, "C-HTS: A concept-based hierarchical text segmentation approach," *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 1519–1528, 2019.

[32] J. W. Wu, J. C. R. Tseng, and W. N. Tsai, "An efficient linear text segmentation algorithm using hierarchical agglomerative clustering," *Proc. - 2011 7th Int. Conf. Comput. Intell. Secur. CIS 2011*, pp. 1081–1085, 2011, doi: 10.1109/CIS.2011.240.

[33] M. A. Hearst, "TextTiling : Segmenting Text into Multi-Paragraph Subtopic Passages," *1997 Assoc. Comput. Linguist.*, vol. 23, no. 1, pp. 33–64, 1997.

[34] F. Y. Y. Y. Choi, "Advances in Domain Independent Linear Text Segmentation," *1st Meet. North Am. Chapter Assoc. Comput. Linguist.*, vol. 1, pp. 26–33, 2000.

[35] G. Glavaš, F. Nanni, and S. P. Ponzetto, "Unsupervised Text Segmentation Using Semantic Relatedness Graphs," *Assoc. Comput. Linguist.*, pp. 125–130, Aug. 2016, doi: 10.18653/v1/S16-2016.

[36] S. Zhafiirah, Y. Rahmawati, D. Purwitasari, and D. O. Siahaan, "Semantic Relatedness Graph for Text Segmentation of Patient-Centered Communications in Question-Answer Data," *2024 9th Int. Conf. Informatics Comput. ICIC 2024*, pp. 1–6, 2024, doi: 10.1109/ICIC64337.2024.10957542.

[37] A. A. Alemi and P. Ginsparg, "Text Segmentation based on Semantic Word Embeddings," Mar. 2015. doi: https://doi.org/10.48550/arXiv.1503.05543.

[38] V. Gupta, G. Zhu, A. Yu, and D. E. Brown, "A Comparative Study of the Performance of Unsupervised Text Segmentation Techniques on Dialogue Transcripts," *2020 Syst. Inf. Eng. Des. Symp. SIEDS 2020*, pp. 0–5, 2020, doi: 10.1109/SIEDS49339.2020.9106639.

[39] B. S. Wiguna, D. Purwitasari, and D. O. Siahaan, "Deep Learning Approach for Health Question and Answer Text Segmentation based on Physician-Patient Communication Aspect," *Procedia Comput. Sci.*, vol. 234, pp. 213–221, 2024, doi: 10.1016/j.procs.2024.02.168.

[40] I. Pak and P. L. Teh, "Text segmentation techniques: A critical review," *Stud. Comput. Intell.*, vol. 741, no. July, pp. 167–181, 2018, doi: 10.1007/978-3-319-66984-7_10.

[41] I. Ghinassi, "Unsupervised Text Segmentation via Deep Sentence Encoders: a First Step Towards a Common Framework for Text-Based Segmentation, Summarization and Indexing of Media Content," *Proc. 2nd Int. Work. Data-driven Pers. Telev. ACM Int. Conf. Interact. Media Exp. (IMX 2021), June 2021*, vol. 2, no. September, pp. 1–15, 2021, doi: 10.5281/zenodo.4744399.

[42] J. Zhu, C. Zhang, and M. Y. Ma, "Multi-aspect rating inference with aspect-based segmentation," *IEEE Trans. Affect. Comput.*, vol. 3, no. 4, pp. 469–481, 2012, doi: 10.1109/T-AFFC.2012.18.

[43] S. Chowdhury, H. Yerebakan, Y. Shinagawa, and P. S. Yu, "MedTextSeg: A Deep Dual Sequential Model for Section Segmentation in Medical Reports," *Proc. - 2021 IEEE Int. Conf. Big Data, Big Data 2021*, pp. 1582–1588, 2021, doi: 10.1109/BigData52589.2021.9671518.

[44] J. Eisenstein, "Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion," *NAACL HLT 2009 - Hum. Lang. Technol. 2009 Annu. Conf. North Am. Chapter Assoc. Comput. Linguist. Proc. Conf.*, vol. 1, no. June, pp. 353–361, 2009, doi: 10.3115/1620754.1620806.

