JSCE (Journal of System and Computer Engineering)
Vol.7, No.4, October 2026, pp. 532-541
ISSN (online): 2723-1240
DOI : https://doi.org/10.61628/jsce.v7i4.2963

Research Article Open Access (CC–BY-SA) ■

DNN versus CNN for Abusive Audio Classification in YouTube Shorts

Dwi Atmoko [1,a,*]; Ryan Ari Setyawan [2,b]; Jemmy Edwin Bororing [3,c]

[1.2.3] Department of Informatics, Faculty of Engineering, Universitas Janabadra, Jl. Tentara Rakyat Mataram No. 55–57, Yogyakarta 55231, Indonesia
[a] dwi_atmoko@student.janabadra.ac.id; [b] ryan@janabadra.ac.id; [c]jemmy@janabadra.ac.id
** Corresponding author*

Abstract

Spoken abusive words in YouTube Shorts may escape text-only moderation when captions are absent. This study compares a deep neural network (DNN) using 183-dimensional engineered acoustic features, a convolutional neural network (CNN) using 64 × 87 Mel-spectrograms, and an RBF-kernel support vector machine (SVM) using the same features as the DNN. From 923 retrieved videos, 272 audio segments were curated; 151 source clips were augmented to 906 two-second samples at 22,050 Hz for eight Indonesian abusive keywords. Evaluation used stratified five-fold cross-validation and a 70:30 holdout split. DNN accuracy was 87.08% ± 2.12% across folds and 83.82% on the holdout (macro F1: 83.24%). CNN accuracy was 81.33% ± 30.48% across folds but only 13.97% on the holdout (macro F1: 4.00%), with predictions collapsing to two classes. The SVM achieved 98.50% ± 1.20% across folds and 99.17% holdout accuracy (macro F1: 99.17%). Within this protocol, engineered features and the SVM baseline were more effective than added network depth, while the DNN was more stable than the tested CNN. However, augmentation preceded sample-level splitting, allowing related samples across partitions and potentially inflating performance. The contribution is an audio-only Indonesian keyword comparison with a conventional baseline and explicit analysis of model stability; it does not establish contextual abusiveness or performance on unseen recordings.

Keywords—*Abusive audio classification, Convolutional neural network, Cross-validation, Deep neural network, Support vector machine*

1. Introduction

Abusive language in YouTube Shorts may be conveyed through speech without captions. Consequently, text-only moderation cannot identify every spoken expression. This motivates an audio-only investigation of Indonesian abusive keywords rather than an assumption that results from written-language detection apply unchanged to speech.

Indonesian abusive-language research has largely used social-media text: Salim and Suhartono (2021) investigated YouTube comments with LSTM, and Ibrohim and Budi (2023) reviewed text-centered detection studies. Such research does not establish how accurately the spoken form of an abusive word can be classified without transcription. Moreover, identifying a keyword differs from judging harmful intent, which depends on annotation and context (Waseem, 2016).

Closer short-video studies also address different tasks. Pratama et al. (2026) compared DNN and CNN for binary abusive-content classification using text and image cues from YouTube Shorts; their CNN performed better. Supriyanto et al. (2026) studied Indonesian video toxicity through audio-text fusion. In contrast, this study uses only manually isolated audio to distinguish

eight spoken keywords, with no captions, images, or transcribed text. Binary toxicity results and eight-class lexical accuracy are therefore not directly comparable.

CNNs have been applied successfully to environmental sounds (Piczak, 2015), augmented audio (Salamon & Bello, 2017), and multilingual speech (Sercu et al., 2016). Those settings do not resolve whether Mel-spectrogram learning is preferable to engineered acoustic features for a small Indonesian keyword corpus, especially when augmentation cannot create independent speakers or contexts.

The research gap is limited controlled evidence on feature representation, conventional baselines, and validation stability for Indonesian abusive-keyword audio in short videos. We compare a 183-dimensional feature-based DNN, a 64 × 87 Mel-spectrogram CNN, and a feature-matched RBF-SVM (Cortes & Vapnik, 1995). The research question is how their accuracy, class-specific errors, and fold stability differ under the same small-data evaluation protocol.

