Implementation of Singular Value Decomposition for Badminton Stroke Training

Geraldo Artemius - 13524005
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
Email : [1]*13524005@std.stei.itb.ac.id*, [2]*geraldoartemius2005@gmail.com*

***Abstract*—Usually badminton stroke quality is evaluated subjectively based on visual observation and experience. This paper presents an implementation of Singular Value Decomposition (SVD) to analyze badminton stroke quality using signals generated during racket and shuttlecock impact. Each audio signal was preprocessed and transformed into a time-frequency in the form of spectogram matrix. SVD is implemented to decompose the matrix into finding dominant and non-dominant components. Then, stroke quality will be quantified by analyzing the singular values obtained. A perfect hit puts most of its sound energy into one primary signal. Because of this, clean strokes have a higher ratio of dominant values, while messy hits spread their energy across many smaller and noisier signals. The results can be an effective method for analyzing badminton stroke quality in training.**

Keywords—Spectogram Matrix, Singular Value Decomposition, Badminton, Stroke Quality, Signal Processing

I. INTRODUCTION

Badminton is a world-wide known sport and is considered one of the most fast-paced sports, requiring proper technique and consistent stroke execution. The quality of a badminton stroke significantly affects shuttlecock speed, control, and accuracy. In training, stroke quality is commonly evaluated based on the visual observation and sound perception by coaches. Although such evaluations are effective, they remain subjective and may vary depending on individual experience.

Signal processing provides an alternative approach for objective sports analysis through the use of acoustic data. In badminton, the sound created when a racket strikes a shuttlecock reflects the quality of the hit. Clean strokes tend to produce clear and consistent sounds, whereas off-center or imperfect strokes generate irregular vibrations and noise.

Singular Value Decomposition (SVD) is a technique in linear algebra that has been widely applied in signal processing, data compression, and pattern recognition. SVD decomposes a matrix into dominant and non-dominant components, allowing important signal structures to be separated from noise. This makes SVD very suitable for analyzing time-frequency representations of acoustic signals.

This paper aims to implement SVD to analyze badminton stroke quality based on acoustic signals. The objective of this study is to investigate whether the distribution of singular values can be used as a quantitative indicator of stroke quality. The contribution of this work lies in the application of a simple SVD-based approach using audio data from public internet.

II. LITERATURE REVIEW

A. Vectors and Basic Operations

Vector is an object that has both magnitude and direction, and is commonly represented as an ordered list of real numbers. A vector in $\mathbb{R}^n$ is written as

$$\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}.$$

Vectors support several operations. Given two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$, vector addition is defined as

$$\mathbf{u} + \mathbf{v} = \begin{bmatrix} u_1 + v_1 \\ u_2 + v_2 \\ \vdots \\ u_n + v_n \end{bmatrix},$$

and scalar multiplication by a real number c is defined as

$$c\mathbf{v} = \begin{bmatrix} cv_1 \\ cv_2 \\ \vdots \\ cv_n \end{bmatrix}.$$

The magnitude (or norm) of a vector $\mathbf{v}$ is given by

$$\|\mathbf{v}\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}.$$

These operations form the basis for vector projections and transformations used in eigenvalue and SVD analysis.

B. Matrices and Matrix Operations

A matrix is a list of array of real numbers arranged in rows and columns. An $m \times n$ matrix A is written as

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

Matrix addition and scalar multiplication are defined element. Given matrices A and B of the same dimension and a scalar c,

$$A + B = [a_{ij} + b_{ij}], \quad cA = [ca_{ij}].$$

Matrix multiplication is defined when the number of columns of A equals the number of rows of B. For $A \in \mathbb{R}^{m\times p}$ and $B \in \mathbb{R}^{p\times n}$, the product $C = AB$ is given by

$$c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}.$$

The transpose of a matrix A is denoted by A^T and is obtained by interchanging rows and columns. These matrix operations are essential for defining eigenvalue problems and singular value decomposition.

C. Euclidean Distance

Euclidean distance is used for measuring the distance between two points in a Euclidean space. In an n-dimensional real vector space $\mathbb{R}^n$, the Euclidean distance between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ is defined as

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

In this study, Euclidean distance is used to measure the similarity between the feature vector extracted from an unknown badminton stroke and the centroid of each stroke quality class. A smaller distance indicates that the stroke shares more similar structural characteristics with the corresponding class.

