Embedded Systems Entrepreneurship

Team: Arpit Baranwall, Ali Sarlak, Natik, Mohamadreza Sharifi

July 22, 2023

1 Introduction

Our research has identified a gap in the medical market pertaining to the availability of a device capable of quantifying the masticatory patterns exhibited by patients when consuming various types of food. This device would prove beneficial in conjunction with expert analysis , aiding in the diagnosis of potential anomalies in patients. The process of mastication in humans holds significant importance across multiple domains, particularly within dentistry and the digestive system. Proper chewing of food is crucial for optimal digestive functioning, as inadequate mastication can lead to complications within the digestive system and potentially contribute to related ailments. Additionally, through the examination of collected data, dentists can identify any irregularities in the jaws and teeth, which may be indicative of suboptimal chewing patterns.

In response to the identified demand for such a device, a novel pair of glasses has been developed (Figure 1), incorporating specialized sensors that possess a high sensitivity to resonance. These sensors enable the accurate capture of jaw movements during the process of mastication for various food types. Notably, these sensors have the capability to detect and record a wide range of jaw movements, encompassing activities such as speaking, chewing, and other facial muscle activities. Consequently, postprocessing of the collected data, specifically signal processing, is imperative to isolate and extract the relevant portion of the signal corresponding to the chewing of food. Furthermore, the project's objective is twofold: firstly, to identify the specific segment of the recorded signal that corresponds to the patient's chewing activity, and secondly , to discern the type of food being chewed based on the characteristic chewing patterns exhibited in the signal.

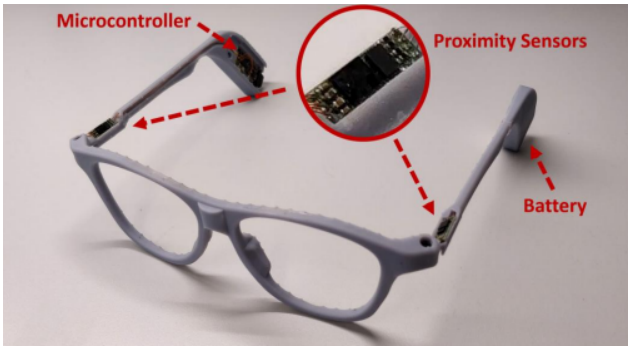


Figure 1: Vibration Glasses

In addressing the aforementioned issue, a previous group of researchers attempted to tackle food classification based on input signals by employing the Extended Kalman filter in conjunction with additional mounted sensors. However, this approach necessitated the placement of uncomfortable and non-portable Electromyography (EMG) sensors on the participant's face. Consequently, the current project aims to overcome these limitations by exclusively utilizing captured signals, eliminating the reliance on EMG data. Deep learning and machine learning techniques, which have demonstrated promising outcomes in various fields, including signal processing, will be employed for food classification based on the signals obtained from the participant's chewing activities.

2 Data analysis

Certainly, the quality and reliability of the data play a crucial role in determining the effectiveness of any model. Therefore, this section seeks to delve into the attributes of the previously recorded data and to validate the accuracy of the measurements and assertions made.

In order to provide the readers of this report with a comprehensive understanding of the data, it is imperative to elucidate the nature of the signals and the specific circumstances under which the data was collected.

The experimental cohort consisted of eleven participants who actively engaged in the study. Each participant wore the specialized glasses and consumed a set of five distinct food items, namely apple, beef jerky, baguette, cheddar, and chips. However, it is important to acknowledge that an uneven distribution of data was observed among the food types. Consequently, certain food items were not grasped by a few participants, resulting in a scarcity of data for those specific food categories in comparison to others.

Furthermore, separate signals corresponding to both the food types and individual participants were obtained and made available for analysis.

The data was acquired using a sampling rate of 24 kHz, resulting in the collection of 24,000 samples per second. However, it is important to note that the accuracy of the data acquisition was not consistent. As a result, the actual number of samples acquired per second ranged from 24,000 to 24,104, with slight variations observed in different cases. The following table provides an overview of the participants and the specific food items they consumed during the data collection process.

#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11
apple	beef jerky	beef jerky	chips	chips	apple	chips	apple	apple	baguette	apple
beef jerky	apple	chips	chips	chips	apple	chips	apple	chips	baguette	chips
apple		chips	chips	beef jerky	cheddar	chips		chips	baguette	beef jerky
		chips	chips	beef jerky	cheddar	chips		chips	apple	beef jerky
		beef jerky	chips	beef jerky		chips		chips	apple	apple
		beef jerky	chips	beef jerky		chips			apple	chips
						baguette			apple	chips
									chips	
									chips	
									chips	
									cheddar	
									cheddar	
									beef jerky	
									chips	
									chips	

As evident from the aforementioned table, it is apparent that there is an uneven distribution of data among the participants. This uneven distribution poses a challenge for many machine learning and deep learning models, as they often require a balanced dataset for optimal performance. However, it is worth noting that various approaches exist to mitigate this issue.

One possible approach is data augmentation, which involves artificially increasing the size of the dataset by applying transformations such as phase shift, scaling, or adding noise to the existing samples. This can help balance the dataset and provide additional variations for training the models.

Another approach is to employ techniques such as oversampling or undersampling. Oversampling involves replicating the minority class samples to match the number of majority class samples, while undersampling involves randomly removing samples from the majority class to match the number of minority class samples. Both techniques aim to achieve a more balanced dataset.

Additionally, ensemble methods, which combine predictions from multiple models, can be utilized to account for the class imbalance. By training several models on different subsets of the data or using different algorithms, ensemble methods can improve the overall performance and robustness of the models.

Overall, while dealing with an uneven distribution of data among participants presents a challenge, employing techniques such as data augmentation, oversampling, undersampling, or ensemble methods can help mitigate this problem and enhance the performance of machine learning and deep learning models.