This study contributes (1) a documented audio-only corpus construction process, from 923 Shorts and 272 curated segments to 151 source clips augmented into 906 samples; (2) a matched evaluation of DNN, CNN, and RBF-SVM, including class-wise scores and confusion matrices; and (3) an analysis of model instability and possible optimism caused by splitting after augmentation. Its novelty lies in this Indonesian lexical-audio setting and critical baseline comparison, not in a new architecture or full toxicity detector. The operational vocabulary is "anjing," "babi," "bajingan," "bangsat," "goblok," "jancuk," "tai," and "tolol"; recognizing these words does not establish abusive intent.

2. **Method**

Figure 1 summarizes corpus curation, preprocessing, feature extraction, model training, and evaluation with shared class labels and partitions.

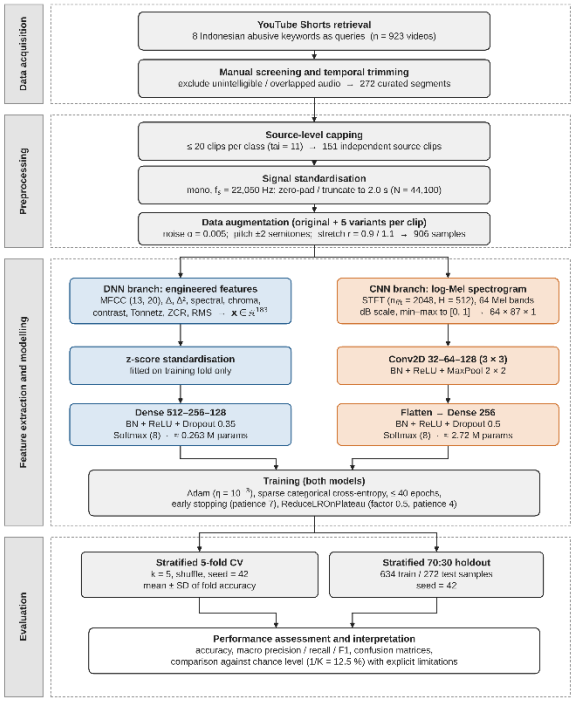


Figure 1. Experimental workflow of the DNN and CNN pipelines

2.1 Data Acquisition, Keyword Selection, and Manual Curation

A total of 923 YouTube Shorts were retrieved using eight Indonesian abusive keywords (Salim & Suhartono, 2021). Recordings were converted to audio, manually trimmed to the target

utterance, and screened to exclude unintelligible or severely distorted segments and dominant overlapping sounds. This yielded 272 curated clips. The lexicon was chosen operationally for retrievability, not to represent every form of Indonesian abuse.

The curated pool contained 25 clips for anjing, 24 for babi, 24 for bajingan, 20 for bangsat, 47 for goblok, 87 for jancuk, 11 for tai, and 34 for tolol. At most 20 source clips were retained per class, producing 151 independent source clips: 20 for each of seven classes and 11 for tai. This source-level imbalance remains despite augmentation; speaker and recording diversity was not measured. Curation involved one rater, so inter-annotator agreement is unavailable.

2.2 Signal Preprocessing and Data Augmentation

Each clip is loaded as a mono discrete-time signal (Jenkins et al., 2002) sampled at $f_s = 22{,}050$ Hz and standardized to a duration of $T_d = 2.0$ s, so that every input contains a fixed number of samples N:

$$x[n] = x_a\left(\frac{n}{f_s}\right), \quad n = 0,1,\dots,N-1, \quad N = f_s T_d = 44{,}100 \tag{1}$$

where $x_a(t)$ is the continuous recording. A clip with L samples is zero-padded at the end when $L < N$ and truncated when $L > N$:

$$\bar{x}[n] = \begin{cases} x[n], & 0 \le n < \min(L,N) \\ 0, & L \le n < N \end{cases} \tag{2}$$

Clips were end-padded or truncated after isolating the keyword. Five augmentations per source clip were generated (Salamon & Bello, 2017): Gaussian noise with amplitude factor 0.005, pitch shifts of ±2 semitones, and time stretching at 0.9 and 1.1:

$$\tilde{x}_{\text{noise}}[n] = \bar{x}[n] + \alpha\,\varepsilon[n], \quad \varepsilon[n] \sim \mathcal{N}(0,1) \tag{3}$$

Pitch shifting by s semitones scales every frequency component by the factor ρ, while time stretching with rate r changes the duration without altering pitch:

$$\rho = 2^{s/12}, \quad s \in \{-2,+2\}; \qquad \tilde{x}_{\text{stretch}}(t) = \bar{x}(r\,t), \quad r \in \{0.9,1.1\} \tag{4}$$

Including originals, 151 clips yielded 906 samples, each returned to two seconds using Eq. (2). Augmentation increases variation but not the number of independent speakers or recording contexts.

