



Multi-Label Text Classification in the Translated Hadith of Bukhari Using A Backpropagation Neural Network

**Yuliana Mislianingsih¹, Nazruddin Safaat Harahap^{2*},
Rahmad Abdillah³, Reski Mai Candra⁴**

^{1,2,3,4}Department of Informatics Engineering, Faculty of Science and Technology,
Universitas Islam Negeri Sultan Syarif Kasim Riau, Indonesia

E-Mail: ¹12250123497@students.uin-suska.ac.id, ²nazruddin.safaat@uin-suska.ac.id,
³rahmad.abdillah@uin-suska.ac.id, ⁴reski.candra@uin-suska.ac.id

Received Apr 31th 2026; Revised May 29th 2026; Accepted Jun 23th 2026; Available Online Jun 26th 2026

Corresponding Author: Nazruddin Safaat Harahap

Copyright © 2026 by Authors, Published by Institut Riset dan Publikasi Indonesia (IRPI)

Abstract

Hadith is the second primary source of Islamic law. This study aims to classify the translations of Sahih al-Bukhari Hadith into three thematic categories simultaneously, namely recommendations, prohibitions, and information, using a multi-label classification approach with the Backpropagation Neural Network (BPNN) method. The dataset consists of 7,000 labeled instances. The preprocessing stage includes case folding and text cleaning without stopword removal or stemming to preserve the original meaning of the text. The novelty of this study lies in the comprehensive exploration of various model configurations, including variations in hidden layer architecture, learning rate values, and n-gram features, to identify the optimal network architecture. Feature extraction was performed using Term Frequency-Inverse Document Frequency (TF-IDF) with n-gram configurations of (1,1) and (1,2) and a maximum of 7,000 features. Three variations of the BPNN architecture (H1, H2, and H3) were evaluated using 10-fold cross-validation. The experimental results show that the simplest architecture, consisting of a single hidden layer with 64 neurons, a learning rate of 0.001, and an n-gram configuration of (1,1), achieved the best performance, with macro accuracy of 91.43%, F1-score of 89.59%, and Hamming loss of 0.0857. These findings indicate that a simple BPNN architecture is highly effective for multi-label Hadith text classification and that increasing the depth of hidden layers does not necessarily improve model performance.

Keywords: Backpropagation Neural Network, Bukhari Hadith, Hamming Loss, Multilabel Classification

1. INTRODUCTION

As the world's most populous Muslim-majority nation, Indonesia has witnessed a steadily increasing volume of academic work in Islamic text analysis, with hadith studies emerging as a particularly active research front [1]. Hadith is one of the sources of law or foundational texts in Islam, after the Qur'an, serving as a reference when an issue is not explained in detail in the Qur'an [2]. Scholars regard hadith as an indispensable compass for directing both individual conduct and communal affairs [3]. One of the most frequently cited hadith collections is Shahih al-Bukhari, which contains thousands of authentic hadiths compiled by Imam Bukhari [4]. The Sahih al-Bukhari is highly authentic and serves as a primary reference, containing a variety of teachings, including recommendations, prohibitions, and practical guidance for daily life [5].

Research conducted [4] classified multilabel hadith texts by comparing the Support Vector Machine and Long Short-Term Memory methods, achieving the best performance on the "Prohibition" label with a Macro F1-score of 82.57%, through the application of a combination of SVM and TF-IDF using a linear kernel, a regularization parameter $C = 1$, without stopword removal, and without data balancing techniques. Additionally, under the parameter setting of Epochs 1–20, Dropout = 0.5, Dense = 128, and Batch Size = 64, the LSTM model reached a competing Macro F1-score of 82.66% on the same label, with both configurations recording a minimum Hamming Loss of 10.452%.

Research that also classified multilabel hadith texts showed that the Random Forest method achieved the highest accuracy rate of 89.48%, slightly better than the Long Short -Term Memory method, which achieved an accuracy of 88.52%, and both methods produced identical Hamming Loss values, namely 0.1048 or equivalent to 89.52% [5]. In a separate experiment, a study applying Artificial Neural Networks (ANN) and Support Vector Machines (SVM), conducted by [6] to classify hadith texts into the categories of advice, information, and prohibitions with testing conducted in scenarios with and without stemming it was shown that

SVM consistently outperformed ANN, achieving a highest accuracy of 85% without stemming and 82% with stemming, while ANN reached only 77% and 72%.

In a study by [7] on sentiment classification of Shopee product reviews using a backpropagation neural network, the accuracy, precision, and recall were 87.5%, 88.2%, and 86.4%, respectively. Furthermore, the study [8] developed a neural network architecture using the backpropagation algorithm to detect COVID-19 vaccine-related hoaxes on Twitter. The dataset consisted of 7,130 tweets, obtained through data scraping in Python and manually labeled. Subsequently, the data underwent preprocessing and TF-IDF weighting before being processed by the model. The results showed that the best model configuration was achieved using the backpropagation method with a 20% training and 80% test split, employing a (3, 5) architecture with two hidden layers, achieving an accuracy of 77.12% and an error rate of 22.88%. Based on a study conducted [9] using a Backpropagation Neural Network (BPNN) for the classification of Diabetes Mellitus (DM), which involves the issue of data imbalance. Using the Pima Indians Diabetes dataset, consisting of 768 data points, a BPNN with a single hidden layer achieved the best performance, with an accuracy of 90.89%.

Furthermore, a study by [10] that used the Backpropagation Neural Network method to classify ontological instances in the Indonesian-language Quran yielded fairly competitive results, although they were not the best among other algorithms. Based on testing on the Tafsir Quraish Shihab dataset, the Backpropagation Neural Network model within the standard TF-IDF framework (Framework 1) achieved an accuracy of 62.26%. However, when integrated into Framework 3, which applies imbalance learning techniques via the Synthetic Minority Oversampling Technique and TF-IDF feature extraction, the Backpropagation Neural Network's accuracy performance drops to 58.49%. By comparison, using Word2vec-based word embeddings in the Backpropagation Neural Network method actually results in a decline in performance. In Framework 2, which uses Word2vec without data balancing, the Backpropagation Neural Network achieved an accuracy of only 42.92%. Meanwhile, in Framework 4, which combines Word2vec with the SMOTE technique, the Backpropagation Neural Network achieved an accuracy of 46.23%. Based on these results, the TF-IDF-based feature weighting approach consistently outperformed Word2vec when applied to the Backpropagation Neural Network method.

Although previous research has extensively addressed the classification of hadith texts, most studies remain limited to single-label classification or rely on conventional machine learning methods such as SVM and Random Forests, without in-depth exploration of artificial neural network architectures. The identified research gap is the lack of studies on the optimization of artificial neural network models, such as Backpropagation Artificial Neural Networks (BPNNs), which have been comprehensively explored through variations in architecture and parameters for multi-label cases in hadith.

This study aims to contribute to the exploration of optimal BPNN model configurations by systematically testing 18 configuration scenarios, each comprising variations in hidden-layer architecture, learning rate, and n-gram TF-IDF. This study employs a multi-output approach, using a single BPNN to classify all three labels (recommendations, prohibitions, and information) simultaneously via three independent output neurons, enabling the model to learn a comprehensive feature representation of hadith texts. Additionally, evaluation is conducted using macro-averaged accuracy, micro-F1 score, and Hamming Loss metrics to provide a comprehensive picture of the model's prediction quality on multi-label data.