D. Eigenvalue and Eigenvector

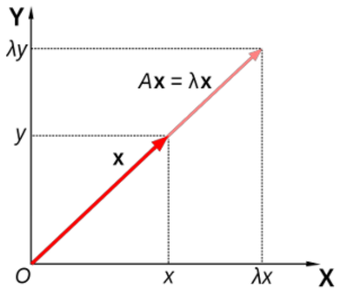


Fig. 1. Representation of Eigenvector and Eigenvalue. source

Eigenvalues and eigenvectors are fundamental concepts before jumping into SVD. Let $A \in \mathbb{R}^{n\times n}$ be a square matrix. A non-zero vector $x \in \mathbb{R}^n$ is called an eigenvector of A if it satisfies

$$Ax = \lambda x,$$

where λ is a scalar known as the eigenvalue associated with x.

The eigenvalues of matrix A are obtained by arranging the equation into

$$(A - \lambda I)x = 0,$$

where I is the identity matrix. Non-trivial solutions exist if and only if $\det(A - \lambda I) = 0$. This equation is called the characteristic equation of A, and its roots correspond to the eigenvalues. The eigenvectors are then obtained by substituting each eigenvalue into $(A - \lambda I)x = 0$ and solving the resulting system using Gaussian elimination.

E. Singular Value Decomposition

Singular Value Decomposition (SVD) is a matrix factorization technique that can be applied to any real matrix $A \in \mathbb{R}^{m\times n}$, including non-square matrices. The SVD of matrix A is defined as

$$A = U\Sigma V^T,$$

where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\Sigma \in \mathbb{R}^{m\times n}$ is a diagonal matrix containing the singular values of A.

The singular values σ_i are obtained from the eigenvalues of $A^T A$ as

$$\sigma_i = \sqrt{\lambda_i}, \quad \lambda_i \geq 0,$$

where λ_i are the eigenvalues of $A^T A$ sorted in descending order. The diagonal elements of Σ are arranged such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k \geq 0,$$

with $k = \text{rank}(A)$.

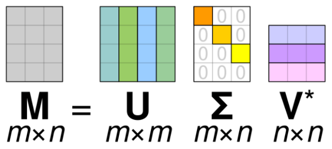


Fig. 2. Singular Value Decomposition. source

The columns of U, denoted as $\{u_1, u_2, \ldots, u_k\}$, are called left singular vectors, while the columns of V, denoted as $\{v_1, v_2, \ldots, v_k\}$, are called right singular vectors. These vectors form orthonormal bases for the column space and row space of matrix A.

SVD enables the separation of dominant and non-dominant components of a matrix, making it particularly useful for signal processing applications such as noise reduction, feature extraction, and others. In this study, SVD is applied to time-frequency representations of acoustic signals to analyze badminton stroke quality. Key properties of SVD include :

1) Orthogonality and Projection: One important property of SVD is the orthogonality of matrices U and V. A matrix Q is said to be orthogonal if

$$Q^T Q = QQ^T = I.$$

This property ensures that the singular vectors form an orthonormal basis. As a result, any vector $x \in \mathbb{R}^n$ can be projected onto the subspace spanned by the right singular vectors as

$$x = \sum_{i=1}^{n} (v_i^T x) v_i,$$

where v_i are the right singular vectors forming an orthonormal basis.

2) Low-Rank Approximation: One of the most important applications of SVD is low-rank matrix approximation. Given the SVD of matrix A,

$$A = \sum_{i=1}^{k} \sigma_i u_i v_i^T,$$

a rank-r approximation of A can be obtained by retaining only the first r dominant singular values,

$$A_r = \sum_{i=1}^{r} \sigma_i u_i v_i^T, \quad r < k.$$

This approximation minimizes the reconstruction error in the Frobenius norm sense,

$$\|A - A_r\|_F = \min_{\text{rank}(B)=r} \|A - B\|_F,$$

making truncated SVD an optimal method for noise reduction and dimensionality reduction.

3) Energy Interpretation of Singular Values: The magnitude of each singular value σ_i reflects the energy contribution of the corresponding singular mode. The total energy of matrix A can be expressed as

$$\|A\|_F^2 = \sum_{i=1}^{k} \sigma_i^2.$$

In practical applications, dominant singular values capture most of the signal energy, while smaller singular values often correspond to noise or minor variations. Therefore, the ratio between dominant singular values and the total energy can be used as an indicator of signal quality.