2.1 Collected Signals

Prior to delving into other aspects, it is essential to focus on the characteristics of the signals themselves and the context in which the previous team collected the data. Each participant was seated in a comfortable chair and engaged in a series of activities, including responding to questions to capture muscle activity during speech, followed by the consumption of various food items. Subsequently, the recorded data was partitioned into multiple CSV files, with each file encompassing approximately 48 MB of data, corresponding to approximately 1,000,000 records. The number of CSV files ranged from 20 to 25 for each participant, depending on the number of foods they consumed. Within this extensive dataset, the region of interest lies in the portion pertaining to food chewing.

Additionally, another dataset was compiled specifically for food chewing, consisting of signals isolated for each food item and participant. This dataset serves as the training, validation, and testing sets for the subsequent models. Furthermore, a supplementary dataset comprising text files was included, containing annotations denoting the onset of chewing for each chewing sequence within the signal. Each line in the text files represents a time value in seconds, signifying the initiation of chewing as mentioned.

2.2 Verification

The initial task at hand involves confirming the presence of the chewing sequence for a specific participant within the complete signal. To accomplish this, the individual CSV files were combined to construct a cohesive and comprehensive dataset stored in memory. Subsequently, given that the recorded signals were susceptible to noise, a preprocessing technique was employed to enhance the signal quality and eliminate outliers. Among the various filters tested, the "hampel" filter was selected for its ability to effectively remove outliers and achieve signal smoothing. Notably, it is important to mention that certain filters, such as the median filter, proved unsuitable for this particular task as they exhibited a tendency to distort the input signal.

The Hampel filter is a statistical method commonly used for outlier detection and removal in signal processing. It operates by identifying data points that deviate significantly from the surrounding values and replaces them with more representative estimates. This filter considers the median absolute deviation (MAD) as a robust measure of dispersion to detect outliers. The MAD is calculated by taking the median of the absolute differences between each data point and the median of its neighboring values. The Hampel filter compares each data point to a threshold based on a specified multiple of the MAD. If the difference exceeds this

threshold, the data point is considered an outlier and replaced with a more suitable estimate, such as the median of the surrounding values.

On the other hand, the median filter is a widely used technique for signal smoothing and noise reduction. It operates by replacing each data point with the median value within a defined neighborhood window. The median is less sensitive to extreme values, making it effective in mitigating the impact of outliers and preserving the overall shape of the signal. However, it is worth noting that the median filter can also introduce certain **distortions** in the signal, particularly when applied to signals with abrupt changes or sharp features. These distortions may be undesirable in certain applications, such as the analysis of chewing sequences, where preserving the fine details of the signal is crucial.

For a comprehensive understanding of the input signal and the impact of noise, Figure 2 displays the plotted signal in its raw form. This visualization provides insights into the inherent characteristics and fluctuations within the signal. However, to mitigate the effects of noise and enhance the signal quality, preprocessing techniques were applied. Figure 3 presents the signal after undergoing the designated preprocessing steps, showcasing the result of noise reduction and smoothing. By comparing these two figures, one can gain a better appreciation of the improvements achieved through the preprocessing stage.

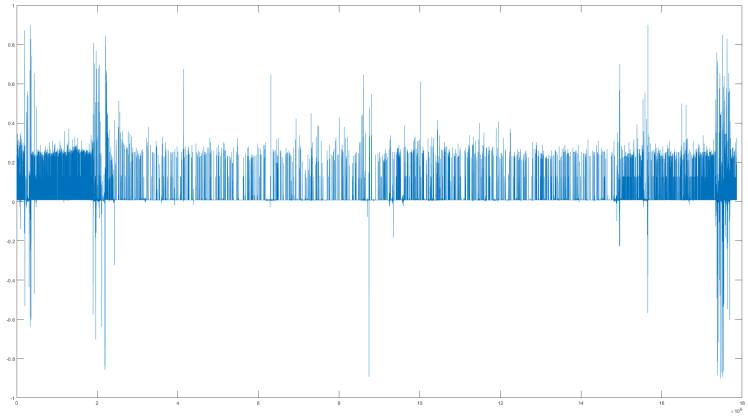


Figure 2: Raw signal

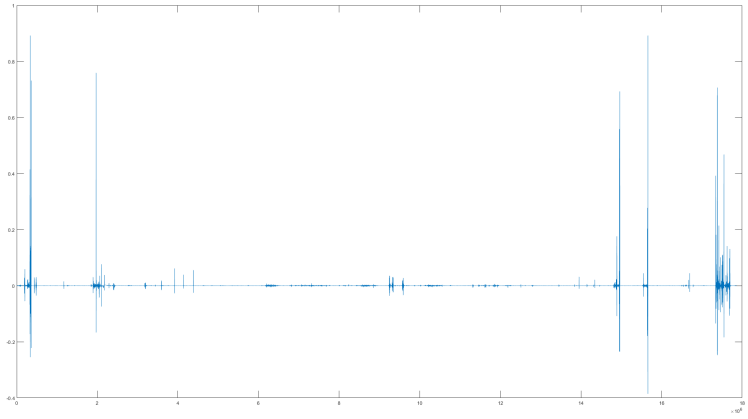
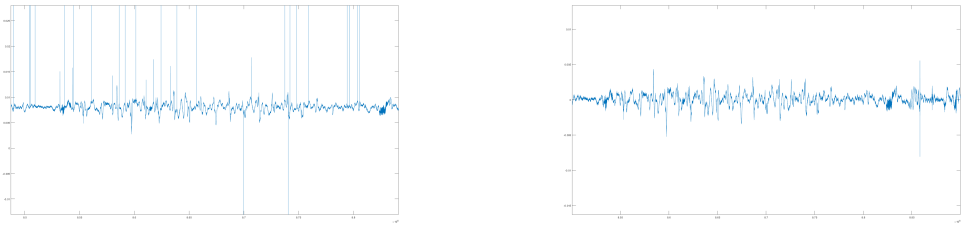


Figure 3: Filtered signal

Figure 4 showcases a zoomed-in portion of both the raw signal and the filtered signal. This magnified view provides a closer examination of the details within the signal, allowing for a more precise analysis of the impact of the filtering process. By visually comparing the raw and filtered versions in this smaller segment, one can observe the specific changes and improvements brought about by the applied filtering technique.