2. MATERIALS AND METHOD

This study classifies the Bukhari hadith dataset using a Backpropagation Neural Network. The steps in the classification process are shown in Figure 1.

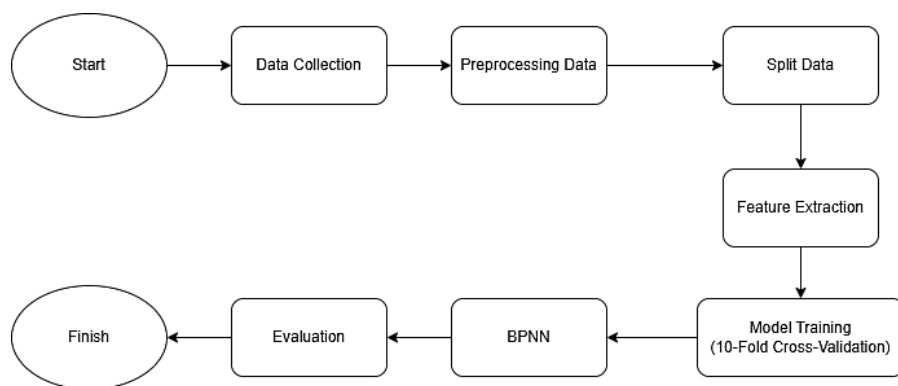


Figure 1. Research Methodology

The research steps include data collection, data preprocessing, data splitting, feature extraction using TF-IDF, training the BPNN model using 10-fold cross-validation, and model evaluation.

2.1. Data Collection

The data collection for this study utilized the Indonesian-translated Bukhari Hadith dataset obtained from study [5].

2.2. Dataset

The dataset used consists of 7,000 multilabel text data points from the Indonesian translation of the Sahih al-Bukhari, categorized into three classes: recommendations (Recommendation), prohibitions (Prohibition), and information (Information). In this approach, the meaning of each hadith is analyzed to determine whether it contains an exhortation (recommendation), a prohibition against an action, or simply conveys information. Since the dataset is written in Indonesian, the original label names Recommendation, Prohibition, and Information are used throughout the labeling process, where Recommendation refers to recommended actions, Prohibition to prohibited actions, and Information to hadith conveying factual or contextual information. This data was obtained from previous research [5]. Table 1 represents the dataset structure in a multilabel format.

Category Abbreviations:

1. Recommendation : Rec
2. Prohibition : Pro
3. Information : Inf

Table 1. Representation of the Bukhari Hadith Translation Dataset

No	Hadith	Category		
		Rec	Pro	Inf
6352	Telah menceritakan kepada kami [Abu 'Ashim] dari [Malik] dari [Sumayy] dari [Abu Shalih] dari [Abu Hurairah] dari Nabi shallallahu 'alaihi wasallam beliau bersabda: "(meninggal) karena sakit perut adalah syahid, dan (meninggal) karena lepra juga syahid."	0	0	1
6391	Telah bercerita kepada kami [Musaddad] dari [Yahya] dari ['Ubaidullah] berkata, telah bercerita kepadaku [Nafi'] dari [Ibnu 'Umar radliallahu 'anhuma] dari Nabi shallallahu 'alaihi wasallam bersabda: "Penyakit panas (demam) berasal dari didihan api jahannam maka redakanlah dengan air".	1	0	1
6572	Telah menceritakan kepada kami [Ali bin Abdullah] telah menceritakan kepada kami [Sufyan] dari [Ayyub] dari [Ibnu Sirin] saya mendengar [Abu Hurairah] bahwa Abu Al Qasim shallallahu 'alaihi wasallam bersabda: "Berilah nama dengan namaku dan jangan kalian menjuluki dengan julukanku."	1	1	1

Table 1 presents a representation of the structure of the Bukhari Hadith translation dataset. A value of '1' indicates that a hadith belongs to a specific category, while a value of '0' indicates the opposite. Each hadith has a complete text that includes the sanad (chain of narrators) and the matn (text of the hadith). These data details are shown in Figure 2.

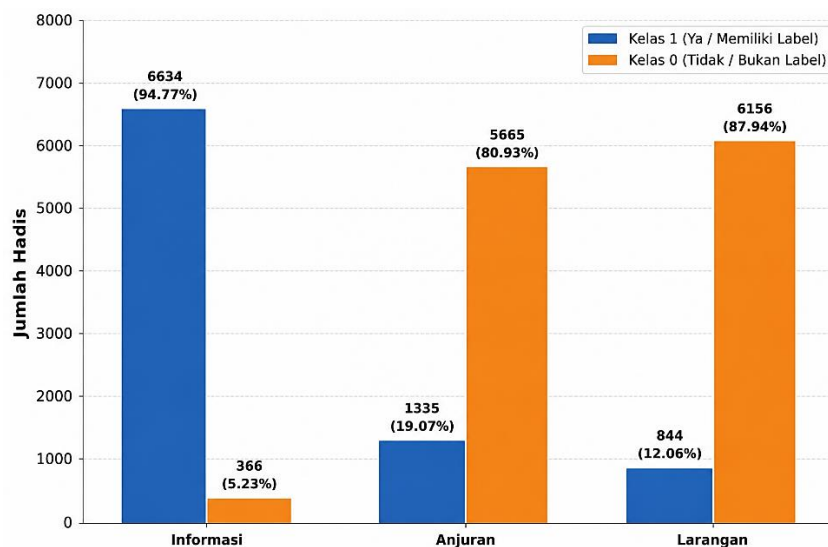


Figure 2. Visualization of Hadith Data Distribution

As shown in Figure 2, the “information” category dominates the dataset, with a significantly larger number of hadiths than the “recommendations” or ‘prohibitions’ categories. In contrast, the “prohibitions” category ranks lowest, with the fewest number of hadiths.

2.3. Preprocessing Data

Preprocessing transforms raw text into a normalized form amenable to machine learning algorithms [5]. Data preprocessing is performed separately and produces a ‘cleaned_text’ column used as model input. The applied steps include [4]:

1. *Case Folding*, all characters are converted to lowercase to enforce uniform text representation.
2. *Cleaning*, punctuation marks, numerals, special symbols, and other extraneous elements are excised. Stemming was deliberately omitted because it risks distorting the nuanced meaning of hadith passages, and stopword removal was similarly excluded because discarding high-frequency function words can degrade Hamming Loss on this corpus [11].

This study does not apply stemming because it can alter the meaning of sentences in the hadith dataset used, and stopword removal is avoided because it can reduce information from the hadith dataset sentences, thereby reducing the accuracy of the Hamming loss [11].

2.4. Split Dataset

The data was randomly split into two parts: 80% for training (5,600 data points) and 20% for testing (1,400 data points) [4]. The value *random_state* = 42 was used to maintain the consistency of this split.

Table 2. Distribution of training data labels

Class	Category		
	Rec	Pro	Inf
0	4.537	4.933	296
1	1.063	667	5.304

Table 2 shows the details regarding the distribution of data in classes 0 and 1 for each label in the training data.

Table 3. Distribution of test data labels

Class	Category		
	Rec	Pro	Inf
0	1.128	1.223	70
1	272	177	1.330

Furthermore, Table 3 shows the distribution of data across classes 0 and 1 for each label category in the test data.