F. Frobenius Norm

The Frobenius norm is a widely used matrix norm that generalizes the Euclidean norm from vectors to matrices. For a matrix $A \in \mathbb{R}^{m\times n}$, the Frobenius norm is defined as

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2}.$$

Frobenius norm can also be expressed using the trace operator as

$$\|A\|_F = \sqrt{\text{trace}(A^T A)}.$$

An important property of the Frobenius norm is its invariance under orthogonal transformations. If U and V are orthogonal matrices, then

$$\|A\|_F = \|UAV^T\|_F.$$

When the singular value decomposition of A is given by $A = U\Sigma V^T$, the Frobenius norm can be expressed directly in terms of its singular values as

$$\|A\|_F^2 = \sum_{i=1}^{k} \sigma_i^2,$$

where σ_i are the singular values of A and $k = \text{rank}(A)$.

G. Audio Signal Representation

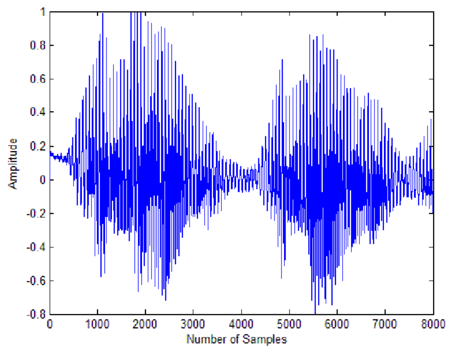


Fig. 3. Audio Signal Graphical Representation. source

A badminton stroke produces an acoustic signal that can be represented as a discrete-time signal

$$x[n], \quad n = 1, 2, \ldots, N,$$

where $x[n]$ denotes the audio amplitude at time index n. This signal can be viewed as a vector in $\mathbb{R}^N$.

However, a time domain representation lacks frequency information, which is crucial for identifying the unique vibration patterns of the impact. Therefore, the signal will be transformed into a time-frequency matrix representation using the Short-Time Fourier Transform (STFT).

The STFT produces a matrix

$$A \in \mathbb{R}^{F\times T},$$

where F denotes the number of frequency bins and T denotes the number of time frames. Each element a_{ij} represents the magnitude of frequency component i at time frame j.

H. Short-Time Fourier Transform

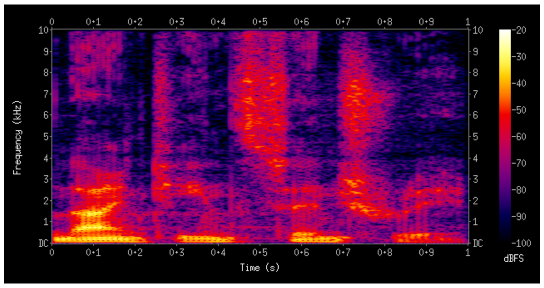


Fig. 4. Short-Time Fourier Transform. source

The Short-Time Fourier Transform (STFT) is used to analyze the frequency of a signal over time. Given an audio signal $x[n]$, the STFT is defined as

$$X(m,k) = \sum_{n=-\infty}^{\infty} x[n]\, w[n-m]\, e^{-j2\pi kn/N},$$

where $w[n]$ is a window function centered at time index m, k denotes the frequency index, and N is the window length.

The STFT divides the signal into overlapping frames using a sliding window and computes the Fourier transform for each frame. As a result, the signal is represented in the time-frequency domain. This time-frequency matrix will be the input matrix for Singular Value Decomposition.

III. METHODOLOGY

A. System Overview, Programming Language, and Tools

The implementation of this project is developed using the Python programming language. Python is chosen due to its simplicity, readability, and collection of libraries that support numerical computation and signal processing.

Several open-source libraries are used in this project. The NumPy library is used for efficient numerical operations, matrix manipulation, and the implementation of Singular Value Decomposition. Librosa is employed for audio signal processing tasks, including audio loading, resampling, time-frequency analysis, and feature extraction. Additionally, the SoundFile library is used particularly for reading waveform data in .wav format.

The system is designed to transforms raw sound recordings into numerical representations. Each audio file is first processed to extract a normalized impact segment, followed by a time-frequency representation using the Short-Time Fourier Transform (STFT). The resulting spectrogram is then decomposed using Singular Value Decomposition to obtain features from the signal.