[45] K. Ganesan and M. Subotin, "A General Supervised Approach to Segmentation of Clinical Texts," *Proc. - 2014 IEEE Int. Conf. Big Data, IEEE Big Data 2014*, pp. 33–40, 2014, doi: 10.1109/BIGDATA.2014.7004390.

[46] E. Apostolova, D. S. Channin, D. Demner-Fushman, J. Furst, S. Lytinen, and D. Raicu, "Automatic Segmentation of Clinical Texts," 2009, doi: 10.1109/IEMBS.2009.5334831.

[47] T. Edinger, D. Demner-Fushman, A. M. Cohen, S. Bedrick, and W. Hersh, "Evaluation of Clinical Text Segmentation to Facilitate Cohort Retrieval," *AMIA Annu. Symp. Proc.*, vol. 2017, p. 660, 2018.

[48] N. Sadoughi *et al.*, "Detecting Section Boundaries in Medical Dictations: Toward Real-Time Conversion of Medical Dictations to Clinical Reports," in *International Conference on Speech and Computer*, Springer Verlag, 2018, pp. 563–573. doi: 10.1007/978-3-319-99579-3_58/TABLES/5.

[49] F. Ginter, H. Suominen, S. Pyysalo, and T.

Salakoski, “Combining Hidden Markov Models and Latent Semantic Analysis for Topic Segmentation and Labeling: Method and Clinical Application,” *Int. J. Med. Inform.*, vol. 78, no. 12, pp. 1–6, 2009, doi: 10.1016/j.ijmedinf.2009.02.003.

[50] A. J. C. Trappey, C. V. Trappey, M. H. Chao, and C. T. Wu, “VR-Enabled Engineering Consultation Chatbot For Integrated And Intelligent Manufacturing Services,” *J. Ind. Inf. Integr.*, vol. 26, p. 100331, Mar. 2022, doi: 10.1016/J.JII.2022.100331.

[51] X. Luo, X. Li, Y. M. Goh, X. Song, and Q. Liu, “Application of Machine Learning Technology for Occupational Accident Severity Prediction in the Case of Construction Collapse Accidents,” *Saf. Sci.*, vol. 163, p. 106138, Jul. 2023, doi: 10.1016/J.SSCI.2023.106138.

[52] R. Sun, X. Li, J. Shen, and W. Jin, “An Effective Hybrid Automated Chinese Scoring System for Medical Education,” *Expert Syst. Appl.*, vol. 234, p. 121114, Dec. 2023, doi: 10.1016/J.ESWA.2023.121114.

[53] D. Yokokawa, K. Noda, T. Uehara, Y. Yanagita, Y. Ohira, and M. Ikusaka, “Do Japanese Word-Embedded Representations Obtained in the Academic Corpus Retain the Medical Concepts of ‘Infarction’?,” *Artif. Intell. Med.*, vol. 143, p. 102604, Sep. 2023, doi: 10.1016/J.ARTMED.2023.102604.

[54] Z. Zhu, J. Li, Q. Zhao, and F. Akhtar, “A Dictionary-Guided Attention Network for Biomedical Named Entity Recognition in Chinese Electronic Medical Records,” *Expert Syst. Appl.*, vol. 231, p. 120709, Nov. 2023, doi: 10.1016/J.ESWA.2023.120709.

[55] P. Wen, Y. Ma, and R. Wang, “Systematic Knowledge Modeling and Extraction Methods for Manufacturing Process Planning Based on Knowledge Graph,” *Adv. Eng. Informatics*, vol. 58, p. 102172, Oct. 2023, doi: 10.1016/J.AEI.2023.102172.

[56] C. Y. Wang and J. J. H. Lin, “Utilizing Artificial Intelligence to Support Analyzing Self-Regulated Learning: A Preliminary Mixed-Methods Evaluation from a Human-Centered Perspective,” *Comput. Human Behav.*, vol. 144, p. 107721, Jul. 2023, doi: 10.1016/J.CHB.2023.107721.

[57] M. Y. Landolsi, L. Ben Romdhane, and L. Hlaoua, “Medical Named Entity Recognition using Surrounding Sequences Matching,” *Procedia Comput. Sci.*, vol. 207, pp. 674–683, Jan. 2022, doi: 10.1016/J.PROCS.2022.09.122.