2.3 Feature Extraction

Both representations are derived from the short-time Fourier transform (STFT) of the standardized signal (Purwins et al., 2019), computed with a Hann window $w[n]$ of length $N_{\text{fft}} = 2048$ and hop length $H = 512$:

$$X(m,k) = \sum_{n=0}^{N_{\text{fft}}-1} \bar{x}\,[n + mH]\,w[n]\,e^{-j2\pi kn/N_{\text{fft}}} \tag{5}$$

where m is the frame index and k the frequency bin. With centered framing, the number of frames is

$$M = 1 + \left\lfloor \frac{N}{H} \right\rfloor = 1 + \left\lfloor \frac{44{,}100}{512} \right\rfloor = 87 \tag{6}$$

The power spectrum is projected onto B triangular filters $H_b(k)$ spaced uniformly on the Mel scale (Davis & Mermelstein, 1980):

$$f_{\text{mel}} = 2595\log_{10}\left(1 + \frac{f}{700}\right), \qquad S(b,m) = \sum_k H_b\,(k)\,|X(m,k)|^2 \tag{7}$$

For the CNN, $B = 64$ Mel bands are used. Power is converted to the decibel scale and min–max normalized to [0,1], producing an input tensor of 64 × 87 × 1 (McFee et al., 2015):

$$S_{\text{dB}}(b,m) = 10\log_{10}\frac{S(b,m)}{\max S}, \qquad \hat{S}(b,m) = \frac{S_{\text{dB}}(b,m) - \min S_{\text{dB}}}{\max S_{\text{dB}} - \min S_{\text{dB}}} \tag{8}$$

For the DNN, MFCCs are obtained by applying the discrete cosine transform to the log Mel energies, and their temporal dynamics are described by first-order deltas computed over a window of Θ frames; second-order deltas (Δ^2) apply Eq. (10) to the deltas:

$$c_q(m) = \sum_{b=1}^{B} \log S(b,m) \cos\left[\frac{\pi q(b-0.5)}{B}\right] \tag{9}$$

$$\Delta c_q(m) = \frac{\sum_{\theta=1}^{\Theta} \theta\left(c_q(m+\theta) - c_q(m-\theta)\right)}{2\sum_{\theta=1}^{\Theta} \theta^2} \tag{10}$$

The 183-dimensional DNN/SVM feature vector combines MFCCs (13 and 20 coefficients), first- and second-order deltas, spectral centroid, roll-off, bandwidth, chroma, contrast, Tonnetz, tempo, zero-crossing rate, RMS energy, and Mel statistics (Davis & Mermelstein, 1980; Giannakopoulos & Pikrakis, 2014). Appropriate statistics are summarized across frames; vectors are padded or truncated to 183 dimensions. Features are standardized using training-partition statistics only:

$$z_i = \frac{x_i - \mu_i}{\sigma_i}, \quad i = 1, \ldots, 183 \tag{11}$$

The main reproducibility parameters are summarized in Table 1.

Table 1. Main reproducibility parameters of the experiment

Parameter	Value	Parameter	Value
Sampling rate	22,050 Hz	CNN input	64 × 87 × 1 log-Mel
Clip duration	2.0 s; end zero-pad or truncate	FFT / hop length	2048 / 512 samples
Source clips	151 (20 per class; tai = 11)	Cross-validation	Stratified 5-fold, seed = 42
Augmentation	Noise 0.005; pitch ±2; stretch 0.9/1.1	Final split	70:30 stratified, seed = 42
Total samples	906	Epochs	Maximum 40
SVM baseline	RBF kernel on the same 183-D vector	SVM tuning	Grid search; inner 3-fold CV
DNN input	183-D feature vector	Batch size	DNN 64; CNN 32

2.4 Model Architectures

Figure 2 shows the DNN's three dense hidden layers (512, 256, and 128 units), each followed by batch normalization (Ioffe & Szegedy, 2015), ReLU (Glorot et al., 2011), and dropout (Srivastava et al., 2014):

$$\mathbf{h}^{(l)} = \mathrm{Dropout}_p\left(\mathrm{ReLU}\left(\mathrm{BN}\left(\mathbf{W}^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\right)\right)\right), \qquad \mathrm{ReLU}(u) = \max(0,u) \tag{12}$$

where $\mathbf{h}^{(0)} = \mathbf{z}$ is the standardized feature vector. Batch normalization rescales each pre-activation using the mini-batch statistics μ_B and σ_B^2 and the learnable parameters γ and β:

$$\mathrm{BN}(u) = \gamma \frac{u - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} + \beta \tag{13}$$