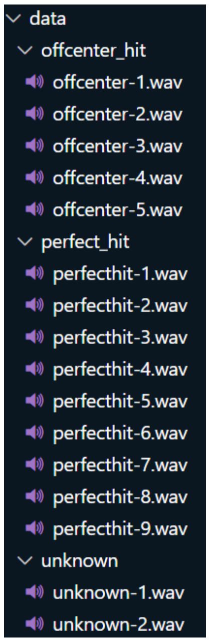
The final output of the system consists of structured feature values and classification results stored in comma-separated value (CSV) files, enabling quantitative comparison between perfect hits and off-center hits.

B. Data Collection

In this section, we focus on finding the right and specific sound of a perfect hit in badminton stroke. The audio data used in this paper were collected from publicly available online source, primarily youtube. The recording consists of acoustic signals from badminton racket and shuttlecock impact sounds. Each audio only consist of single stroke event.

The audios were divided into two groups, which are perfect hits and off-center hits. Perfect hits refer to strokes in which the shuttlecock is struct near the sweet spot of the racket, they tend to produce clear and consistent sound. Off-center hits refer to strokes with irregular impact, producing a much more noisier and less stable acoustic characteristics. Due to the lack of off-center records in public media, off-center samples are identified based on auditory characteristics such as reduced loudness consistency, increased noise components, and irregular frequency content.

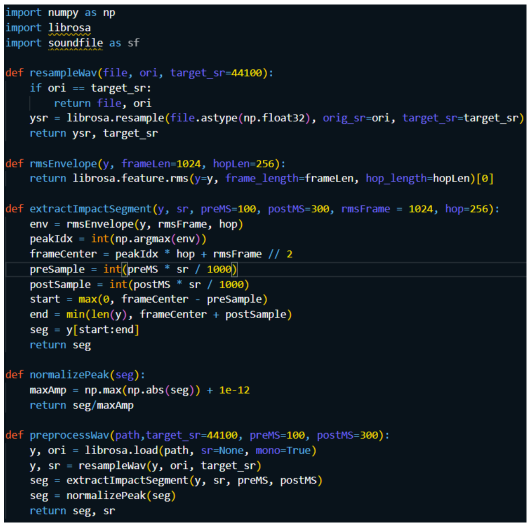
The collected audios are organized into two separate directories, namely perfect_hit/ dan offcenter_hit/, and unknown/, as illustrated in *Figure 5*. This directories are implemented in the source code.



```
data
  offcenter_hit
    offcenter-1.wav
    offcenter-2.wav
    offcenter-3.wav
    offcenter-4.wav
    offcenter-5.wav
  perfect_hit
    perfecthit-1.wav
    perfecthit-2.wav
    perfecthit-3.wav
    perfecthit-4.wav
    perfecthit-5.wav
    perfecthit-6.wav
    perfecthit-7.wav
    perfecthit-8.wav
    perfecthit-9.wav
  unknown
    unknown-1.wav
    unknown-2.wav
```

Fig. 5. Perfect Hit, Off-Center, and Unknown Audio Sounds. source : Writer's own code

C. Preprocessing



```python
import numpy as np
import librosa
import soundfile as sf

def resampleWav(file, ori, target_sr=44100):
    if ori == target_sr:
        return file, ori
    ysr = librosa.resample(file.astype(np.float32), orig_sr=ori, target_sr=target_sr)
    return ysr, target_sr

def rmsEnvelope(y, frameLen=1024, hopLen=256):
    return librosa.feature.rms(y=y, frame_length=frameLen, hop_length=hopLen)[0]

def extractImpactSegment(y, sr, preMS=100, postMS=300, rmsFrame = 1024, hop=256):
    env = rmsEnvelope(y, rmsFrame, hop)
    peakIdx = int(np.argmax(env))
    frameCenter = peakIdx * hop + rmsFrame // 2
    preSample = int(preMS * sr / 1000)
    postSample = int(postMS * sr / 1000)
    start = max(0, frameCenter - preSample)
    end = min(len(y), frameCenter + postSample)
    seg = y[start:end]
    return seg

def normalizePeak(seg):
    maxAmp = np.max(np.abs(seg)) + 1e-12
    return seg/maxAmp

def preprocessWav(path,target_sr=44100, preMS=100, postMS=300):
    y, ori = librosa.load(path, sr=None, mono=True)
    y, sr = resampleWav(y, ori, target_sr)
    seg = extractImpactSegment(y, sr, preMS, postMS)
    seg = normalizePeak(seg)
    return seg, sr
```

Fig. 6. Preprocessing Source Code. source : Writer's own code

Each audio signal is converted into a monophonic waveform and resampled to a uniform sampling rate of 44.1 kHz for consistency. To focus only on the impact, the author extract (cut) a small part of the audio around the peak RMS energy of the signal.