[58] M. Mirzapour, A. Abdaoui, A. Tchechmedjiev, W. Digan, S. Bringay, and C. Jonquet, “French FastContext: A publicly accessible system for detecting negation, temporality and experiencer in French clinical notes,” *J. Biomed. Inform.*, vol. 117, no. December 2020, p. 103733, 2021, doi: 10.1016/j.jbi.2021.103733.

[59] A. A. Shaikina and A. A. Funkner, “Medical Corpora Comparison Using Topic Modeling,” *Procedia Comput. Sci.*, vol. 178, no. 2019, pp. 244–253, 2020, doi: 10.1016/j.procs.2020.11.026.

[60] A. Zehe *et al.*, “Detecting Scenes in Fiction: A New Segmentation Task,” pp. 3167–3177, Apr. 2021, doi: 10.18653/v1/2021.eacl-main.276.

[61] J. Abbas, C. Zhang, and B. Luo, “EnsCL-CatBoost: A Strategic Framework for Software Requirements Classification,” *IEEE Access*, vol. 12, no. October, pp. 127614–127628, 2024, doi: 10.1109/ACCESS.2024.3452011.

[62] F. Alzamzami, M. Hoda, and A. El Saddik, “Light Gradient Boosting Machine for General Sentiment Classification on Short Texts: A Comparative Evaluation,” *IEEE Access*, vol. 8, pp. 101840–101858, 2020, doi: 10.1109/ACCESS.2020.2997330.

[63] K. He, R. Mao, T. Gong, C. Li, and E. Cambria, “Meta-Based Self-Training and Re-Weighting for Aspect-Based Sentiment Analysis,” *IEEE Trans. Affect. Comput.*, vol. 14, no. 3, pp. 1731–1742, Jul. 2023, doi: 10.1109/TAFFC.2022.3202831.

[64] J. Li *et al.*, “KRA: K-Nearest Neighbor Retrieval Augmented Model for Text Classification,” *Electron. 2024*, vol. 13, no. 16, p. 3237, Aug. 2024, doi: 10.3390/ELECTRONICS13163237.

[65] E. Elsaeed, O. Ouda, M. M. Elmogy, A. Atwan, and E. El-Daydamony, “Detecting Fake News in Social Media Using Voting Classifier,” *IEEE Access*, vol. 9, pp. 161909–161925, 2021, doi: 10.1109/ACCESS.2021.3132022.

[66] R. Mahadeva, M. Kumar, S. P. Patole, and G. Manik, “Desalination Plant Performance Prediction Model Using Grey Wolf Optimizer Based ANN Approach,” *IEEE Access*, vol. 10, pp. 34550–34561, 2022, doi: 10.1109/ACCESS.2022.3162932.

[67] A. Al Bataineh, D. Kaur, and S. M. J. Jalali, “Multi-Layer Perceptron Training Optimization Using Nature Inspired Computing,” *IEEE Access*, vol. 10, pp. 36963–36977, 2022, doi: 10.1109/ACCESS.2022.3164669.

[68] M. Alazab, R. A. Khurma, A. Awajan, and M. Wedyan, “Digital Forensics Classification Based on a Hybrid Neural Network and the Salp Swarm Algorithm,” *Electron.*, vol. 11, pp. 1–17, 2022, doi: https://doi.org/10.3390/electronics11121903.

[69] M. S. Khan, F. Jabeen, S. Ghouzali, Z. Rehman, S. Naz, and W. Abdul, “Metaheuristic Algorithms in Optimizing Deep Neural Network Model for Software Effort Estimation,” *IEEE Access*, vol. 9, pp. 60309–60327, 2021, doi: 10.1109/ACCESS.2021.3072380.

[70] S. Udaiyakumar, C. L. Chinnadurrai, C. Anandhakumar, and S. Ravindran, “Electricity Price Forecasting using Multilayer Perceptron Optimized by Particle Swarm Optimization,” *Proc. - 2nd Int. Conf. Smart Technol. Commun. Robot. 2022, STCR 2022*, vol. 2, no. 12, pp. 1–6, 2022, doi: 10.1109/STCR55312.2022.10009414.

[71] N. Zlobinsky, D. L. Johnson, A. K. Mishra, and A.