The CNN uses three 3 × 3 convolutional blocks with 32, 64, and 128 filters, batch normalization, ReLU, and 2 × 2 max pooling (LeCun et al., 2015):

$$Y_c(u,v) = \sum_{c'} \sum_{i=-1}^{1} \sum_{j=-1}^{1} K_{c,c'}(i,j)\, X_{c'}(u+i, v+j) + b_c \tag{14}$$

$$P_c(u,v) = \max_{0 \le i,j \le 1} Y_c(2u+i, 2v+j) \tag{15}$$

A flattened 8 × 10 × 128 tensor feeds a 256-unit dense layer with batch normalization and 0.5 dropout. Both models use eight-class Softmax outputs:

$$\hat{y}_k = \frac{\exp(z_k)}{\sum_{j=1}^{K} \exp(z_j)}, \quad k = 1, \ldots, K \tag{16}$$

Dense, convolutional, and batch-normalization parameter counts are calculated as follows:

$$P_{\mathrm{dense}} = (d_{\mathrm{in}} + 1)\, d_{\mathrm{out}}, \qquad P_{\mathrm{conv}} = (k^2 C_{\mathrm{in}} + 1)\, C_{\mathrm{out}}, \qquad P_{\mathrm{BN}} = 4d \tag{17}$$

The DNN contains 263,048 parameters and the CNN 2,718,344. The CNN's flatten-to-dense layer alone has 2,621,696 (96.4%) parameters, increasing its vulnerability to overfitting on few independent clips.

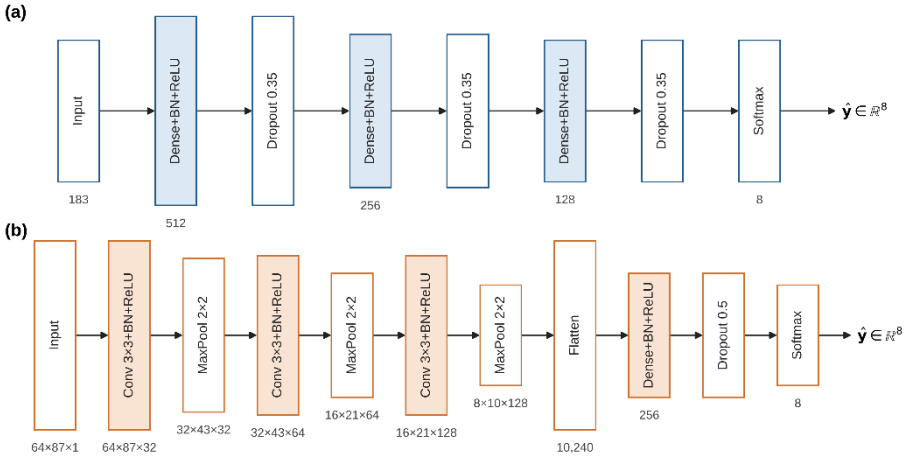


Figure 2. Implemented architectures: (a) feature-based DNN and (b) spectrogram-based CNN; numbers below the blocks denote output dimensions

2.5 Conventional Machine-Learning Baseline

The baseline is an RBF-SVM (Cortes & Vapnik, 1995) trained on the DNN's same 183 standardized features. Its binary soft-margin optimization is:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i, \text{ s.t. } y_i\left(w^{\top}\varphi(x_i) + b\right) \geq 1 - \xi_i,\ \xi_i \geq 0 \tag{18}$$

where φ is the implicit feature map, ξ_i are slack variables, and C controls the trade-off between margin width and training error. The resulting decision function uses the RBF kernel

$$f(x) = \text{sign}\left(\sum_{i=1}^{n}\alpha_i y_i K(x_i, x) + b\right),\ K(x_i, x) = \exp\left(-\gamma\|x_i - x\|^2\right) \tag{19}$$

Here αi denote dual coefficients and γ the kernel width. LIBSVM's one-vs-one strategy builds 28 binary classifiers for the eight classes (Chang & Lin, 2011). Scikit-learn SVC (Pedregosa et al., 2011) was tuned over C ∈ {0.1, 1, 10, 100} and γ ∈ {scale, 0.001, 0.01, 0.1} using inner stratified three-fold cross-validation with macro F1 selection. Scaling was fitted within each training fold, not on its validation set. No class weights were used. The final configuration, C = 10 and γ = 0.01, used 85 support vectors.