(a) small portion of raw signal (b) small portion of filtered signal

Figure 4: Filtered and raw signal comparison

2.3 Finding masticate signals in entire signal

It is of utmost importance to validate the previous team's efforts in accurately determining the chewing sequences within the entire signal. This verification process serves not only to ensure the correctness and accuracy of the data but also plays a critical role in establishing an approach or procedure for identifying such sequences in future work. By confirming the reliability of the chewing sequence determination, researchers can establish a robust foundation for subsequent analyses and investigations related to this specific aspect.

The subsequent section provides a detailed explanation of the chosen approach for identifying the mentioned sequences. Cross correlation, a widely accepted method for detecting a sequence in a signal, was initially considered. However, extensive research and experimentation revealed that cross correlation struggles to handle large sequences effectively, often resulting in confusion regarding the optimal match within the larger signal. Cross correlation may face challenges when dealing with large sequences due to several reasons. One of the main limitations is the computational complexity associated with cross correlation calculations for large data sets. As the sequence length increases, the number of comparisons required grows exponentially, resulting in a significant increase in computational resources and time.

Moreover, as the sequence size increases, the likelihood of finding multiple occurrences or similar patterns within the larger signal also increases. This can lead to ambiguity in determining the exact match or alignment between the two signals, making it difficult to identify the precise location of the desired sequence.

Furthermore, cross correlation assumes stationarity and linear relationships between the signals, which may not hold true in the case of large and complex data sets. Large sequences often exhibit nonlinear behavior and non-stationary characteristics, which can affect the accuracy of cross correlation-based matching.

To address these challenges, alternative approaches such as the multi-cross correlation method mentioned earlier can be employed. These modified techniques take into account the specific limitations of cross correlation for large data sets and offer more efficient and effective solutions for sequence identification.

To address this limitation, a modified approach called "multi-cross correlation" was devised.

In the multi-cross correlation approach, a sliding window was employed to divide the larger signal into segments with 50% overlap, ensuring comprehensive coverage of the entire signal. Each window was twice the length of the small signal portion (the sequence of samples) under consideration, and regular cross correlation was applied within each window. Consequently, a match, representing the most probable match sequence, was identified within each window.

To determine the correct match, a norm 2 distance calculation and thresholding technique were utilized. The norm 2 distance measured the dissimilarity between non-matching signals, with a larger value indicating a

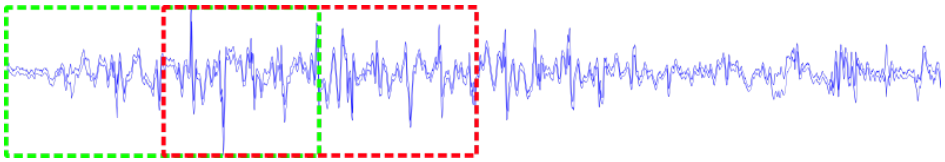
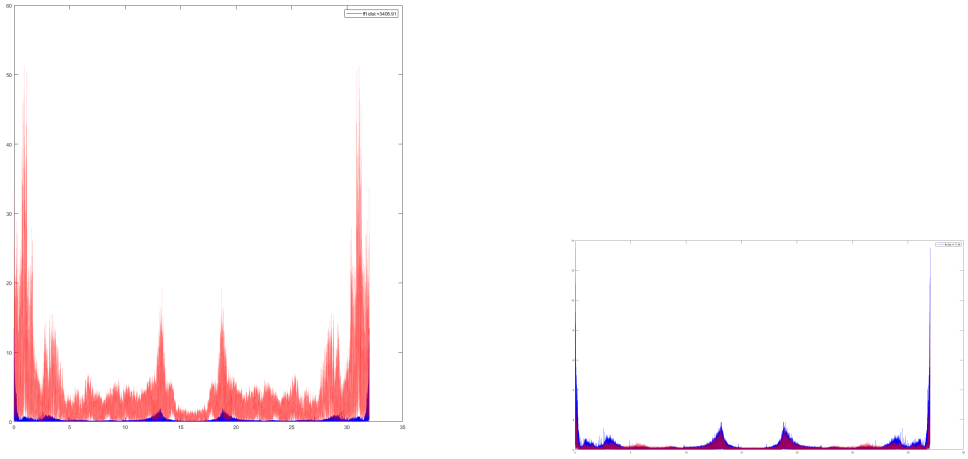


Figure 5: Sliding window with 50 per cent overlap

greater deviation from a perfect match. Note that in the proposed approach, a Fast Fourier Transform (FFT) method has been utilized to decompose the matched portion of signals within each window. By applying the FFT, the frequency components of the matched portion can be analyzed and represented in the frequency domain. Subsequently, the norm 2 distance calculation is performed on the decomposed signals.

By computing the norm 2 of the decomposed signals, the dissimilarity between the matched portion and the larger signal can be quantified. The norm 2 distance serves as a measure of the overall difference between the frequency components of the matched portion and the signal itself. This enables a more comprehensive evaluation of the match and enhances the accuracy of sequence identification.

The inclusion of the FFT and subsequent norm 2 distance calculation in the approach provides a more refined and detailed analysis of the frequency characteristics of the matched portion within the larger signal. This additional step enhances the effectiveness and reliability of the sequence identification process. By applying a threshold, the approach accounted for inconsistencies between the two signals under investigation, allowing for flexibility in accommodating perturbations and imperfect matches.