A. Lysko, "Comparison of Metaheuristic Algorithms for Interface-Constrained Channel Assignment in a Hybrid Dynamic Spectrum Access - Wi-Fi Infrastructure WMN," *IEEE Access*, vol. 10, pp. 26654–26680, 2022, doi: 10.1109/ACCESS.2022.3155642.

[72] A. Melman and O. Evsutin, "Comparative Study of Metaheuristic Optimization Algorithms for Image Steganography Based on Discrete Fourier Transform Domain," *Appl. Soft Comput.*, vol. 132, p. 109847, Jan. 2023, doi: 10.1016/J.ASOC.2022.109847.

[73] T. Nagadurga, R. Devarapalli, and Ł. Knypiński, "Comparison of Meta-Heuristic Optimization Algorithms for Global Maximum Power Point Tracking of Partially Shaded Solar Photovoltaic Systems," *Algorithms 2023, Vol. 16, Page 376*, vol. 16, no. 8, p. 376, Aug. 2023, doi: 10.3390/A16080376.

[74] E. ; Bayona *et al.*, "Comparative Analysis of Metaheuristic Optimization Methods for Trajectory Generation of Automated Guided Vehicles," *Electron. 2024, Vol. 13, Page 728*, vol. 13, no. 4, p. 728, Feb. 2024, doi: 10.3390/ELECTRONICS13040728.

[75] G. Innocenti, F. Corinto, and E. Kaya, "A Comprehensive Comparison of the Performance of Metaheuristic Algorithms in Neural Network Training for Nonlinear System Identification," *Math. 2022, Vol. 10, Page 1611*, vol. 10, no. 9, p. 1611, May 2022, doi: 10.3390/MATH10091611.

[76] S. Ghaemifard and A. Ghannadiasl, "A Comparison of Metaheuristic Algorithms for Structural Optimization: Performance and Efficiency Analysis," *Adv. Civ. Eng.*, vol. 2024, no. 1, p. 2054173, Jan. 2024, doi: 10.1155/2024/2054173.

[77] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, 2014, doi: 10.1016/j.advengsoft.2013.12.007.

[78] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, IEEE, Aug. 1995, pp. 1942–1948. doi: 10.1109/ICNN.1995.488968.

[79] S. Mirjalili and A. Lewis, "The Whale Optimization Algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016, doi: 10.1016/J.ADVENGSOFT.2016.01.008.

[80] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper Optimisation Algorithm: Theory and application," *Adv. Eng. Softw.*, vol. 105, pp. 30–47, Mar. 2017, doi: 10.1016/J.ADVENGSOFT.2017.01.004.

[81] S. Arora and S. Singh, "Butterfly Optimization Algorithm: a Novel Approach for Global Optimization," *Soft Comput.*, vol. 23, no. 3, pp. 715–734, Feb. 2019, doi: 10.1007/S00500-018-3102-4/FIGURES/10.

[82] X.-S. Yang, "Firefly Algorithm, Lévy Flights and Global Optimization," 2010, doi: https://doi.org/10.48550/arXiv.1003.1464.

[83] X.-S. Yang, "A New Metaheuristic Bat-Inspired Algorithm," vol. 284, pp. 65–74, 2010, doi: https://doi.org/10.48550/arXiv.1004.4170.

[84] G. Dhiman and V. Kumar, "Seagull Optimization Algorithm: Theory and its Applications for Large-Scale Industrial Engineering Problems," *Knowledge-Based Syst.*, vol. 165, pp. 169–196, Feb. 2019, doi: 10.1016/J.KNOSYS.2018.11.024.

[85] P. Bajpai, "Genetic Algorithm – an Approach to Solve Global Optimization Problems," *Indian J. Comput. Sci. Eng.*, vol. 1, no. 3, pp. 199–206, 2016.

[86] D. Karaboga and B. Akay, "Artificial Bee Colony (ABC) Algorithm on Training Artificial Neural Networks," *2007 IEEE 15th Signal Process. Commun. Appl. SIU*, 2007, doi: 10.1109/SIU.2007.4298679.