2.5. Feature Extraction

Based on the results of data preprocessing, text features were extracted from the text data and converted to numerical arrays or vectors using TF-IDF [12]. Term Frequency-Inverse Document Frequency (TF-IDF) is a feature-extraction technique used to assign weights to words in a text. These weights are determined by two factors: how frequently a word appears in a specific document and how unique that word is compared to the entire document collection [4]. TF represents the number of occurrences of each word in each document, and IDF represents the number of documents containing a specific word relative to the total number of words in the text [13]. The TF-IDF calculation formula is shown in Equations 1–3 [14].

$$TF_{(i,j)} = \frac{\text{jumlah kemunculan kata } i \text{ dalam dokumen } j}{\text{total kata dalam dokumen } j} \quad (1)$$

$$IDF_{(i,j)} = \log \log \frac{\text{jumlah dokumen}}{\text{jumlah dokumen yang mengandung kata } i} \quad (2)$$

$$TF - IDF = TF \times IDF \quad (3)$$

The TF-IDF configuration used is *max_features* = 7,000, *n-gram* = (1, 1) and (1, 2), *min_df* = 2, *max_df* = 0.9, and *sublinear_tf* = *True* to apply a logarithmic scale to TF. To prevent data leakage, the TF-IDF fitting process is performed separately on each fold during *cross-validation*.

2.6. K-Fold Cross-Validation

Cross-validation is a data mining technique aimed at improving the accuracy of results by dividing the data into two subsets: training data and test data. One of the most commonly used cross-validation methods is K-fold cross-validation [15]. The present study adopts a 10-fold scheme in which the data is divided into ten non-overlapping blocks. In each iteration, nine blocks constitute the training set while the remaining block serves as validation; all ten blocks eventually serve as test data exactly once and as training data nine times [9]. This cycling strategy furnishes an unbiased estimate of generalization performance and guards against overfitting [16]. Additionally, standard deviation is used to measure the consistency of the model's performance across different data subsets, thereby evaluating the overall variation in the model's performance [17]. The workflow of *10-Fold Cross Validation* is illustrated in Figure 3.

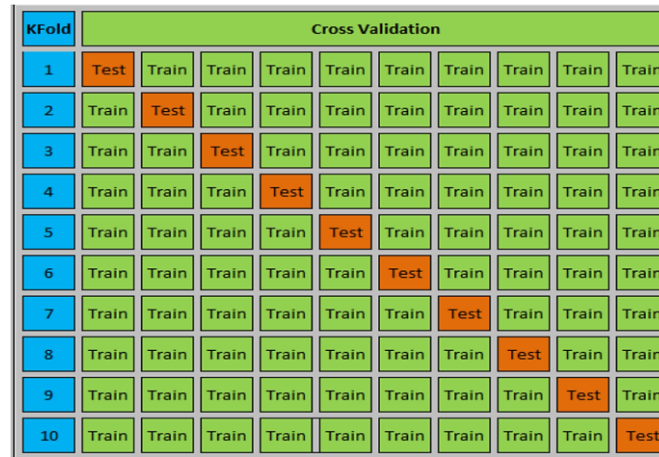


Figure 3. 10-Fold Cross-Validation Workflow [16]

2.7. Backpropagation Neural Network (BPNN)

BPNN is an Artificial Neural Network (ANN) learning algorithm that updates its weights via backpropagation [18]. Its architecture comprises three functional tiers: (1) an input layer that receives the feature vector, (2) one or more hidden layers that apply nonlinear transformations to intermediate representations, and (3) an output layer that produces the final class predictions. The BPNN architecture is illustrated in Figure 4.

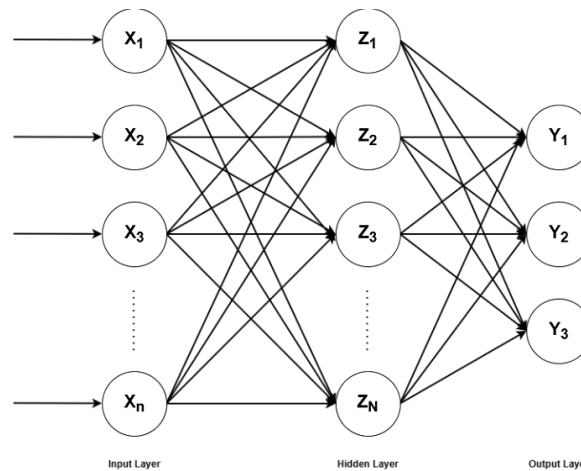


Figure 4. BPNN Architecture

The initial stage involves initializing the weights and biases with random values. During the forward propagation process, the signal from the input layer is passed to the hidden layer and then to the output layer. The calculations for the neuron values in the hidden layer are given by Equations 4 and 5 [19].

$$Z_{in_j} = v_{0j} + \sum_{i=1}^n x_i \cdot v_{ij} \quad (4)$$

$$Z_j = f(z_{in_j}) \quad (5)$$

Next, the values of the neurons in the output layer are calculated using equations 6 and 7.

$$Y_{in_k} = w_{0k} + \sum_{i=1}^n Z_i \cdot w_{jk}; \quad (6)$$

$$Y_k = f(y_{in_k}) = \frac{1}{1+e^{-y_{in_k}}}; \quad (7)$$

Once the output is obtained, a backward propagation process is performed to update the weights and biases based on the resulting error. The correction factor in the output layer is calculated using Equation 8.

$$\delta_l = (t_k - y_k) f'(y_{in_k}) \quad (8)$$

The weights in the output layer are updated using Equation 9.

$$\Delta w_{jk} = \alpha \delta_k z_j; \quad (9)$$

Next, the error is propagated to the hidden layer using Equation 10.

$$\delta_{in_j} = \sum_{k=1}^n \delta_k \cdot w_{jk}; \quad (10)$$

and the weight updates are performed as follows Equation 11:

$$\Delta v_{ij} = \alpha \delta_j x_i; \quad (11)$$

The BPNN implementation in this study uses the MLPClassifier from the scikit-learn library, with a multi-output approach that simultaneously predicts all three labels. Three variations of the hidden layer architecture were tested: 1 hidden layer with 64 neurons (H1), 2 hidden layers with a 128-64 neuron configuration (H2), and 3 hidden layers with a 256-128-64 neuron configuration (H3). Each architecture was combined with three learning rate values (0.001, 0.01, 0.1) and 2 n-gram TF-IDF configurations, namely (1,1) and (1,2), resulting in a total of 18 experimental scenarios. The ReLU activation function was used in all hidden layers, while the output layer used the logistic/sigmoid function automatically determined by scikit-learn for multi-label classification. The Adam optimizer was used, and early stopping was applied during the training of the final model to prevent overfitting.

2.8. Evaluation

At this stage, the classification model built using the Backpropagation Neural Network (BPNN) algorithm is evaluated. The test data, consisting of 1,400 hadiths (20% of the total data), is used to assess the model's performance acquired during the training process. This test data has never been used in the training or validation process (unseen data), so the evaluation results can reflect the model's generalization ability to new data. The evaluation of results reviewed in this study includes: (1) the performance of the BPNN model with various hyperparameter configurations tested on the test data, as assessed by accuracy, F1-score, and Hamming Loss; (2) a comparison of the Hamming Loss value from the best BPNN model against the Hamming Loss value in previous studies. The equations for the evaluation are given in Equations (12)–(16) [5][8].