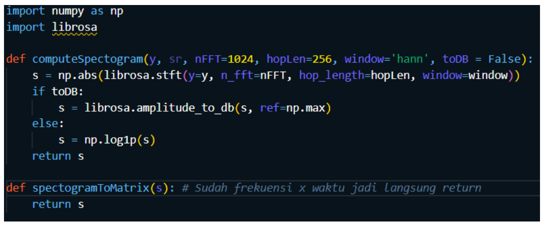
Let $y[n]$ denote the audio waveform. The RMS envelope is defined as

$$\mathrm{RMS}(k) = \sqrt{\frac{1}{N}\sum_{n=0}^{N-1} y^2[n+kH]},$$

where N is the frame length and H is the hop size.

A fixed-length segment centered around the maximum RMS value is extracted and normalized to unit peak amplitude.

D. Time-Frequency Representation



```
import numpy as np
import librosa

def computeSpectogram(y, sr, nFFT=1024, hopLen=256, window='hann', toDB = False):
    s = np.abs(librosa.stft(y=y, n_fft=nFFT, hop_length=hopLen, window=window))
    if toDB:
        s = librosa.amplitude_to_db(s, ref=np.max)
    else:
        s = np.log1p(s)
    return s

def spectogramToMatrix(s): # Sudah frekuensi x waktu jadi langsung return
    return s
```

Fig. 7. STFT implementation. source : Writer's own code

To capture the spectral characteristics of the impact sound, the Short-Time Fourier Transform (STFT) is applied to each segment that were preprocessed.

The STFT of signal $y[n]$ is given by

$$X(f,t) = \sum_n y[n]w[n - tH]e^{-j2\pi fn}, \tag{1}$$

where $w[n]$ is a Hann window and H is the hop length.

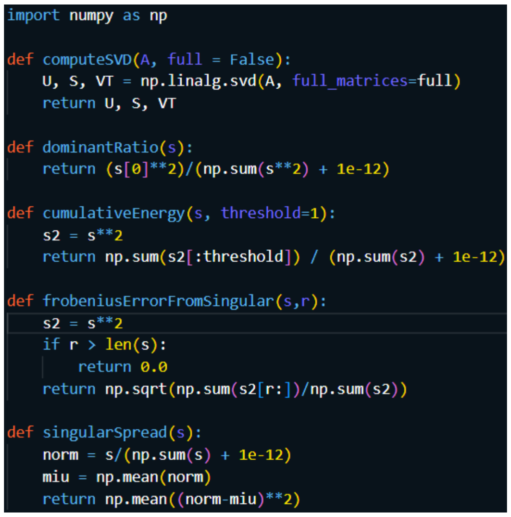
The magnitude spectrogram $S = |X|$ is further compressed using a logarithmic mapping:

$$S = \log(1 + |X|). \tag{2}$$

The result will be the matrix representation of the audio signal.

In this implementation, a Hann window is used with an FFT size of 1024 samples and a hop length of 256 samples. These parameters provide a balance between time and frequency resolution, which is suitable for capturing the characteristics of racket and shuttlecock impacts.

E. Singular Value Decomposition and Feature Extraction



```
import numpy as np

def computeSVD(A, full = False):
    U, S, VT = np.linalg.svd(A, full_matrices=full)
    return U, S, VT

def dominantRatio(s):
    return (s[0]**2)/(np.sum(s**2) + 1e-12)

def cumulativeEnergy(s, threshold=1):
    s2 = s**2
    return np.sum(s2[:threshold]) / (np.sum(s2) + 1e-12)

def frobeniusErrorFromSingular(s,r):
    s2 = s**2
    if r > len(s):
        return 0.0
    return np.sqrt(np.sum(s2[r:])/np.sum(s2))

def singularSpread(s):
    norm = s/(np.sum(s) + 1e-12)
    miu = np.mean(norm)
    return np.mean((norm-miu)**2)
```

Fig. 8. Singular Value Decomposition and Feature Extraction. source : Writer's own code

After extracting singular values from the spectrogram matrix, several features are computed to characterize the structure of the signal. These features are designed to quantify how concentrated or dispersed the energy of the impact sound is in the singular value domain.