2.6 Training Procedure and Implementation

Audio processing used librosa (McFee et al., 2015), neural-network training used TensorFlow/Keras (Chollet et al., 2015), and splitting, metrics, and SVM tuning used scikit-learn (Pedregosa et al., 2011). NumPy and TensorFlow seeds were 42. Both neural models minimized sparse categorical cross-entropy:

$$\mathcal{L} = -\frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}\log\hat{y}_{i,y_i} \tag{20}$$

The loss is optimized with Adam (Kingma & Ba, 2015) at its default initial learning rate $\eta = 10^{-3}$, with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. With gradient $\mathbf{g}_t$, the moment estimates and parameter update are

$$\mathbf{m}_t = \beta_1\mathbf{m}_{t-1} + (1-\beta_1)\,\mathbf{g}_t, \qquad \mathbf{v}_t = \beta_2\mathbf{v}_{t-1} + (1-\beta_2)\,\mathbf{g}_t^2 \tag{21}$$

$$\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} - \eta \frac{\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t}+\epsilon}, \qquad \hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \quad \hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t} \tag{22}$$

Models were trained for up to 40 epochs using Adam, with batch sizes 64 (DNN) and 32 (CNN). Early stopping monitored validation accuracy (patience 7, best weights restored); ReduceLROnPlateau monitored validation loss (factor 0.5, patience 4). Checkpoints kept the best validation-accuracy weights. The optimizer, label set, and evaluation protocol were shared.

2.7 Evaluation Protocol

Stratified five-fold cross-validation (shuffle = True; seed = 42) measured validation stability, with feature scaling fitted within DNN/SVM training folds. A stratified 70:30 split (seed = 42) produced a 272-sample holdout for classification reports and confusion matrices. All three models used identical partitions and metrics: accuracy and class-wise precision, recall, and F1 (Sokolova & Lapalme, 2009):

$$\mathrm{Acc} = \frac{\sum_{k=1}^{K} C_{kk}}{\sum_{i=1}^{K}\sum_{j=1}^{K} C_{ij}}, \qquad P_k = \frac{C_{kk}}{\sum_{i=1}^{K} C_{ik}}, \qquad R_k = \frac{C_{kk}}{\sum_{j=1}^{K} C_{kj}} \tag{23}$$

$$F1_k = \frac{2P_kR_k}{P_k+R_k}, \qquad F1_{\text{macro}} = \frac{1}{K}\sum_{k=1}^{K} F\,1_k, \qquad F1_{\text{weighted}} = \sum_{k=1}^{K}\frac{n_k}{n}F1_k \tag{24}$$

Macro F1 weights classes equally, whereas weighted F1 accounts for their support. Fold accuracies are summarized using their mean and population standard deviation:

$$\bar{A} = \frac{1}{F}\sum_{f=1}^{F} A_f\,, \qquad s_A = \sqrt{\frac{1}{F}\sum_{f=1}^{F}\left(A_f - \bar{A}\right)^2}\,, \qquad F = 5 \tag{25}$$

Random guessing across eight classes has an expected accuracy of 12.5%; no statistical significance is claimed from these unrepeated partitions.

Because augmentation preceded sample-level splitting, variants of one original clip may occur in both training and validation/test partitions. Therefore, even the SVM's high scores may be optimistic: cross-validation mainly diagnoses stability under this protocol, not generalization to independent recordings (Wen et al., 2019).

3. **Results and Discussion**

3.1 Overall Performance and Cross-Validation Stability

Table 2 reports accuracy and macro/weighted scores. DNN cross-validation accuracy was 87.08% ± 2.12% and holdout accuracy 83.82% (macro F1: 83.24%). CNN accuracy was 81.33% ± 30.48% across folds but 13.97% on the holdout (macro F1: 4.00%). The RBF-SVM achieved 98.50% ± 1.20% across folds and 99.17% holdout accuracy (macro F1: 99.17%).