(a) Norm2 not matched FFT (b) Norm2 matched FFT

Figure 6: FFT matching comparison with norm2 distance

The inclusion of a threshold was essential to accommodate variations in the signals and ensure robustness in identifying the correct match. This adaptive approach enabled accurate sequence detection despite potential deviations and inconsistencies between the signals.

Figure 7 clearly demonstrates the effectiveness of the proposed algorithm in accurately identifying the desired match within the entire signal. The high precision achieved by the algorithm is evident, as indicated by the close alignment between the identified match and the expected sequence. This successful outcome validates the robustness and reliability of the algorithm in accurately locating the desired sequence within the larger signal. The results depicted in Figure 7 provide strong evidence of the algorithm's ability to achieve precise and accurate match identification, further affirming its suitability for the intended purpose.

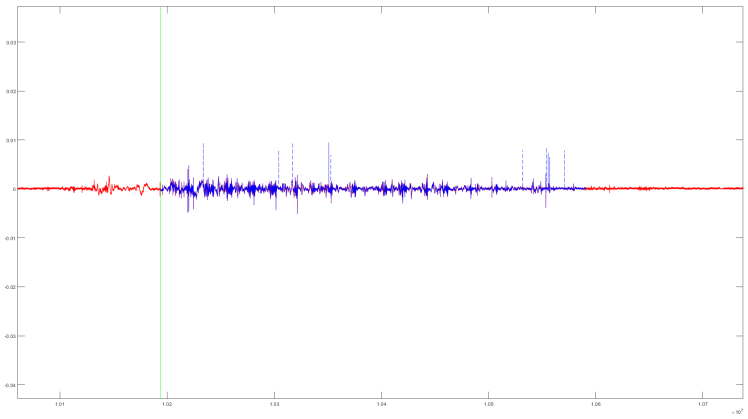


Figure 7: Start of match and matched signal in entire signal

2.4 Start of chewing annotation

Although not directly impacting the modeling section, it is imperative to validate the information regarding the start of chewing for each specific type of food within the chewing sequence dataset. While this validation process may not have a direct bearing on the modeling aspect, it plays a crucial role in ensuring the accuracy and reliability of the data. By verifying the provided information, researchers can ascertain the correctness of the annotated start times for chewing events, which in turn enhances the overall quality and integrity of the dataset.

In the pursuit of identifying the start of chewing within a sequence of chewing food, a specific approach developed by a group of researchers was employed. This approach, implemented by the researchers, successfully enabled the identification of the precise initiation of chewing events.

The detection of the beginning of each chew was facilitated by a relatively simple algorithm, as outlined in the referenced paper. This algorithm involved evaluating the short-time signal energy within a 20 ms window and comparing it to a predetermined energy threshold. If the short-time signal energy exceeded the threshold, the resulting signal was set to 1; otherwise, it was set to 0. Subsequently, the squared signal was subjected to low-pass filtering using a 4th order Butterworth filter. The specific choice of a filter with a 3dB cut-off frequency of 4 to 5 Hz effectively responded to the pause in phase 4 while filtering out the shorter pause in phase 2. The hill climbing algorithm was then employed to accurately detect the beginning of each chew, as depicted in Figure 8.

According to the findings presented in the paper, this algorithm demonstrated a high success rate, accurately detecting the start point of approximately 90% of all chews. Importantly, the algorithm exhibited minimal false insertions, further affirming its reliability and effectiveness in accurately identifying the onset of chewing events. LaTeX [1]

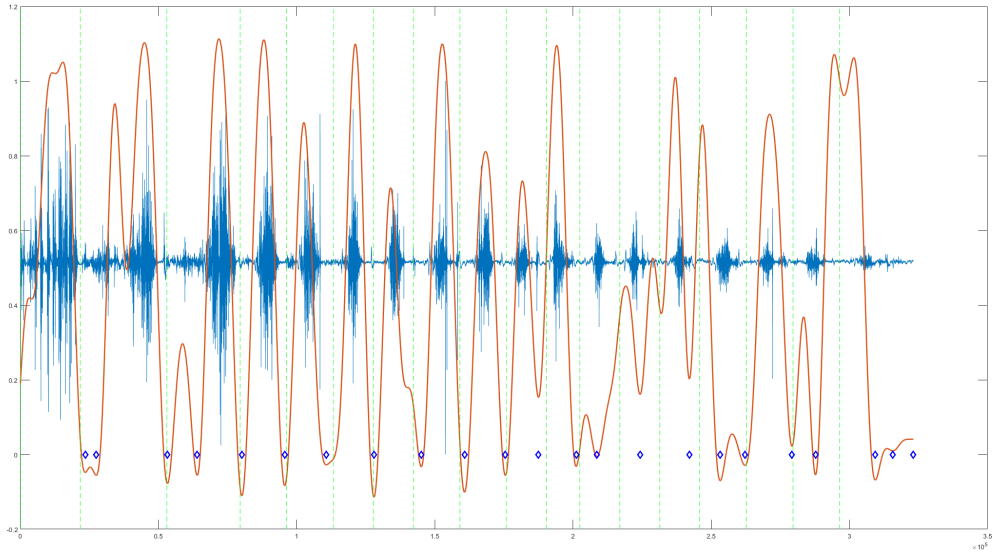


Figure 8: Start of chewing annotation

The observed differences between the annotated start of chewing and the start of chewing identified by the algorithm can be attributed to various factors, including inconsistencies in the definition of the start of chewing and the manner in which the data was recorded.

The definition of the start of chewing may vary among researchers and experts, leading to variations in the identification of this specific event. Different criteria or thresholds may be applied to determine the precise moment when chewing begins, resulting in discrepancies between manual annotations and algorithmic detection.