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (12)$$

$$Precision = \frac{TP}{TP + FP} \quad (13)$$

$$Recall = \frac{TP}{TP + FN} \quad (14)$$

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (15)$$

$$Hamming Loss = \frac{Jumlah\ label\ yang\ salah}{Jumlah\ data \times jumlah\ kategori} \quad (16)$$

Accuracy measures the proportion of correct predictions out of the total data. *Precision* measures how many positive predictions are actually relevant. *Recall* measures how much relevant data is correctly predicted,

while the F1-score is the harmonic mean of precision and recall [20]. *Hamming Loss* is used to assess the overall classification error rate for each category, considering models that use the same preprocessing scheme [5]. In addition to these metrics, this study also uses macro accuracy, which is the average accuracy across the three labels, making it more representative under imbalanced data conditions.

3. RESULTS AND DISCUSSION

This study utilizes the Python programming language to process multi-label hadith translation data. The classification process using the *Backpropagation Neural Network* (BPNN) method is performed immediately after TF-IDF weighting. The classification process was optimized by exploring various network architecture combinations (variations in *hidden layers*, *learning rates*, and *n-grams*) and validated using *10-fold cross-validation* to obtain the most stable and accurate label recognition performance.

3.1. 10-Fold Cross Validation

Table 4 presents the average results of 10-fold cross-validation from 18 scenarios on the training data. The values shown are the mean $\pm$ standard deviation across all folds for each scenario.

Table 4. Average Results of 10-Fold Cross Validation (Average $\pm$ Standard Deviation)

No.	Scenario	Rec	Pro	Inf	Macro	F1 Micro	HL
1.	H1, LR=0.001, n-gram(1,1)	83.89% $\pm 1.74\%$	93.25% $\pm 0.86\%$	94.00% $\pm 1.04\%$	90.38% $\pm 0.74\%$	90.38% ± 0 .74%	0.0962 ± 0.0074
2	H1, LR=0.001, n-gram(1,2)	83.75% $\pm 1.83\%$	93.75% $\pm 0.70\%$	94.36% $\pm 0.91\%$	90.62% $\pm 0.78\%$	88.68% ± 0 .91%	0.0938 ± 0.0078
3	H1, LR=0.01, n-gram(1,1)	84.04% $\pm 1.59\%$	92.54% $\pm 1.16\%$	93.86% $\pm 1.25\%$	90.14% $\pm 0.90\%$	88.16% ± 1 .04%	0.0986 ± 0.0090
4	H1, LR=0.01, n-gram(1,2)	84.05% $\pm 1.96\%$	93.18% $\pm 0.90\%$	94.25% $\pm 0.98\%$	90.49% $\pm 0.90\%$	88.51% ± 1 .02%	0.0951 ± 0.0090
5	H1, LR=0.1, n-gram(1,1)	83.11% $\pm 1.75\%$	92.30% $\pm 1.15\%$	93.20% $\pm 1.51\%$	89.54% $\pm 1.01\%$	87.43% ± 1 .18%	0.1046 ± 0.0101
6	H1, LR=0.1, n-gram(1,2)	81.88% $\pm 2.05\%$	92.12% $\pm 1.37\%$	92.84% $\pm 0.86\%$	88.95% $\pm 0.89\%$	86.89% ± 0 .95%	0.1105 ± 0.0089
7	H2, LR=0.001, n-gram(1,1)	83.41% $\pm 1.74\%$	92.39% $\pm 1.04\%$	93.89% $\pm 1.25\%$	89.90% $\pm 0.96\%$	87.87% ± 1 .12%	0.1010 ± 0.0096
8	H2, LR=0.001, n-gram(1,2)	83.93% $\pm 1.97\%$	92.84% $\pm 1.10\%$	94.09% $\pm 1.21\%$	90.29% $\pm 0.88\%$	88.23% ± 0 .98%	0.0971 ± 0.0088
9	H2, LR=0.01, n-gram(1,1)	84.18% $\pm 1.77\%$	92.54% $\pm 1.10\%$	93.89% $\pm 1.05\%$	90.20% $\pm 0.76\%$	88.15% ± 0 .92%	0.0980 ± 0.0076
10	H2, LR=0.01, n-gram(1,2)	84.04% $\pm 1.67\%$	92.80% $\pm 0.53\%$	94.05% $\pm 1.11\%$	90.30% $\pm 0.70\%$	88.25% ± 0 .80%	0.0970 ± 0.0070
11	H2, LR=0.1, n-gram(1,1)	82.12% $\pm 2.15\%$	91.77% $\pm 1.12\%$	93.75% $\pm 1.51\%$	89.21% $\pm 0.83\%$	87.14% ± 0 .89%	0.1079 ± 0.0083
12	H2, LR=0.1, n-gram(1,2)	81.20% $\pm 3.01\%$	91.95% $\pm 1.31\%$	94.55% $\pm 1.63\%$	89.23% $\pm 1.35\%$	87.32% ± 1 .39%	0.1077 ± 0.0135
13	H3, LR=0.001, n-gram(1,1)	83.75% $\pm 1.61\%$	92.05% $\pm 1.33\%$	93.62% $\pm 1.33\%$	89.81% $\pm 0.81\%$	87.81% ± 0 .95%	0.1019 ± 0.0081
14	H3, LR=0.001, n-gram(1,2)	83.75% $\pm 1.47\%$	92.80% $\pm 1.01\%$	93.93% $\pm 1.46\%$	90.16% $\pm 0.85\%$	88.10% ± 1 .02%	0.0984 ± 0.0085
15	H3, LR=0.01, n-gram(1,1)	85.23% $\pm 1.45\%$	92.86% $\pm 0.97\%$	93.73% $\pm 1.16\%$	90.61% $\pm 0.99\%$	88.59% ± 1 .12%	0.0939 ± 0.0099
16	H3, LR=0.01, n-gram(1,2)	84.25% $\pm 1.73\%$	92.61% $\pm 0.90\%$	94.11% $\pm 0.90\%$	90.32% $\pm 0.63\%$	88.22% ± 0 .68%	0.0968 ± 0.0063
17	H3, LR=0.1, n-gram(1,1)	80.18% $\pm 3.13\%$	90.75% $\pm 2.70\%$	94.71% $\pm 1.33\%$	88.55% $\pm 1.44\%$	86.62% ± 1 .61%	0.1145 ± 0.0144
18	H3, LR=0.1, n-gram(1,2)	78.14% $\pm 5.10\%$	91.36% $\pm 1.53\%$	94.71% $\pm 1.33\%$	88.07% $\pm 2.07\%$	86.01% ± 2 .03%	0.1193 ± 0.0207

Based on Table 4, the *cross-validation* results show that all scenarios with a learning rate of 0.001 consistently produce higher macro accuracy and F1-Scores compared to *learning rates* of 0.01 and 0.1 across all architectures and n-gram variations. Scenario H1 with LR=0.001 and n-gram (1,2) achieved the highest CV macro accuracy of 90.62% $\pm$ 0.78% with a *Hamming Loss* of 0.0938 $\pm$ 0.0078. The lowest standard deviation in macro accuracy was observed in Scenario H3 with LR=0.01 and n-gram (1,2) at $\pm 0.63\%$, indicating excellent model stability across folds despite its absolute performance not being the highest. Conversely, scenarios with LR=0.1, particularly H3 with *n-gram* (1,2), produced the highest standard deviation ($\pm 5.10\%$ for recommendation accuracy and $\pm 2.07\%$ for macro accuracy), reflecting instability in the learning process due to an excessively high *learning rate*.