Table 2. DNN, CNN, and SVM baseline performance summary

Evaluation metric	DNN	CNN	SVM (RBF)
5-fold accuracy (mean ± SD)	87.08% ± 2.12%	81.33% ± 30.48%	98.50% ± 1.20%
Holdout accuracy	83.82%	13.97%	99.17%
Macro precision	83.43%	2.74%	99.22%
Macro recall	83.89%	13.19%	99.17%
Macro F1-score	83.24%	4.00%	99.17%
Weighted F1-score	83.79%	4.23%	99.17%

These results concern the implemented models, not universal DNN/CNN superiority. CNNs have succeeded in other audio and speech tasks (Piczak, 2015; Salamon & Bello, 2017; Sercu et al., 2016). The opposing result reported by Pratama et al. (2026), favoring CNN over DNN, also used binary text-image rather than eight-class audio classification. Different modalities and target labels preclude direct architectural ranking across studies.

Figure 3 reveals the CNN's instability: four folds scored 93.92%–98.34%, but fold 3 fell to 20.44%, yielding a 30.48-point standard deviation. DNN folds ranged only from 83.98% to 89.01%. The CNN's strong folds may also benefit from augmented variants crossing split boundaries, while its 13.97% holdout accuracy shows poor robustness in the final split.

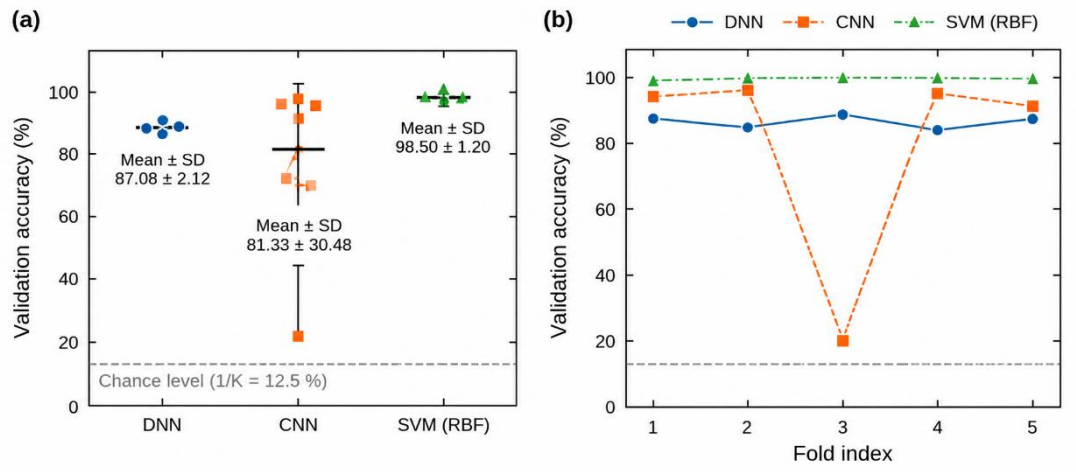


Figure 3. Stratified five-fold validation accuracy: (a) distribution with mean ± SD and (b) fold-wise accuracy; the dashed line marks chance level

3.2 Comparison with the Conventional SVM Baseline

The DNN and SVM use identical engineered inputs, making this comparison more directly informative about classifier choice than the DNN–CNN comparison, which also changes representation. The SVM's 99.17% holdout macro F1 exceeded the DNN's 83.24% by 15.93 percentage points; fold accuracies were 97.00%–99.50%. Thus, added network depth did not improve this feature-based task under the implemented protocol. However, RBF similarity may benefit strongly when augmented variants of the same clip cross partitions. The near-perfect SVM score is consequently not evidence of equivalent performance on new speakers or recordings; source-grouped evaluation is required before generalizing this result.

3.3 Interpretation of the CNN Failure Mode

The CNN's near-perfect training accuracy contrasted with a holdout collapse toward “bajingan” and “bangsat.” Plausible contributors are its 2.72 million parameters relative to 151 independent clips, the large flatten-to-dense layer (96.4% of parameters), class imbalance, and no pre-trained embeddings. Without ablation, these are hypotheses rather than demonstrated causes (Zhang et al., 2021).

Of the 272 holdout samples, the CNN labeled 244 as bajingan and 28 as bangsat, predicting none of the other six classes. Its unstable convergence means the tested configuration is unsuitable for reliable keyword classification; this does not establish a limitation of CNNs generally.

3.4 Class-Wise DNN Performance

Table 3 shows the DNN's highest class F1 for tolol (0.9429), followed by jancuk (0.8889) and bajingan (0.8718). Tai had the lowest F1 (0.7391): recall was 0.8500, but precision only 0.6538. Tai also had only 11 original sources and 20 holdout samples, versus 36 holdout samples for each other class.