Let σ_i denote the i-th singular value of the spectrogram matrix. The dominant ratio is defined as

$$D = \frac{\sigma_1}{\sum_i \sigma_i}, \tag{3}$$

which measures the dominance of the strongest singular component.

The cumulative energy ratio using the first r singular values is defined as

$$E_r = \frac{\sum_{i=1}^{r} \sigma_i^2}{\sum_i \sigma_i^2}. \tag{4}$$

In this study, $r = 1$ and $r = 3$ are used to analyze low-rank energy concentration.

To evaluate the approximation quality of a low-rank representation, the Frobenius reconstruction error is computed as

$$\epsilon_r = \sqrt{\sum_{i=r+1}^{k} \sigma_i^2}, \tag{5}$$

where k is the number of singular values.

Finally, the spread of singular values is measured by computing the variance of normalized singular values,

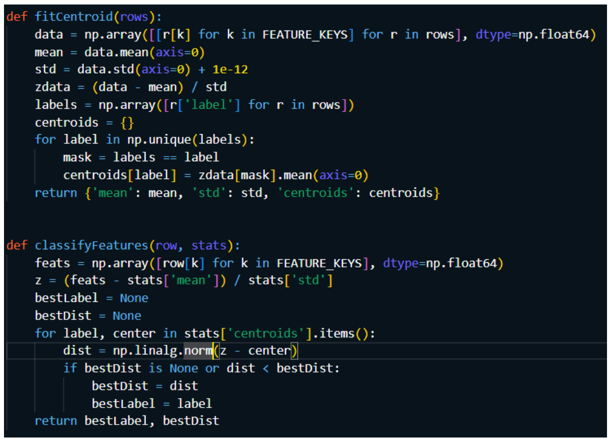
$$S = \frac{1}{k}\sum_{i=1}^{k} (\tilde{\sigma}_i - \mu)^2, \tag{6}$$

where $\tilde{\sigma}_i = \sigma_i / \sum_j \sigma_j$ and μ is their mean.

These features are computed for each audio sample and stored in a structured CSV file for further analysis.

F. Stroke Classification

To make an automatic identification of a stroke quality, feature vectors extracted from labeled samples (perfect and off-center hits) are first normalized using z-score normalization.



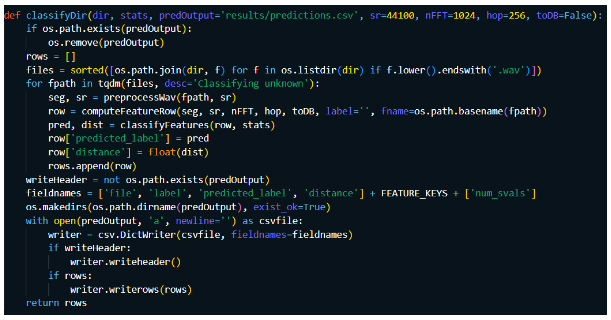
```
def fitCentroid(rows):
    data = np.array([[r[k] for k in FEATURE_KEYS] for r in rows], dtype=np.float64)
    mean = data.mean(axis=0)
    std = data.std(axis=0) + 1e-12
    zdata = (data - mean) / std
    labels = np.array([r['label'] for r in rows])
    centroids = {}
    for label in np.unique(labels):
        mask = labels == label
        centroids[label] = zdata[mask].mean(axis=0)
    return {'mean': mean, 'std': std, 'centroids': centroids}


def classifyFeatures(row, stats):
    feats = np.array([row[k] for k in FEATURE_KEYS], dtype=np.float64)
    z = (feats - stats['mean']) / stats['std']
    bestLabel = None
    bestDist = None
    for label, center in stats['centroids'].items():
        dist = np.linalg.norm(z - center)
        if bestDist is None or dist < bestDist:
            bestDist = dist
            bestLabel = label
    return bestLabel, bestDist
```