[87] S. Juanita, D. Purwitasari, I. K. E. Purnama, A. F. Abdillah, and M. H. Purnomo, "Multi-Label Classification for Doctor's Behavioral Pattern Matching During Online Medical Interview using Machine Learning," *Int. J. Electr. Eng. Informatics*, vol. 15, no. 3, pp. 435–452, 2023, doi: 10.15676/ijeei.2023.15.3.5.

[88] S. I. G. Situmeang, R. K. Lubis, F. J. N. Siregar, and B. J. D. C. Panjaitan, "Movie Summarization based on Indonesian Subtitles with Restricted Boltzmann Machine," *Proc. 2019 4th Int. Conf. Sustain. Inf. Eng. Technol. SIET 2019*, no. 2014, pp. 338–342, 2019, doi: 10.1109/SIET48054.2019.8986127.

[89] R. C. Morales-Hernández, J. G. Juagüey, and D. Becerra-Alonso, "A Comparison of Multi-Label Text Classification Models in Research Articles Labeled with Sustainable Development Goals," *IEEE Access*, vol. 10, pp. 123534–123548, 2022, doi: 10.1109/ACCESS.2022.3223094.

[90] M. K. Dahouda and I. Joe, "A Deep-Learned Embedding Technique for Categorical Features Encoding," *IEEE Access*, vol. 9, pp. 114381–114391, 2021, doi: 10.1109/ACCESS.2021.3104357.

[91] S. Qiu, Q. Liu, S. Zhou, and W. Huang, "Adversarial Attack and Defense Technologies in Natural Language Processing: A Survey," *Neurocomputing*, vol. 492, pp. 278–307, Jul. 2022, doi: 10.1016/j.neucom.2022.04.020.

[92] W. Yang, R. Zhang, J. Chen, L. Wang, and J. Kim, "Prototype-Guided Pseudo Labeling for Semi-Supervised Text Classification," vol. 1, pp. 16369–16382, doi: 10.18653/v1/2023.acl-long.904.

[93] D. Jiménez, O. J. Gambino, H. Calvo, D. Jiménez, O. J. Gambino, and H. Calvo, "Pseudo-Labeling Improves News Identification and Categorization with Few Annotated Data," *Comput. y Sist.*, vol. 26, no. 1, pp. 183–193, 2022, doi: 10.13053/CYS-26-1-4163.

[94] H. L. Vu, K. T. W. Ng, A. Richter, and C. An,

"Analysis of Input Set Characteristics and Variances on K-Fold Cross Validation for a Recurrent Neural Network Model on Waste Disposal Rate Estimation," *J. Environ. Manage.*, vol. 311, p. 114869, Jun. 2022, doi: 10.1016/J.JENVMAN.2022.114869.

[95] L. Chen *et al.*, "SSD Drive Failure Prediction on Alibaba Data Center Using Machine Learning," *2022 IEEE Int. Mem. Work. IMW 2022 - Proc.*, 2022, doi: 10.1109/IMW52921.2022.9779284.

[96] S. Muhamad Isa, G. Nico, and M. Permana, "Indobert for Indonesian Fake News Detection," *ICIC Express Lett.*, vol. 16, no. 3, pp. 289–297, 2022, doi: 10.24507/icicel.16.03.289.

[97] A. R. Baskara, M. A. S. Jati, M. Maulida, Y. Sari, N. F. Mustamin, and E. S. Wijaya, "Classification of User Reviews for Software Maintenance in Indonesian Language Using IndoBERT-BiLSTM (Case Study: MyPertamina)," *2023 8th Int. Conf. Informatics Comput. ICIC 2023*, 2023, doi: 10.1109/ICIC60109.2023.10381946.

[98] A. B. Y. A. Putra, Y. Sibaroni, and A. F. Ihsan, "Disinformation Detection on 2024 Indonesia Presidential Election using IndoBERT," *2023 Int. Conf. Data Sci. Its Appl. ICoDSA 2023*, pp. 350–355, 2023, doi: 10.1109/ICODSA58501.2023.10277572.

[99] Z. Liu, Q. Zhang, and L. Gao, "Optimizing Sentence Embedding with Pseudo-Labeling and Model Ensembles: A Hierarchical Framework for Enhanced NLP Tasks," Jan. 2025, doi: https://doi.org/10.48550/arXiv.2501.15876.