Furthermore, the process of recording the data itself can introduce certain limitations and challenges. Factors such as the positioning of sensors , variations in sensor sensitivity, and noise interference can impact the accuracy of identifying the exact start of chewing. These factors can contribute to differences between the algorithm's marked start of chewing and the annotations provided.

It is important to recognize and acknowledge these inconsistencies and limitations when comparing the algorithm's results with manual annotations. Such discrepancies can provide insights into the complexities of accurately identifying the start of chewing and highlight areas for improvement in future research and algorithm development.

2.5 Data Preparation for modeling

Data preparation plays a pivotal role in the modeling process and significantly influences the ultimate outcomes obtained. In our study, we have identified several models, including Support Vector Machines (SVM), Convolutional Neural Networks (CNN) (some variants including ResNet, EfficientNet, DenseNet), Recurrent Neural Networks (RNN), Transformers, and Vision Transformers (ViT), which we aim to implement. Each of these models has specific requirements and is suitable for different types of datasets. Consequently, we have formulated distinct datasets tailored to meet the specific needs of each model, taking into account their respective strengths. To cater to models that excel in sequence-based data analysis, such as RNNs, we have transformed our input signals into image representations known as spectrograms. By representing the signals as sequences in the time domain, we leverage the inherent sequential nature of the data, enabling effective utilization by these models. Furthermore, we have explored the extraction of relevant features from

the signals, enabling us to create tabular datasets. This approach is particularly useful for models that operate on tabular data, facilitating their application and harnessing the potential insights derived from the transformed signal data.

In the subsequent sections, we will elaborate on the preprocessing techniques employed and discuss the specific conditions and considerations involved in preparing the datasets tailored to each model's requirements.

As described in the preceding sections, we have prepared three distinct datasets for the modeling phase. The first dataset is in a tabular format, composed of statistical features and properties extracted from our signal in both the time and frequency domains. This dataset provides a representation of our signal data suitable for models designed to handle tabular data.

The second dataset consists of time series data, which comprises the original signal segmented into smaller chunks or subsequences, preserving its temporal sequence. This format caters to models that specialize in analyzing time series data.

Lastly, we have transformed the smaller time series data into spectrograms, representing them as images. This dataset, comprised of spectrogram images, is intended for models capable of processing visual data.

By creating these three diverse datasets, we can leverage the unique capabilities of various models and select the most appropriate one for our specific analysis, allowing for a comprehensive exploration and interpretation of the data from multiple perspectives.

The approach employed for data preparation involved using the isolated signals of mastication corresponding to a specific food type. These signals were further segmented into smaller portions using a sliding window technique with a 60% overlap and a window size of approximately 0.8 seconds, resulting in a window size of around 20,000 sample data points. The choice of 0.8 seconds as the window size was based on the statistical distribution of chewing cycles within the available data (Figure 9).

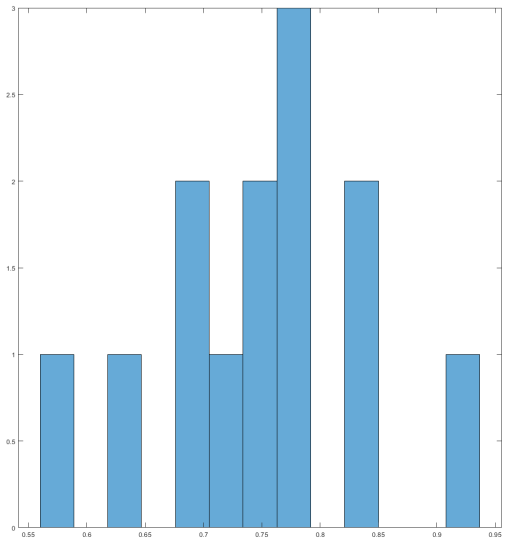


Figure 9: Histogram for one food type based on time distribution

Food Type	MEAN	STD
apple	0.7498	0.0928
baguette	0.8148	0.1013
beef jerky	0.6631	0.1382
cheddar	0.8187	0.0661
chips	0.6871	0.1192
Total	0.7135	0.0700

In the process of data generation, we employed stratified sampling to ensure that the distribution of data remained consistent across the different sets—train, validation, and test. The split distribution for these sets was selected as 75%, 15%, and 15% respectively, ensuring an appropriate representation of the data in each subset.

Note that, stratified sampling is a sampling technique used in statistics and data analysis to ensure that each subgroup or category within the dataset is adequately represented in the sample. The goal of stratified sampling is to create a representative sample that accurately reflects the distribution of the entire population with respect to specific characteristics or categories.

In the context of data preparation, stratified sampling involves dividing the dataset into distinct subgroups based on certain attributes or classes. The samples are then randomly selected from each subgroup in proportion to their representation in the entire dataset. This approach helps to prevent biased sampling and ensures that rare or underrepresented categories are not overlooked.

For instance, if we have a dataset with three classes (A, B, and C) and class A represents 60% of the data, class B represents 30%, and class C represents 10%, stratified sampling will ensure that the samples chosen for the training, validation, and testing sets will maintain this distribution. This helps to produce a balanced and representative sample that allows for more accurate and reliable analysis and modeling.

By employing this data preparation methodology, we aimed to create representative datasets that adequately captured the essential characteristics of the chewing signals while maintaining data integrity and consistency throughout the training, validation, and testing phases of our analysis.

2.5.1 Tabular Dataset

A tabular dataset refers to a structured form of data organization, typically presented in the form of rows and columns like a table. Each row represents a specific data sample or instance, while each column corresponds to a particular attribute or feature of that data sample. In the context of our study, the tabular dataset is constructed based on statistical features and properties extracted from the signal data in both the time and frequency domains.

Support Vector Machines (SVM) is a popular machine learning algorithm used for classification and regression tasks. In the context of classification, SVM aims to divide the data samples into distinct classes by finding an optimal hyperplane that maximizes the margin between different classes. This hyperplane serves as the decision boundary, and data points are classified based on which side of the hyperplane they lie.