Fig. 9. Features Classification. source : Writer's own code

Let $\mathbf{x}$ be a feature vector and μ, σ be the mean and standard deviation computed from the training data. The normalized feature vector is given by

$$\mathbf{z} = \frac{\mathbf{x} - \mu}{\sigma}.$$



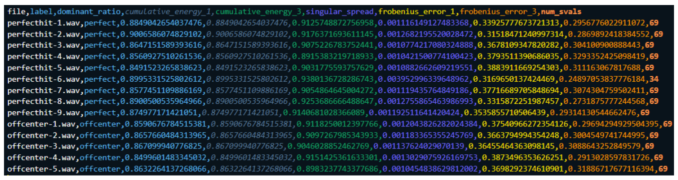
```python
def classifyDir(dir, stats, predOutput='results/predictions.csv', sr=44100, nFFT=1024, hop=256, toDB=False):
    if os.path.exists(predOutput):
        os.remove(predOutput)
    rows = []
    files = sorted([os.path.join(dir, f) for f in os.listdir(dir) if f.lower().endswith('.wav')])
    for fpath in tqdm(files, desc='Classifying unknown'):
        seg, sr = preprocessWav(fpath, sr)
        row = computeFeatureRow(seg, sr, nFFT, hop, toDB, label='', fname=os.path.basename(fpath))
        pred, dist = classifyFeatures(row, stats)
        row['predicted_label'] = pred
        row['distance'] = float(dist)
        rows.append(row)
    writeHeader = not os.path.exists(predOutput)
    fieldnames = ['file', 'label', 'predicted_label', 'distance'] + FEATURE_KEYS + ['num_svals']
    os.makedirs(os.path.dirname(predOutput), exist_ok=True)
    with open(predOutput, 'a', newline='') as csvfile:
        writer = csv.DictWriter(csvfile, fieldnames=fieldnames)
        if writeHeader:
            writer.writeheader()
        if rows:
            writer.writerows(rows)
    return rows
```

Fig. 10. Output Classification Implementation

For each class, a centroid is computed as the mean of normalized feature vectors for the class. Given an unknown sample, its normalized feature vector is compared to each class centroid using Euclidean Distance.

IV. RESULTS AND DISCUSSION

A. Perfect and Off-Center Hits Characteristics



```
file,label,dominant_ratio,cumulative_energy_1,cumulative_energy_3,singular_spread,frobenius_error_1,frobenius_error_3,num_svals
perfecthit-1.wav,perfect,0.8849042654037476,0.8849042654037476,0.9125748872756958,0.0011161491274833368,0.3392577673721313,0.295677602291107 2,69
perfecthit-2.wav,perfect,0.9006586074829102,0.9006586074829102,0.9176371693611145,0.0012682195520028472,0.3151847124099731 4,0.2869892418384552,69
perfecthit-3.wav,perfect,0.8647151589393616,0.8647151589393616,0.9075226783752441,0.0010774217080324888,0.3678109347820282,0.304100900888443,69
perfecthit-4.wav,perfect,0.8560927510261536,0.8560927510261536,0.8915383219718933,0.0010421500774100423,0.379351139068603 5,0.3293352425098419,69
perfecthit-5.wav,perfect,0.8491523265838623,0.8491523265838623,0.9031775593757629,0.0010882662609219551,0.3883911669254303,0.311163067817688,69
perfecthit-6.wav,perfect,0.8995331525802612,0.8995331525802612,0.9380136728286743,0.0039529963396489 62,0.316965013742446 9,0.2489705383777618 4,34
perfecthit-7.wav,perfect,0.857745110988 6169,0.8577451109886169,0.9054864645004272,0.0011194357648491 86,0.3771668970584869 4,0.3074304759502411,69
perfecthit-8.wav,perfect,0.8900500535964966,0.8900500535964966,0.9253686666488647,0.0012755865463986993,0.3315872251987457,0.2731875777244568,69
perfecthit-9.wav,perfect,0.87497717142105 1,0.87497717142105 1,0.9140681028366089,0.0011925116414204 24,0.353585571050643 9,0.2931413054466247 6,69
offcenter-1.wav,offcenter,0.8590676784515381,0.8590676784515381,0.9118250012397766,0.0012043826282024384,0.3754096627235412 6,0.2969429492950439 5,69
offcenter-2.wav,offcenter,0.8657660484313965,0.8657660484313965,0.9097267985343933,0.001183365355245769,0.3663794994354248,0.300454974174499 5,69
offcenter-3.wav,offcenter,0.867099940776825,0.867099940776825,0.9046028852462769,0.0011376240290701 39,0.3645546436309814 5,0.3088643252849579,69
offcenter-4.wav,offcenter,0.8499601483345032,0.8499601483345032,0.9151425361633301,0.0013029075926169753,0.3873496353626251,0.2913028597831726,69
offcenter-5.wav,offcenter,0.8632264137268066,0.8632264137268066,0.8983237743377686,0.0010454838629812002,0.369829237461090 1,0.3188671767711639 4,69
```