From an architectural perspective, H1 (one hidden layer) and H2 (two hidden layers) demonstrate CV performance that is relatively on par with H3 (three hidden layers), and often even higher. This indicates that adding *hidden layers* does not always yield a significant performance improvement on this *dataset*.

3.2. Evaluation results on the test data

After the final model was trained using the entire training dataset, an evaluation was conducted on the test data to measure the model's generalization ability. This process began with a qualitative analysis of the prediction results to map the alignment of the hadith text labels in Tables 5 and 6.

Table 5. Incorrect prediction results

Bukhari Hadith Text	Current			Prediction		
	Rec	Pro	Inf	Rec	Pro	Inf
Telah menceritakan kepada kami [Musa bin Isma'il] telah menceritakan kepada kami [Hammam] dari [Ishaq bin 'Abdullah bin Abu Thalhah] dari [Anas radliallahu 'anhu] berkata: "Nabi shallallahu 'alaihi wasallam tidak pernah pulang kepada keluarganya secara mendadak di malam hari dan tidak memasuki rumah kecuali pada pagi atau siang hari".	1	0	1	0	0	1

Based on Table 5, the model failed to predict the 'Recommendation' label because the hadith uses a narrative rather than a direct command format. Since the text uses a standard storytelling style, the TF-IDF feature extraction grouped these words into the dominant 'information' class. This case shows that the BPNN model remains highly dependent on word frequencies, making it difficult to detect minority labels embedded in historical or narrative texts.

Table 6. Correct prediction results

Bukhari Hadith Text	Current			Prediction		
	Rec	Pro	Inf	Rec	Pro	Inf
Telah menceritakan kepada kami [Adam] telah menceritakan kepada kami [Syu'bah] telah menceritakan kepada kami [Mu'awiyah bin Qurrah] dan [Qatadah] dari [Anas bin Malik] radliallahu 'anhu, dari Nabi Shallallahu'alaihi wa sallam bersabda: "Maula suatu kaum adalah bagian diri mereka." atau sebagaimana beliau sabdakan.	0	0	1	0	0	1

Conversely, Table 6 shows a successful prediction where the model correctly identified the Hadith under the 'Information' label. This success occurred because the text uses explicit and clear statement patterns that strongly represent factual information. Since the keywords align directly with the dominant features learned by the network during training, the BPNN model easily assigned the correct label without confusion.

Next, the quantitative evaluation of the 18 scenarios is presented in Table 7, which includes accuracy per label, F1-Score per label, *micro* F1-Score, *macro* accuracy, and *Hamming Loss*.

Table 7. Evaluation results on the test data

No	Scenario	Accuracy Label			F1-Score per Label			F1 Micro	Accuracy Macro	HL
		Rec	Pro	Inf	Rec	Pro	Inf			
1.	H1, LR=0.001, n-gram(1,1)	87.43%	91.50%	95.36%	61.90%	56.73%	97.61%	89.59%	91.43%	0.0857
2	H1, LR=0.001, n-gram(1,2)	86.43%	92.14%	95.14%	60.08%	60.71%	97.49%	89.39%	91.24%	0.0876
3	H1, LR=0.01, n-gram(1,1)	86.64%	91.36%	95.43%	59.44%	60.07%	97.64%	89.30%	91.14%	0.0886
4	H1, LR=0.01, n-gram(1,2)	85.79%	91.36%	94.71%	59.14%	58.42%	97.27%	88.70%	90.62%	0.0938
5	H1, LR=0.1, n-gram(1,1)	84.79%	90.14%	94.57%	56.62%	54.30%	97.17%	87.71%	89.83%	0.1017
6	H1, LR=0.1, n-gram(1,2)	84.50%	90.93%	94.07%	44.78%	60.68%	96.92%	87.47%	89.83%	0.1017
7	H2, LR=0.001, n-gram(1,1)	86.21%	92.21%	95.29%	59.37%	63.61%	97.55%	89.36%	91.19%	0.0881
8	H2, LR=0.001, n-gram(1,2)	85.71%	91.79%	95.57%	56.33%	61.54%	97.71%	89.13%	91.02%	0.0976
9	H2, LR=0.01, n-gram(1,1)	85.71%	91.50%	95.00%	50.25%	58.82%	97.44%	88.63%	90.74%	0.0926

No	Scenario	Accuracy Label			F1-Score per Label			F1 Micro	Accuracy Macro	HL
		Rec	Pro	Inf	Rec	Pro	Inf			
10	H2, LR=0.01, n-gram(1,2)	84.36%	91.86%	95.00%	41.29%	60.69%	97.44%	88.12%	90.40%	0.0960
11	H2, LR=0.1, n-gram(1,1)	82.50%	90.36%	91.61%	55.20%	58.97%	96.74%	87.73%	89.83%	0.1017
12	H2, LR=0.1, n-gram(1,2)	81.21%	90.29%	94.93%	49.33%	50.00%	97.40%	86.65%	88.81%	0.1119
13	H3, LR=0.001, n-gram(1,1)	86.14%	91.64%	94.86%	54.67%	61.39%	97.33%	88.83%	90.88%	0.0912
14	H3, LR=0.001, n-gram(1,2)	84.57%	91.21%	94.93%	56.45%	58.59%	97.38%	88.30%	90.24%	0.0976
15	H3, LR=0.01, n-gram(1,1)	85.43%	91.43%	93.93%	51.20%	62.96%	96.82%	88.03%	90.26%	0.0974
16	H3, LR=0.01, n-gram(1,2)	86.14%	90.79%	94.21%	56.89%	61.49%	96.99%	88.36%	90.38%	0.0962
17	H3, LR=0.1, n-gram(1,1)	79.93%	87.36%	95.00%	48.44%	0.00%	97.44%	84.70%	87.43%	0.1257
18	H3, LR=0.1, n-gram(1,2)	80.57%	87.36%	95.00%	27.66%	0.00%	97.44%	84.19%	87.64%	0.1236

Based on Table 5, Scenario 1 (H1, LR=0.001, n-gram (1,1)) achieved the best performance with a macro accuracy of 91.43%, a micro F1-score of 89.59%, and a Hamming loss of 0.0857. In terms of architecture comparison, H1 consistently yields higher macro accuracy than H2 and H3 at the same learning rate configuration. At LR=0.001, H1's average macro accuracy is 91.33%, higher than H2 (91.10%) and H3 (90.56%). The same pattern is observed in the micro F1-Score, where H1 averages 89.49%, outperforming H2 (89.24%) and H3 (88.56%), proving that simpler architectures are no less effective than more complex ones on this dataset. Regarding the effect of the learning rate, LR=0.001 consistently yields the highest performance across all architecture variations, while LR=0.1 significantly reduces performance, particularly in H3, which produces the lowest macro accuracy of 87.43% and 87.64% in scenarios 17 and 18.