To utilize the tabular dataset for classification using SVM, we first define classes based on the problem at hand. For example, in a binary classification scenario, we might have two classes : "chew" and "non-chew". Each row in the tabular dataset represents a data sample, and the columns correspond to the statistical features extracted from the signal data.

During the training phase, SVM learns the optimal hyperplane that best separates the data samples into their respective classes. It does this by finding support vectors, which are data points located closest to the hyperplane. The goal is to maximize the margin between these support vectors while minimizing the classification error.

Once the SVM model is trained, it can classify new, unseen data samples by determining on which side of the learned hyperplane they lie. Data samples falling on one side are assigned to one class (e.g., "chew"), while those on the other side are assigned to the other class (e.g., "non-chew").

The SVM algorithm's ability to work effectively with tabular datasets lies in its ability to handle high-dimensional feature spaces and find complex decision boundaries. By extracting meaningful statistical features from the signal data and constructing a tabular dataset, we can leverage SVM's strengths for accurate and efficient classification tasks in our study.

In the modeling section, a comprehensive and detailed explanation of Support Vector Machines (SVM) will be provided.

In this section of the study, we employed a tabular dataset consisting of more than 2500 rows, each representing a distinct sample. The dataset comprised 16 different features, encompassing both time domain and frequency domain attributes extracted from the signal data.

The time domain features included:

1. Mean
2. Standard deviation
3. Root Mean Square (RMS)
4. Variance
5. Skewness
6. Kurtosis
7. Entropy

The frequency domain features included:

1. Mean power
2. Standard deviation
3. Spectral centroid
4. Spectral spread
5. Spectral skewness
6. Spectral kurtosis
7. Band energy ratio
8. Peak frequency
9. Spectral entropy

Here is a brief explanation of each feature:

1. Mean: The arithmetic average of the data points, representing the central tendency of the distribution.
2. Standard Deviation (std): A measure of the dispersion or spread of the data points around the mean, indicating the variability of the data.
3. Root Mean Square (RMS): The square root of the average of the squared data points, capturing the effective magnitude of the signal.
4. Variance: The average of the squared differences between each data point and the mean, measuring the spread of the data.
5. Skewness: A measure of the asymmetry of the data distribution, indicating the degree of deviation from a symmetrical bell-shaped curve.
6. Kurtosis: A measure of the tailedness or heaviness of the data distribution, quantifying the presence of outliers or extreme values.

7. Entropy: A measure of the uncertainty or disorder in the data, assessing the amount of information contained in the signal.
8. Mean Power: The average power of the signal in the frequency domain.
9. Spectral Centroid: The center of mass of the power spectrum, indicating the average frequency of the signal.
10. Spectral Spread: A measure of the spread of the power spectrum around its centroid, capturing the bandwidth of the signal.
11. Spectral Skewness: A measure of the asymmetry of the power spectrum, indicating the skewness of the frequency distribution.
12. Spectral Kurtosis: A measure of the tailedness of the power spectrum, quantifying the presence of extreme values in the frequency distribution.
13. Band Energy Ratio: The ratio of the energy in a specific frequency band to the total energy in the signal.
14. Peak Frequency: The frequency with the highest amplitude or power in the signal.
15. Spectral Entropy: A measure of the amount of information or complexity in the frequency distribution.

Each feature provided valuable insights into the characteristics and properties of the signal data, capturing relevant information from both the time and frequency domains. The tabular dataset, equipped with these comprehensive features, served as a robust foundation for the subsequent implementation of the Support Vector Machines (SVM) algorithm for classification tasks. By leveraging this feature-rich dataset, we aimed to harness the discriminative power of SVM to effectively classify the data samples into their respective classes, thereby addressing the objectives of our study with precision and accuracy.

It is also worth mention that Support Vector Machines (SVM) are versatile and capable of handling various types of data, including **image** data. While SVM is widely known for its effectiveness in handling tabular data and sequence-based data (time series), it can also be extended to work with image data through appropriate feature engineering.

When dealing with image data, the key is to represent the images as feature vectors that SVM can process. This process often involves extracting relevant features from the images, such as color histograms, texture descriptors, or using techniques like bag-of-visual-words to represent the images as vectors. Once the images are suitably represented as feature vectors, SVM can be applied for image classification, object recognition, and other image-related tasks.

By utilizing SVM for image data analysis, we can benefit from its ability to find optimal decision boundaries in high-dimensional feature spaces, making it suitable for tasks that involve complex and non-linear relationships in image data. SVM's versatility and effectiveness in handling multiple data types make it a valuable tool in various machine learning applications , including those involving image analysis and processing.

2.5.2 Time Series Dataset

By implementing the sliding window method for segmenting the signals, we can conveniently store the resulting small segments into separate files, effectively referring to them as time series data. This approach allows for the organized and structured storage of the segmented data, facilitating easy access and analysis of individual time series segments during subsequent stages of data processing and modeling.

2.5.3 Spectrogram Dataset

A spectrogram is a visual representation of the frequency content of a signal over time. It is a commonly used tool in signal processing and audio analysis to analyze the frequency components of a time-varying signal, such as audio, speech, or any other time-series data.

1. Time Segmentation: To create a spectrogram, the time-varying signal is first divided into short, overlapping segments or frames. Each frame typically ranges from a few milliseconds to tens of milliseconds in duration.
2. Fourier Transform: For each frame, a mathematical operation known as the Fourier Transform is applied. The Fourier Transform converts the time-domain signal into its corresponding frequency-domain representation. This process decomposes the signal into its individual frequency components, revealing the contribution of each frequency to the signal at that specific moment in time.
3. Power Spectrum: The magnitude of the Fourier Transform represents the amplitude or strength of each frequency component in the signal. By squaring the magnitude values, we obtain the power spectrum, which quantifies the energy distribution across different frequencies.
4. Visualization: The power spectrum of each frame is typically displayed as a 2D image, where the x-axis represents time, the y-axis represents frequency, and the color intensity or shading denotes the power or magnitude of the corresponding frequency component. Brighter colors indicate higher power or energy at specific frequencies, while darker colors represent lower power.