Table 3. DNN holdout performance by class

Class	Precision	Recall	F1-score	Support
anjing	0.8788	0.8056	0.8406	36
babi	0.7692	0.8333	0.8000	36
bajingan	0.8095	0.9444	0.8718	36
bangsat	0.8214	0.6389	0.7188	36
goblok	0.8824	0.8333	0.8571	36
jancuk	0.8889	0.8889	0.8889	36
tai	0.6538	0.8500	0.7391	20
tolol	0.9706	0.9167	0.9429	36

These differences may reflect source diversity, imbalance, or recording conditions; no causal phonetic interpretation is justified.

3.5 Confusion-Matrix Analysis

Figure 4(a) shows dispersed DNN errors, especially bangsat confused with goblok or tai. In Figure 4(b), CNN predictions collapse to 244 bajingan and 28 bangsat samples. Although bajingan recall reached 100%, its low precision explains the very low macro F1.

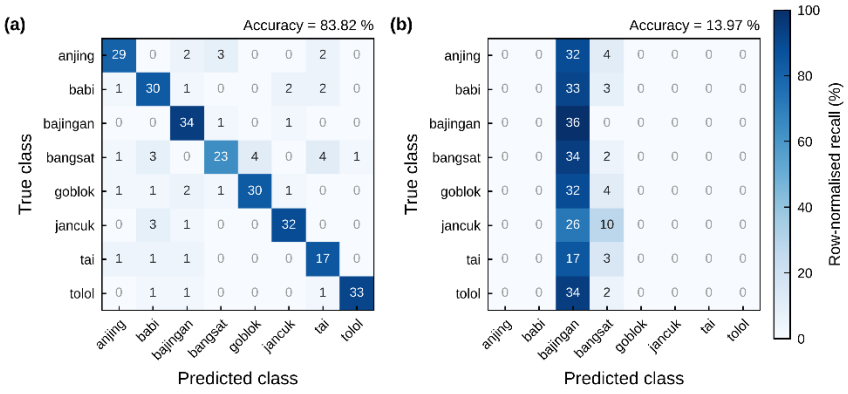


Figure 4. Holdout confusion matrices of (a) the DNN and (b) the CNN; cell shading is row-normalized recall and numbers are sample counts

3.6 Relation to Previous Work and Implications for Indonesian Audio Research

Compared with Indonesian text-based work (Salim & Suhartono, 2021; Ibrohim & Budi, 2023), this study tests spoken lexical identity without transcription. Unlike binary text-image Shorts classification (Pratama et al., 2026) or audio-text toxicity fusion (Supriyanto et al., 2026), its eight-class audio-only task cannot infer abusive intent. Its scientific contribution is evidence about a possible acoustic component of moderation, not a complete detector. Differences in labels and splits also prohibit direct accuracy comparisons with those studies.

The SVM's feature-matched performance shows why conventional baselines remain important for small audio datasets; the DNN's stable folds and CNN's collapse further motivate reporting variance and class-wise errors, not just mean accuracy. These observations refine rather than contradict successful CNN findings on other sound and speech corpora (Piczak, 2015; Sercu et al., 2016). Because representations differ, no causal architecture claim follows. Future evaluation should split by source before augmentation, repeat independent seeds, test smaller or pre-trained spectrogram networks, and assess untrimmed Shorts.

3.7 Limitations and External Validity

External validity is limited by 151 original source clips, artificial augmentation, uneven class counts, single-rater labeling, and isolation of eight explicit keywords. The two-second segments

do not capture unknown keyword locations, overlapping speech, or conversational intention in full videos. Only one conventional baseline was tested; no Random Forest or pre-trained YAMNet/VGGish comparison was performed.

Most critically, related augmented segments can cross the current train–validation/test boundaries. A stronger protocol would group samples by original recording before splitting, augment training folds only, repeat multiple seeds, and report uncertainty. All current scores, especially the SVM's, may be optimistic and should not be treated as deployment benchmarks.