3.3. Analysis and Discussion

The learning rate is the most influential factor on model performance in this study. In Figure 5, LR=0.001 consistently yields the highest macro accuracy and F1-Score, as well as the smallest Hamming Loss across all architecture and n-gram configurations. The baseline in Figure 5 visually corroborates that when the learning rate is fixed at an optimal conservative value LR=0.001, the network behavior is primarily governed by the interaction between architectural depth and feature representation. The steady performance across H1, H2, and H3 configurations shows that a smaller step size grants the optimizer sufficient granularity to navigate the sparse 7.000 dimensional TF-IDF space smoothly, regardless of layer variations. This indicates that hyperparameter stability at low learning rates acts as a buffer against structural changes in the network, maintaining high micro and macro metrics across folds.

Conversely, LR=0.1 significantly degrades performance. In the H3 scenario with LR=0.1 (scenarios 17 and 18), the model is completely unable to predict the "prohibited" label, as evidenced by a prohibited F1-Score of 0.00% caused by TP=0 and FP=0 in the confusion matrix, as shown in Figure 6 and Figure 7.

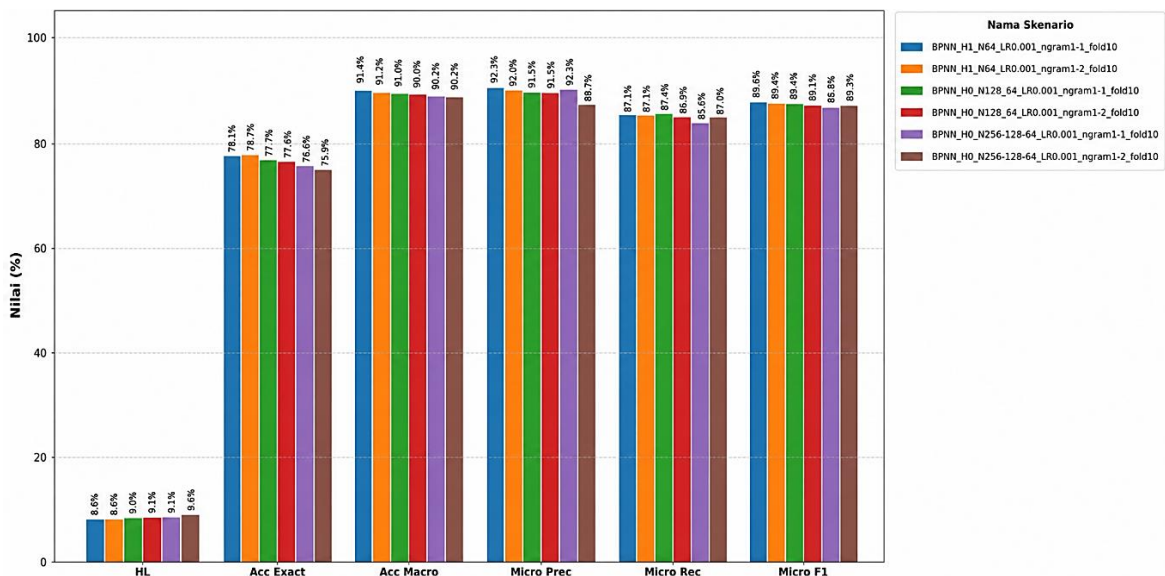


Figure 5. Comparison of test data metrics for LR=0.001

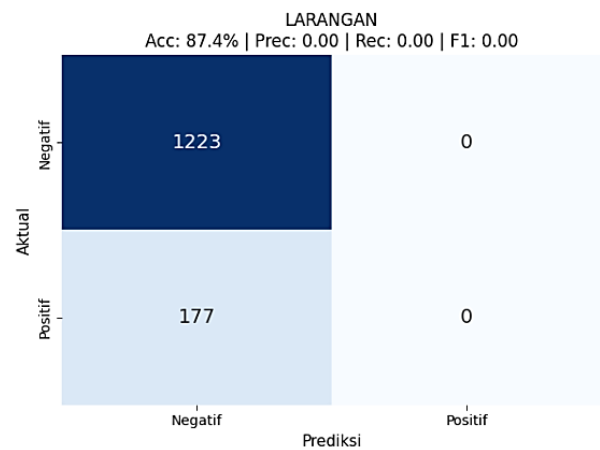


Figure 6. H3 confusion matrix, LR=0.1, n-gram(1,1)

In Figure 6, the "prohibited" label has TP=0, FP=0, and FN=177, meaning all positive "prohibited" data were predicted as negative. This results in Precision=0 and Recall=0, yielding an F1-Score of 0.00% for the "prohibitions" label. This numerical drop shows that when paired with a learning rate that is too high, LR=0.1, within the H3 architecture, the network weights fail to adjust correctly to the minor patterns of this label. Because the "prohibition" class contains the fewest positive data points in the dataset, the aggressive step size causes the model to jump over these limited features during training, forcing the network to simply predict the negative class to lower the total error. The same pattern occurs in Scenario 18 as shown in Figure 7, where the "prohibition" label again yields TP=0 and FN=177, further confirming that the model fails to converge when the learning rate is not optimized for imbalanced data distribution.

This demonstrates that the H3 architecture with a learning rate of 0.1 fails to converge on the "prohibition" label, which has the fewest positive data points (844 out of 7,000). Furthermore, this trend proves that adding more hidden layers (H3) combined with a high learning rate does not help the model learn complex features. Instead, the multi-layered structure makes the network weights more unstable during backpropagation, causing the model to lose its ability to recognize the minority class across both n-gram configurations.

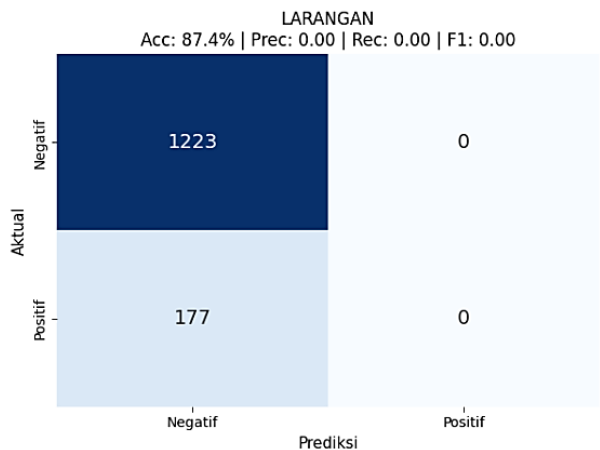


Figure 7. H3 confusion matrix, LR=0.1, n-gram(1,2)

H1 (one hidden layer, 64 neurons) consistently produces the highest macro accuracy and F1-Score compared to H2 and H3. The best overall scenario is H1 with LR=0.001, proving that simpler architectures do not always lose out to more complex ones, as shown in Figure 8.