The resulting spectrogram provides valuable insights into the frequency content and spectral characteristics of the time-varying signal. It allows us to visualize how the signal's frequency components change over time, identifying patterns, harmonics, and other interesting features that may not be evident in the time-domain representation alone. Spectrograms are widely used in various fields, including speech processing, music analysis, radar signal processing, and many other applications that involve analyzing time-frequency representations of signals.

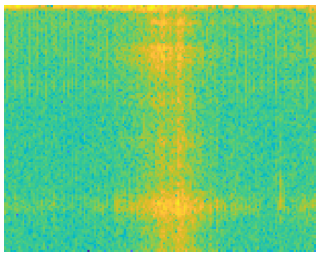


Figure 10: Spectrogram 1

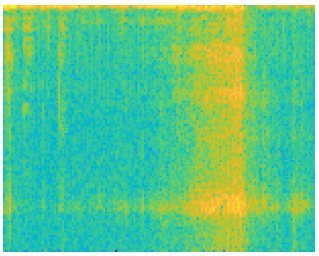


Figure 11: Spectrogram 2

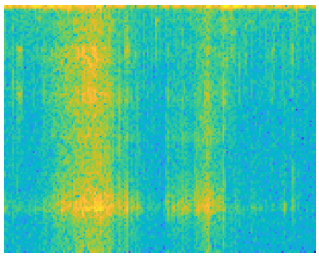


Figure 12: Spectrogram 3

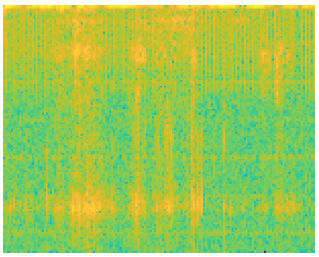


Figure 13: Spectrogram 4

Figure 14: Spectrogram samples from chewing an apple

3 Modeling

3.1 SVM

3.2 RNN

A Recurrent Neural Network (RNN) is a type of neural network architecture designed to handle sequential data. Unlike feed-forward neural networks, RNNs introduce feedback loops that allow them to process inputs in a sequential manner, where the current input not only depends on the current time step but also on the information from previous time steps. This recurrent structure makes RNNs suitable for tasks involving time series data, natural language processing, and other sequential data analysis.

The primary challenge with traditional RNNs is the vanishing gradient problem, where gradients become exponentially small during backpropagation through time, leading to difficulties in learning long-range dependencies. As a result, traditional RNNs may struggle to retain information over long sequences, limiting their effectiveness in capturing long-term patterns.

3.2.1 BiRNN

Bidirectional RNNs (BiRNNs) address the limitation of traditional RNNs by processing input data in both forward and backward directions.LaTeX [2] By considering the context from both past and future time steps, BiRNNs can capture bidirectional dependencies in the sequential data. This dual processing of information allows BiRNNs to gain a better understanding of the data and improves their ability to model complex temporal relationships.

3.2.2 LSTM

Long Short-Term Memory (LSTM) is a specialized variant of RNNs that overcomes the vanishing gradient problem.LaTeX [3] LSTMs introduce additional memory cells and gating mechanisms, which include an input gate, a forget gate, and an output gate. These gates control the flow of information in and out of the LSTM cell, allowing the network to selectively retain or forget information over multiple time steps.

The input gate determines how much of the new input to incorporate into the memory cell, while the forget gate controls what information to discard from the cell's memory. The output gate governs how much information to pass to the next time step. The ability to selectively remember and forget information makes LSTMs suitable for tasks requiring long-term dependencies and memory retention.

Following training with various configurations, ranging from very low learning rates (1e-6) to finer learning rates (1e-3), and experimenting with increasing the model's capacity through additional layers and larger hidden states, the unfortunate outcome was that the train loss reached a value around 1.4. Moreover, as shown in Figure 15, the train accuracy did not surpass approximately 36 percent, indicating that the model did not effectively learn meaningful patterns from the data.

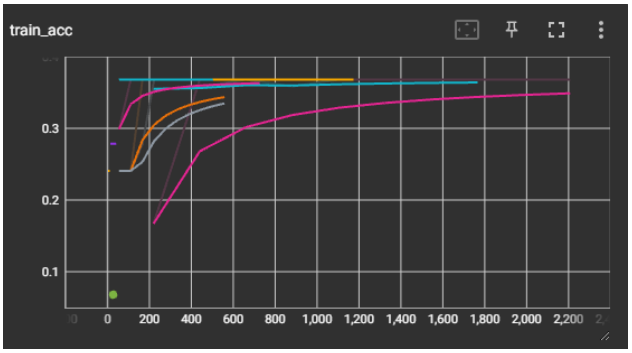


Figure 15: Train accuracy for LSTM on dataset v2.0

The limited performance can be attributed to several factors, including:

1. A potential bug in the implementation.
2. Poor quality of the data used for training.
3. Insufficient model capacity, which may hinder its ability to capture complex patterns.
4. Low learning rate, which might slow down the learning process and hinder convergence.

After careful debugging and ruling out the presence of a bug, as well as addressing the issues of low model capacity and learning rate, the most likely remaining cause for the suboptimal performance is the quality of the data used to train the model. Improving the data quality might lead to better results and allow the model to learn more meaningful representations from the available data.