[100] A. Kag, L. M. Jenila Livingston, L. M. Livingston Merlin, and L. G. X. Agnel Livingston, "Multiclass Single Label Model for Web Page Classification," *2019 Int. Conf. Recent Adv. Energy-Efficient Comput. Commun. ICRAECC 2019*, Mar. 2019, doi: 10.1109/ICRAECC43874.2019.8995087.

[101] B. R. Awangditama *et al.*, "Web Service Classification From Network Flow Using Undersampling Technique," *2023 Int. Conf. Adv. Mechatronics, Intell. Manuf. Ind. Autom. ICAMIMIA 2023 - Proc.*, pp. 1–6, 2023, doi: 10.1109/ICAMIMIA60881.2023.10427874.

[102] D. Kim, J. Koo, and U. M. Kim, "OSP-Class: Open Set Pseudo-labeling with Noise Robust Training for Text Classification," *Proc. - 2022 IEEE Int. Conf. Big Data, Big Data 2022*, pp. 5520–5529, 2022, doi: 10.1109/BIGDATA55660.2022.10020273.

[103] G. Patel, J. Allebach, and Q. Qiu, "Seq-UPS: Sequential Uncertainty-aware Pseudo-label Selection for Semi-Supervised Text Recognition," *Proc. - 2023 IEEE Winter Conf. Appl. Comput. Vision, WACV 2023*, pp. 6169–6179, 2023, doi: 10.1109/WACV56688.2023.00612.

[104] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. doi: 10.18653/v1/D19-1410.

[105] I. Hussain, O. Ormandjieva, and L. Kosseim, "Automatic QQuality Assessment of SRS Text by Means of a Decision-Tree-Based Text Classifier," *Proc. - Int. Conf. Qual. Softw.*, pp. 209–218, 2007, doi: 10.1109/QSIC.2007.4385497.

[106] F. Halim and D. Siahaan, "Detecting Non-Atomic Requirements in Software Requirements Specifications Using Classification Methods," *2019 1st Int. Conf. Cybern. Intell. Syst. ICORIS 2019*, pp. 269–273, Aug. 2019, doi: 10.1109/ICORIS.2019.8874888.

[107] S. Lei, "A feature selection method based on information gain and genetic algorithm," *Proc. - 2012 Int. Conf. Comput. Sci. Electron. Eng. ICCSEE 2012*, vol. 2, pp. 355–358, 2012, doi: 10.1109/ICCSEE.2012.97.

[108] A. Saad, A. P. Engelbrecht, and S. A. Khan, "An Analysis of Differential Evolution Population Size," *Appl. Sci. 2024, Vol. 14, Page 9976*, vol. 14, no. 21, p. 9976, Oct. 2024, doi: 10.3390/APP14219976.

[109] J. Schlauwitz and P. Musilek, "Dimension-Wise Particle Swarm Optimization: Evaluation and Comparative Analysis," *Appl. Sci. 2021*, vol. 11, no. 13, p. 6201, Jul. 2021, doi: 10.3390/APP11136201.

[110] B. Li and R. Y. Xiao, "The Particle Swarm Optimization Algorithm: How to Select the Number of Iteration," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP)*, 2007, pp. 191–193. doi: 10.1109/IIH-MSP.2007.298.

[111] D. Chicco and G. Jurman, "The Advantages of the Matthews Correlation Coefficient (MCC) Over F1 Score and Accuracy in Binary Classification Evaluation," *BMC Genomics*, vol. 21, no. 1, pp. 1–13, Jan. 2020, doi: 10.1186/S12864-019-6413-7/TABLES/5.

[112] P. Stoica and P. Babu, "Pearson–Matthews correlation Coefficients for Binary and Multinary Classification," *Signal Processing*, vol. 222, pp. 1–15, 2024, doi: https://doi.org/10.1016/j.sigpro.2024.109511.

[113] S. Manchanda and G. Karypis, "Text Segmentation on Multilabel Documents: A Distant-Supervised Approach," *Proc. - IEEE Int. Conf. Data Mining, ICDM*, vol. 2018-November, pp. 1170–1175, Dec. 2018, doi: 10.1109/ICDM.2018.00154.