The superior performance of the H1 architecture over H2 and H3 can be attributed to several interconnected factors beyond dataset size alone. From the perspective of the bias-variance tradeoff, deeper architectures such as H3 introduce a significantly larger number of trainable parameters, which increases the risk of overfitting when the available training data is limited to 5,600 samples. This is consistent with findings that deeper networks require substantially larger datasets to fully benefit from their additional depth, and that network depth alone does not guarantee improved classification performance. Furthermore, from a text data characteristic standpoint, TF-IDF produces high-dimensional but sparse feature vectors (7,000 features) in which most values are zero. Hadith texts in Indonesian translation also tend to contain repetitive structural

patterns in the sanad section, which produces a relatively consistent feature space after TF-IDF transformation. Previous studies have shown that shallow networks tend to be more effective at learning discriminative patterns from such sparse representations. These factors collectively explain why H1 demonstrated more stable and consistent performance, as also evidenced by its lower standard deviation ($\pm 0.74\%$) across 10-fold cross-validation compared to H2 ($\pm 0.96\%$) and H3 ($\pm 0.85\%$). These results align with the study [9], which shows that BPNNs with a single hidden layer yield the best performance, as deeper architectures require more data for optimal convergence. With a total of 7,000 data points split 80:20 and a training data capacity of 5,600, the simple H1 architecture proved more effective than both H2 and H3.

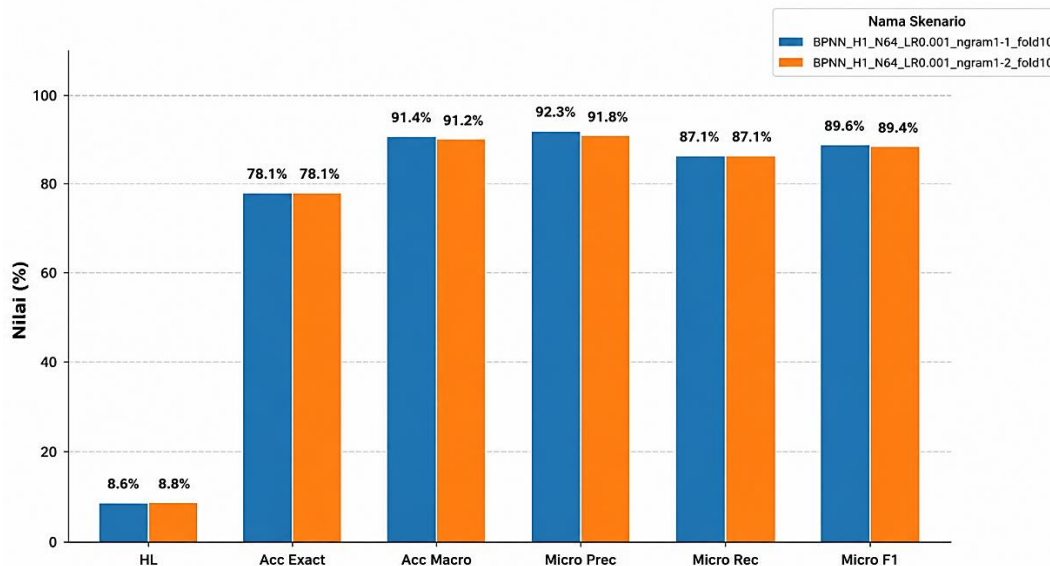


Figure 8. Metrics for H1, LR=0.001

In terms of computational efficiency, the H1 architecture (64 neurons in a single hidden layer) strikes the most optimal balance between processing time and accuracy. Compared to the deeper H3 configuration, which demands significantly higher computational overhead due to a larger parameter space and suffers from divergence at higher learning rates, H1 achieves stable convergence in far fewer training epochs. A comparison of the use of n-gram features (1,1) and (1,2) shows very small differences. In the best-case scenario (H1, LR=0.001) as shown in Figure 8, n-gram (1,1) yields a higher macro accuracy (91.43% vs 91.24%), a higher micro F1-Score (89.59% vs 89.39%), and a smaller Hamming Loss (0.0857 vs 0.0876). These differences are not consistent across all scenarios, so adding bigrams does not provide significant benefits.

The information label consistently achieved the highest accuracy (91.61%–95.57%) and the highest F1-Score (96.74%–97.71%) across all scenarios, supported by the largest number of positive data points (6,634 out of 7,000). In contrast, the recommendation label achieved the lowest accuracy (79.93%–87.43%) and F1-Score (27.66%–61.90%) compared to the other two labels. A similar pattern was observed for the prohibition label, with accuracy ranging from 87.36% to 92.21% and an F1-Score ranging from 0.00% to 63.61%. The significant discrepancy between high macro accuracy and low F1-scores for the 'recommendation' and 'prohibition' labels is a direct consequence of the severe class imbalance shown in Figure 2. The 'information' label provides a massive feature space for the TF-IDF extractor, allowing the network weights to optimize effectively for this dominant class. Conversely, the scarcity of data in the other two labels restricts the model's exposure to their unique semantic keywords during training. As a result, the empirical loss minimization biases the BPNN model toward predicting the negative class for these minority labels, which heavily suppresses their precision and recall, ultimately leading to deflated F1-scores despite high individual accuracies as illustrated in Table 8.

Table 8. Comparison of Model Performance

Method	Macro Accuracy	Hamming Loss
BPNN	91.43%	0.0857
Random Forest [5]	89.48%	0.1048
LSTM [5]	88.52%	0.1045
ANN (single-label) [6]	77.00%	-
SVM [4]	82.57% (F1-Label)	0.1116

Compared to the reference study using the same dataset, the best BPNN model in this study demonstrates significant superiority. Study [5] reported that Random Forest achieved the highest accuracy of 89.48% with a Hamming Loss of 0.1048, while LSTM achieved an accuracy of 88.52% with an identical Hamming Loss of 0.1048. The best BPNN model in this study (scenario H1, LR=0.001, n-gram(1,1)) achieved a macro accuracy of 91.43% and a Hamming Loss of 0.0857, outperforming both RF and LSTM with a 1.95% increase in accuracy over RF and a 0.0191 reduction in Hamming Loss compared to both methods. Furthermore, compared to the study [6] that used a conventional ANN with a maximum accuracy of 77% without stemming, the BPNN in this study demonstrated a highly significant accuracy improvement of 14.43%. This improvement is supported by the use of TF-IDF for feature extraction, the selection of an appropriate architecture through scenario variations, and rigorous evaluation using 10-fold cross-validation.

3.4. Limitations of the Study

This study has several limitations. The dataset used is imbalanced, which results in a low F1- Score for the recommendation and prohibition labels despite the high accuracy per label. Furthermore, a full comparison of datasets with and without stemming has not been fully conducted, and the resulting model has not yet been integrated into a system that can be used directly.

4. CONCLUSION

This study demonstrates that BPNN is capable of classifying Bukhari hadith texts in a multi-label setting with the best performance under scenario H1 (64 neurons), a learning rate of 0.001, and an n-gram of (1,1), yielding a macro accuracy of 91.43%, a micro F1-Score of 89.59%, and a Hamming Loss of 0.0857, surpassing Random Forest (89.48%) and LSTM (88.52%) from the reference study [5]. The insight from this research reveals that network depth and optimization steps are highly sensitive to data distribution, where a moderate H1 architecture provides stable convergence and minimizes misclassification (Hamming Loss), while deeper architectures combined with aggressive learning rates trigger optimization failure on minority classes like 'prohibition'. Future research is recommended to address the issue of data imbalance using techniques such as SMOTE, as well as to explore the use of modern word embeddings and Transformer-based models such as IndoBERT to improve the semantic accuracy of hadith texts. In addition, the resulting optimal model should be implemented into real-world applications to test its scalability and performance when used directly by users.