As shown in Figure 18, it is evident that signals with the same label exhibit a highly diverse range of shapes and patterns. This substantial variation among signals with identical labels could potentially be the underlying cause for the model's confusion and subsequent low accuracy. The diverse shapes in signals belonging to the same class may introduce significant challenges for the model during the learning process. It becomes difficult for the model to generalize effectively and distinguish between different instances of the same class due to the substantial intra-class variations.

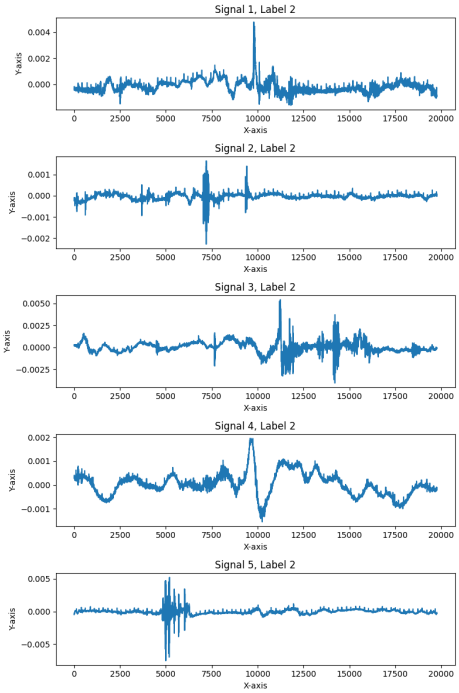


Figure 16: Label 2 (beef jerky)

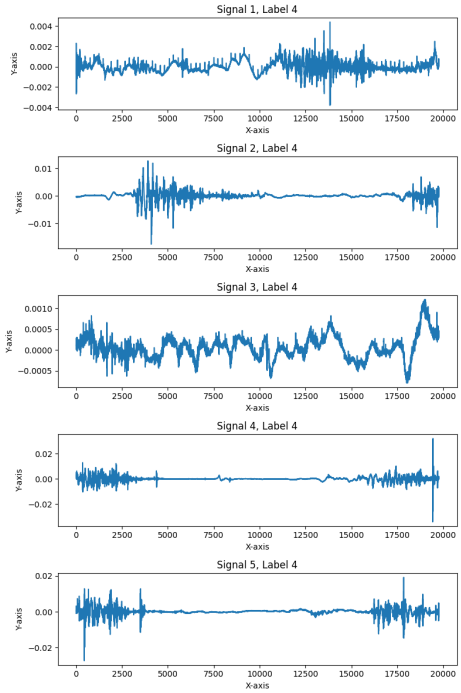


Figure 17: Label 4 (chips)

Figure 18: Signals from two specific labels

In such cases, the model may struggle to learn representative and discriminative features for each class, resulting in misclassifications and reduced accuracy. To address this issue, efforts should be made to enhance the quality of the training data , ensure consistency in data collection, and potentially explore data augmentation techniques to increase the diversity of samples within each class. These measures can aid the model in better capturing the underlying patterns and improve its accuracy when dealing with signals exhibiting considerable intra-class variations.

3.2.3 New Version of the Data Set

A promising and effective approach to address the challenges posed by the diverse shapes and characteristics within the same class of signals is to split the data based on annotations. By considering both the food signals and the corresponding chewing annotations, a newer version of the dataset has been created, specifically tailored for RNN and Transformer models.

The advantage of this new dataset lies in its ability to accommodate different data lengths, which is essential for RNN and Transformer architectures that can handle variable-length sequences. By associating the food signals with their corresponding chewing annotations, the dataset aims to establish a higher level of similarity among signals with similar labels in terms of their shapes and other characteristics.

This approach seeks to provide the models with more coherent and consistent examples for each class, reducing the intra-class variations and enabling them to better capture the underlying patterns and similarities. As a result, the RNN and Transformer models can benefit from a more focused and representative dataset, potentially leading to improved accuracy and performance in classifying signals with diverse shapes but similar labels.

3.2.4 GRU

The Gated Recurrent Unit (GRU) LaTeX [4] is another variant of RNNs that shares some similarities with LSTM but has a simpler architecture. GRUs use a reset gate and an update gate, combining the functionality of the input and forget gates in LSTM. The reset gate controls which past information to forget, while the update gate determines how much of the new input to incorporate into the cell's memory. GRUs have been shown to be computationally more efficient than LSTMs, making them particularly suitable for smaller datasets and applications requiring medium-range dependencies.

In summary, RNNs and their variants like BiRNN, LSTM, and GRU are powerful tools for processing sequential data. Each variant has specific strengths and applications, and researchers continue to explore improvements and variations to better address the challenges of capturing temporal dependencies and long-term information in sequential data.

3.3 Transformers

The Transformer is a deep learning architecture that was initially introduced for natural language processing tasks, particularly machine translation. It has since become a fundamental building block in various applications due to its ability to handle sequential data efficiently.LaTeX [5]

In the context of time domain signal classification tasks, the Transformer can be adapted to process the sequential data effectively. Instead of using traditional RNN-based approaches, the Transformer employs self-attention mechanisms to capture long-range dependencies within the time domain signal.

Self-attention allows the model to weigh the importance of different time steps when processing a particular time step. This attention mechanism enables the Transformer to consider the entire signal simultaneously, making it well-suited for capturing temporal relationships and dependencies, even across long sequences.

In the classification task, the Transformer can be combined with additional layers to perform the final classification based on the extracted features from the time domain signal. The model can efficiently process variable-length signals without the need for recurrent connections, leading to faster training and reduced computational complexity.

Overall, the Transformer's attention-based architecture makes it a powerful choice for time domain signal classification tasks, allowing it to effectively capture temporal patterns and dependencies in the data, leading to accurate and efficient classification results.

3.4 CNN

References

[1] Amft, O., et al. "Analysis of Chewing Sounds for Dietary Monitoring." section. 5.1, pp. [12].

[2] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681.

[3] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[4] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

[5] Attention Is All You Need by Vaswani et al. (2017)