[114] R. Fauzan, D. Siahaan, S. Rochimah, and E. Triandini, "Class Diagram Similarity Measurement: A Different Approach," pp. 215–219, May 2019, doi: 10.1109/ICITISEE.2018.8721021.

[115] M. K. Chegeni, A. Rashno, and S. Fadaei, "Convolution-Layer Parameters Optimization in Convolutional Neural Networks," *Knowledge-Based Syst.*, vol. 261, p. 110210, Feb. 2023, doi: 10.1016/J.KNOSYS.2022.110210.

[116] J. C. Campbell, A. Hindle, and E. Stroulia, "Latent

Dirichlet Allocation: Extracting Topics from Software Engineering Data," *Art Sci. Anal. Softw. Data*, vol. 3, pp. 139–159, 2015, doi: 10.1016/B978-0-12-411519-4.00006-9.

[117] M. Jeong and I. Titov, "Multi-document topic segmentation," in *International Conference on Information and Knowledge Management, Proceedings*, 2010, pp. 1119–1128. doi: 10.1145/1871437.1871579.

[118] D. Marutho, S. Hendra Handaka, E. Wijaya, and Muljono, "The Determination of Cluster Number at k-Mean Using Elbow Method and Purity Evaluation on Headline News," *Proc. - 2018 Int. Semin. Appl. Technol. Inf. Commun. Creat. Technol. Hum. Life, iSemantic 2018*, pp. 533–538, Nov. 2018, doi: 10.1109/ISEMANTIC.2018.8549751.

[119] E. Scheihing, M. Vernier, J. Guerra, J. Born, and L. Cárcamo, "Understanding the Role of Micro-Blogging in B-Learning Activities: Kelluwen Experiences in Chilean Public Schools," *IEEE Trans. Learn. Technol.*, vol. 11, no. 3, pp. 280–293, Jul. 2018, doi: 10.1109/TLT.2017.2714163.

[120] H. G. Daniel Thamrin and T. E. Widagdo, "Natural Language Interface to Database (NLIDB) for Interrogative Sentences with Temporal Elements in Indonesian Language," in *Proceedings of 2022 International Conference on Data and Software Engineering, ICoDSE 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 120–125. doi: 10.1109/ICODSE56892.2022.9971944.

[121] N. Fadhil Abbas, T. Ali Qasim, and H. Abdul-Salam Jasim, "Request Constructions in Classical Arabic versus Modern Arabic," *J. Ethn. Cult. Stud.*, vol. 10, no. 5, pp. 1–15, 2023, doi: 10.2307/48756354.

[122] E. Cahyaningtyas and D. Arifianto, "HMM-Based Indonesian Speech Synthesis System with Declarative and Question Sentences Intonation," in *2015 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2015*, Institute of Electrical and Electronics Engineers Inc., Mar. 2016, pp. 153–158. doi: 10.1109/ISPACS.2015.7432756.

[123] N. F. Nabila *et al.*, "Using Probability Theory to Identify the Unsure Value of an Incomplete Sentence," in *Proceedings - UKSim-AMSS 17th International Conference on Computer Modelling and Simulation, UKSim 2015*, Institute of Electrical and Electronics Engineers Inc., Sep. 2016, pp. 497–501. doi: 10.1109/UKSIM.2015.90.

[124] Y. Lv, "Negative Rhetoric Based on the Error Sentences of Foreign Chinese Learners from the Perspective of Computer Aided Syntax Recognition Algorithms," in *Proceedings of the 2nd International Conference on Artificial Intelligence and Smart Energy, ICAIS 2022*, Institute of Electrical and Electronics Engineers Inc., 2022, pp. 704–707. doi: 10.1109/ICAIS53314.2022.9743002.

[125] E. C. Montiel-Vazquez, C. A. Cruz, J. A. R. Uresti, and R. Gomez, "EmpatheticExchanges: Towards Understanding the Cues for Empathy in Dyadic Conversations," *IEEE Access*, 2024, doi: 10.1109/ACCESS.2024.3520000.

[126] J. M. Müller, "Sympathy, Interpersonal Awareness and Acknowledgment," *Topoi*, vol. 41, no. 5, pp. 849–858, Nov. 2022, doi: 10.1007/S11245-022-09803-3/METRICS.