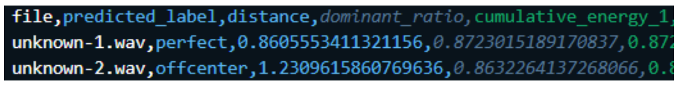
Fig. 11. Perfect and Off-Center Audio Result. source : Writer's own code

From the results in metrics.csv, perfect hits samples has a higher dominant singular value ratio, which indicates a large proportion of signal energy is extracted on the first singular value. Even though there's not much difference between the two categories, due to the lack and hard collection of data audios. But we can conclude off-center hit tend to have lower dominant ratios and higher Frobenius reconstruction errors.

Dominant Ratios Interval Classification :

- Perfect hit : $85 \leq \text{Dominant Ratio} \leq 90$.
- Off-Center hit : $85 \leq \text{Dominant Ratio} \leq 87$.

B. Unknown Audio Classification



```
file,predicted_label,distance,dominant_ratio,cumulative_energy_1
unknown-1.wav,perfect,0.8605553411321156,0.8723015189170837,0.87
unknown-2.wav,offcenter,1.2309615860769636,0.8632264137268066,0.8
```

Fig. 12. Unknown Audio Classification Result. source : Writer's own code

The system is able to correctly analyze between perfect hits and off-center hits for previously unseen/unclassified audio samples.

The file *unknown-1.wav* is classified as a perfect hit with a relatively small distance value. This prediction is consistent because the audio was sourced from a publicly available video containing a clean and perfect badminton stroke.

The file *unknown-2.wav* is classified as an off-center hit with a larger distance from the perfect hit centroid. This result is also correct because the audio corresponds to a racket frame impact, which produces a noisier and less stable acoustic response.

V. CONCLUSION

This paper shows that Singular Value Decomposition (SVD) can be used to analyze a badminton stroke quality based on the impact sound signals. By applying SVD to the spectogram of each audio sample, the difference between perfect hits and off-center hits can be seen from the distribution of singular values. Perfect hits tend to concentrate more energy in the dominant singular values than off-center hits. Although the audio samples are limited, the results indicate that SVD-based features can provide a simple and effective approach for badminton stroke quality analysis. This paper's purpose is to improve a person's badminton stroke from being inconsistent to being consistent in executing perfect hit.

VI. APPENDIX

GitHub : https://github.com/GeraldoA07/Badminton-Stroke-SVD-Decomposition

VII. ACKNOWLEDGMENT

The writer would like to express sincere gratitude to Dr. Ir. Rinaldi Munir, M.T., for his dedication in teaching *Linear Algebra and Geometri* for K-1 throughout the third semester. The writer also appreciates the support and encouragement from their family and friends during the completion of this paper. Lastly, the writer gives thanks to God for His grace, and strength throughout this journey. The writer also acknowledges the use of AI tools, which assisted in grammar correction, generating content ideas, and implementing source code.

REFERENCES

[1] R. Munir, "Aljabar Geometri 2025-2026," Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2025-2026/algeo25-26.htm. [Accessed: Dec. 23, 2025].

[2] F. Ruan, "Analisis Komponen Utama," 2025. Available: https://www.fanruan.com/id/glossary/analisis-data/Analisis-Komponen-Utama. [Accessed: Dec. 23, 2025].

[3] NumPy Developers, "NumPy,". Available: https://numpy.org/. [Accessed: Dec. 24, 2025].

[4] S. Sekhar Bhakta, "Audio classification using spectrograms," GeeksforGeeks, Jul. 23, 2025. Available: https://www.geeksforgeeks.org/nlp/audio-classification-using-spectrograms/. [Accessed: Dec. 24, 2025].

[5] E. W. Weisstein, "Frobenius Norm," Wolfram MathWorld. [Online]. Available: https://mathworld.wolfram.com/FrobeniusNorm.html. [Accessed: Dec. 23, 2025].

STATEMENT OF ORIGINALITY

I hereby declare that the paper I have written is my own original work, not an adaptation or translation of someone else's work, and not an act of plagiarism.

Bandung, 24 December 2025

[illegible]

Geraldo Artemius
13524005