4. Conclusions

This study compared feature-based DNN, spectrogram-based CNN, and feature-matched RBF-SVM for eight Indonesian abusive keywords isolated from YouTube Shorts audio (151 original source clips; 906 samples after augmentation). DNN achieved 87.08% ± 2.12% cross-validation accuracy and 83.82% holdout accuracy (macro F1: 83.24%). CNN achieved 81.33% ± 30.48% and 13.97%, respectively (macro F1: 4.00%), with six classes receiving no holdout predictions. SVM obtained 98.50% ± 1.20% and 99.17% (macro F1: 99.17%). The contribution is an audio-only Indonesian lexical case study that combines a conventional baseline with analysis of class collapse and fold stability. The SVM outperformed the deeper DNN on identical engineered inputs, but all estimates may be inflated because augmentation preceded sample-level splitting. These results classify lexical identity, not abusive intent, and do not establish a universal model ranking. Source-grouped validation, independent annotation, broader baselines, and full-video evaluation are priorities for future work.

References

Chang, C.-C., & Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology, 2*(3), 27. https://doi.org/10.1145/1961189.1961199

Chollet, F., et al. (2015). *Keras* [Computer software]. https://keras.io

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning, 20*(3), 273–297. https://doi.org/10.1007/BF00994018

Davis, S. B., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 28*(4), 357–366. https://doi.org/10.1109/TASSP.1980.1163420

Giannakopoulos, T., & Pikrakis, A. (2014). *Introduction to audio analysis: A MATLAB approach*. Academic Press. https://doi.org/10.1016/C2012-0-03524-7

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)* (Vol. 15, pp. 315–323). PMLR.

Ibrohim, M. O., & Budi, I. (2023). Hate speech and abusive language detection in Indonesian social media: Progress and challenges. *Heliyon, 9*(8), e18647. https://doi.org/10.1016/j.heliyon.2023.e18647

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)* (Vol. 37, pp. 448–456). PMLR.

Jenkins, W. K., Jones, D. L., & Hunsinger, B. J. (2002). Discrete-time signal processing. In *Reference data for engineers* (9th ed.). Newnes. https://doi.org/10.1016/B978-075067291-7/50030-3

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1412.6980

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436–444. https://doi.org/10.1038/nature14539

McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and music signal analysis in Python. In *Proceedings of the 14th Python in Science Conference* (pp. 18–25). https://doi.org/10.25080/Majora-7b98e3ed-003

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Piczak, K. J. (2015). Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. https://doi.org/10.1109/MLSP.2015.7324337

Pratama, E. F., Setyawan, R. A., & Bororing, J. E. (2026). DNN vs CNN for YouTube Shorts abusive word classification. *Journal of System and Computer Engineering, 7*(3). https://doi.org/10.61628/jsce.v7i3.2759

Purwins, H., Li, B., Virtanen, T., Schlüter, J., Chang, S.-Y., & Sainath, T. (2019). Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing, 13*(2), 206–219. https://doi.org/10.1109/JSTSP.2019.2908700

Salamon, J., & Bello, J. P. (2017). Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters, 24*(3), 279–283. https://doi.org/10.1109/LSP.2017.2657381

Salim, C. E. R., & Suhartono, D. (2021). Long short-term memory for hate speech and abusive language detection on Indonesian YouTube comment section. In *Proceedings of the 11th International Workshop on Computer Science and Engineering (WCSE)*. https://doi.org/10.18178/wcse.2021.06.029

Sercu, T., Puhrsch, C., Kingsbury, B., & LeCun, Y. (2016). Very deep multilingual convolutional neural networks for LVCSR. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 4955–4959). https://doi.org/10.1109/ICASSP.2016.7472620

Shi, T., et al. (2023). Recent advances in plant disease severity assessment using convolutional neural networks. *Scientific Reports, 13*(1). https://doi.org/10.1038/s41598-023-29230-7

Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management, 45*(4), 427–437. https://doi.org/10.1016/j.ipm.2009.03.002

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*(56), 1929–1958.

Supriyanto, H. A., Setyawan, R. A., & Bororing, J. E. (2026). Automatic detection of toxic content in short videos using deep learning-based text and audio feature integration. *Journal of Deep Learning, Computer Vision and Digital Image Processing, 4*(2), 221–235. https://doi.org/10.61255/decoding.v4i2.1540

Waseem, Z. (2016). Are you a racist or am I seeing things? Annotator influence on hate speech detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science* (pp. 138–142). https://doi.org/10.18653/v1/W16-5618

Wen, J., et al. (2019). How serious is data leakage in deep learning studies on Alzheimer's disease classification? *Annual Meeting of the Organization for Human Brain Mapping (OHBM)*. https://hal.science/hal-02105133

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM, 64*(3), 107–115. https://doi.org/10.1145/3446776