REFERENCES

- [1] R. Dani and D. Jihanna Ilmi, "Klasifikasi Hadis ditinjau dari Segi Kuantitas dan Kualitas Sanad," *J. Kaji. Pendidik. Islam*, 2024, doi: 10.58561/jkpi.v3i2.127|114.
- [2] A. Rizkia Annur, L. Hidayah Ansadatina, N. Leilani Assrie, N. Heriyani, V. Julia Harinda Putri Jurusan Ekonomi Pembangunan, and F. Ekonomi dan Bisnis, "Hadis Sebagai Ajaran dan Sumber Hukum Islam," *J. Relig. J. Agama, Sos. dan Budaya*, vol. 1, 2023, [Online]. Available: <https://maryamsejahtera.com/index.php/Religion/index>
- [3] S. H. Pratiwi, U. Muhammadiyah Sumatera Barat, and N. Anita, "Klasifikasi Hadits Berdasarkan Kualitas: Kajian Tentang Hadis Sahih, Hasan, dan Dhaif," *J. Media Ilmu*, vol. 3, no. 2, 2024.
- [4] A. Ramadhani, N. Safaat, S. Agustian, I. Iskandar, and S. Sanjaya, "Perbandingan Performa Metode Klasifikasi Teks Multilabel Hadis Terjemahan Bukhari Menggunakan Support Vector Machine dan Long Short Term Memory," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 5, no. 3, pp. 896–907, Jun. 2025, doi: 10.57152/malcom.v5i3.2051.
- [5] R. Z. N. Ahmad, N. S. Harahap, S. Agustian, I. Iskandar, and S. Sanjaya, "Perbandingan Performa Random Forest dan Long Short-Term Memory dalam Klasifikasi Teks Multilabel Terjemahan Hadits Bukhari," *MALCOM Indones. J. Mach. Learn. Comput. Sci.*, vol. 5, no. 3, pp. 862–874, Jun. 2025, doi: 10.57152/malcom.v5i3.2046.
- [6] D. Fatmarani Surianto, M. B. Fajar, and M. Rida Mulia, "Comparative Analysis of the Performance of Hadith Text Classification Methods: A Case Study with ANN and SVM," 2024. [Online]. Available: <https://ojs.unm.ac.id/JESSI/index>
- [7] M. Febrina, "Jurnal J-MendiKKom (Jurnal Manajemen, Pendidikan dan Ilmu Komputer) Klasifikasi Sentimen Ulasan Produk Shopee Menggunakan Jaringan Syaraf Tiruan Backpropagation dan Random Forest," Online, 2025.
- [8] R. Syam, W. Sanusi, and A. A. SYahnur, "Deteksi Information hoaks vaksin covid-19 Di media sosial twitter menggunakan jaringan syaraf tiruan backpropagation," *J. Math. Comput. Stat.*, vol. 7, no. 2, pp. 451–462, Oct. 2024, doi: 10.35580/jmathcos.v7i2.4433.
- [9] R. Azhar, S. Kurnia Gusti, I. Afrianty, and E. Budianita, "Bulletin Of Computer Science Research Perbandingan Teknik Penyeimbang Kelas Pada Multi-Layer Perceptron (MLP) Berbasis Backpropagation Untuk Klasifikasi Diabetes Mellitus," *Media Online*, vol. 5, no. 6, pp. 1304–1314, 2025, doi: 10.47065/bulletincsr.v5i6.804.
- [10] F. S. Utomo, Y. Purwati, M. S. Azmi, L. Shafira, and N. Trinarsih, "Word embedding and imbalanced

- learning impact on Indonesian Quran ontology population,” *Indones. J. Electr. Eng. Comput. Sci.*, vol. 39, no. 1, p. 603, Jul. 2025, doi: 10.11591/ijeecs.v39.i1.pp603-613.
- [11] R. Kustiawan, A. Adiwijaya, and M. D. Purbolaksono, “A Multi-label Classification on Topic of Hadith Verses in Indonesian Translation using CART and Bagging,” *J. MEDIA Inform. BUDIDARMA*, vol. 6, no. 2, p. 868, Apr. 2022, doi: 10.30865/mib.v6i2.3787.
- [12] A. Firizkiansah, A. Muhammad, I. R. Maulana, U. S. Indonesia, and K. Bekasi, “Optimasi Klasifikasi Data Teks Menggunakan Algoritma Logistic Regression dengan TF-IDF dan SMOTE,” vol. 2, no. 1, pp. 29–36, 2025.
- [13] D. C. Hidayati, S. Al Faraby, and A. Adiwijaya, “Klasifikasi Topik Multi Label pada Hadis Shahih Bukhari Menggunakan K-Nearest Neighbor dan Latent Semantic Analysis,” *JURIKOM (Jurnal Ris. Komputer)*, vol. 7, no. 1, p. 140, Feb. 2020, doi: 10.30865/jurikom.v7i1.2013.
- [14] R. Antonius, A. R. Zulkarnain, and H. Irsyad, “Pendekatan TF-IDF , SMOTE , dan SVM dalam Klasifikasi Sentimen Masyarakat terhadap Pemblokiran Judi Online,” vol. 2, no. 3, pp. 115–122, 2024, doi: 10.58369/biit.v2i3.65.
- [15] K. Tanaman Herbal Berdasarkan Tekstur Daun Menggunakan, M. Ezperanza Florentiana Letik, and F. Yusuf Bisilisin, “KETIK : Jurnal Informatika Faatuatua Media Karya,” *J. Inform.*, vol. 01, no. 06, pp. 1–10, 2024.
- [16] A. I. Pradana and V. Atina, “Teknik K-Fold Cross Validation untuk Mengevaluasi Kinerja Mahasiswa,” pp. 239–248, 2024, doi: 10.33364/algoritma/v.21-1.1618.
- [17] F. Pritama, Ekat Rueh Daya Leluni, Yovita, and Jadianan Parhusip, “Analisis Distribusi Kinerja SVM dan KNN Berdasarkan Rata Rata Simpangan Baku dan Stabilitas,” *Informatech J. Ilm. Inform. dan Komput.*, vol. 1, no. 2, 2024, doi: 10.69533/xgqngy78.
- [18] A. Wahyudi, H. Nugroho, H. Nur, and R. Yulida, “Klasifikasi Penyakit Jantung Menggunakan Pendekatan Hybrid Iinformation Gain dan Backpropagation Neural Network (BPNN) Heart Disease Classification Using a Hybrid Approach of Information Gain and Backpropagation Neural Network (BPNN) Prodi Rekam Medis,” pp. 22–27, 2025.
- [19] Y. Artha, C. Saragih, and E. Suryani, “Implementation of Hybrid Backpropagation Neural Network and Particle Swarm Optimization for Predicting Household Electricity Consumption Based On Tariff Categories Implementasi Metode Hybrid Backpropagation Neural Network dan Particle Swarm Optimization u,” vol. 6, no. April, pp. 663–673, 2026.
- [20] K. Izza Ilyasa, M. Zoqi Sarwani, and A. Aris Widodo, “Deteksi Kepuasan Konsumen Terhadap Kualitas Produk Dan Layanan Pada Mie Gacoan Menggunakan Metode Backpropagation,” *JATI (Jurnal Mhs. Tek. Inform.*, vol. 9, no. 5, pp. 9077–9082, 2025, doi: 10.36040/jati.v9i5.15261.