Machine Learning for Parenting: Using Supervised Machine Learning to Predict RIFL Scores Using Text and Audio Mediums

By Christopher Mountain

Supervisor: Eldan Cohen

April, 2024

B.A.Sc. Thesis

Division of Engineering Science
UNIVERSITY OF TORONTO

Machine Learning for Parenting: Using Supervised Machine Learning to Predict RIFL Scores Using Text and Audio Mediums

Christopher Mountain
Engineering Science
University of Toronto

Eldan Cohen
Thesis Supervisor
Industrial Engineering
University of Toronto

Abstract

Predicting and classifying Responsive Interactions for Learning (RIFL) scores in parent-child interactions pose significant potential for advancements in developmental psychology. This paper explores the efficacy of various supervised machine learning approaches to predict or classify RIFL scores using text transcripts and audio data. Utilizing a small dataset of 253 labeled samples of parent-child interactions, we implemented and compared a range of techniques including linear regression, XGBoost, Recurrent Neural Networks (RNN) with Gated Recurrent Units (GRU), transformers, and the novel ROCKET (RandOm Convolutional KErnel Transform) method. Despite rigorous experimentation with state-of-the-art machine learning algorithms, our findings indicate that the current volume and distribution of data do not permit the development of a model that generalizes sufficiently to unseen data for accurate RIFL score prediction. A transformer model trained on transcript data showed relative promise, achieving up to 70% validation accuracy while showing clear signs of overfitting. This result suggests a larger, more balanced dataset might yield improvements. The study highlights the complexity of capturing nuanced parent-child interactions and sets a foundation for future work which could include leveraging video data, larger datasets, and novel language models for enhanced prediction and classification of RIFL scores.

The source code for this study is available at
`https://github.com/ChrisMountn/Undergraduate-Thesis`.

Acknowledgements

A huge thank you to Eldan Cohen, my thesis supervisor, for the invaluable wisdom and support throughout this project. Thanks to Michal Perlman, Jenny Jenkins, and Samantha Burns, as well as the Ontario Institute for Studies in Education for their collaboration and for enabling this research to be possible in the first place.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

Our understanding of and ability to classify parent-child interactions at low cost is important to progress in the field of developmental psychology. Can we predict outcomes for children based on parenting styles and behaviours? More importantly, how can we build systems that improve outcomes for children? How do we scale these systems and address the scarcity of highly educated, highly paid experts required to meet the demand for these interventions? In this thesis, we aim to use ML techniques to classify parent-child interactions, deepen our understanding of important features of these interactions, and build systems to support parents, researchers, and workers in the field of developmental psychology. We work with researchers at the Ontario Institute for Studies in Education (OISE) who have prepared audio-video data labeled using the Responsive Interactions for Learning (RIFL) framework, a composite metric designed to gauge the quality of interactions between children and parents [2].

1.2 Objective

We aim to address one central research question: *To what degree of accuracy can we predict or classify RIFL scores using multimodal supervised ML via text transcript or audio mediums*?

1.3 Significance of Proposed Research

The proposed research would have the potential to improve our understanding of parent-child interactions and automate the analysis and classification of these interactions at scale. This would increase our ability to intervene and improve outcomes at scale for children who might otherwise be subject to problematic parent-child interactions at an early stage in life. Examples of good environments to implement interaction supervision would be in social service contexts or classrooms. Depending on the efficacy achieved for results of different mediums, an intervention could be achieved at greater levels of privacy and anonymity. For example, should text transcripts of parent-child interactions be highly indicative of RIFL score or RIFL score-group, an intervention could be conducted simply by automatically transcribing an audio-recording of an interaction and generating an analysis of the sample. However, if pure audio or video provided a higher efficacy or were more

predictive, then the audio recording itself or even an audio-video recording of a parent-child interaction could be fed into the model. We remain cognizant that the less invasive the data collection procedure, the better, as far as developing a practical and applicable intervention for parent-child interactions on a large scale.

The proposed research also stands to improve our understanding of the features that contribute to the context with which we perceive parent-child interactions. For example, a positive result in the purely text/transcript domain would indicate that these interactions, or at least our interpretation of these interactions, is based primarily on the words that are said rather than the way in which they are said. On the other hand, a poor result in the text domain contrasted with a performant model in the audio domain may indicate that intonation and audio-native features in communication do, in fact, play a major role in the developmental impact of interactions between a child and parent.

Achieving a meaningful result also has the potential to make broader impacts on research in developmental psychology. Although the ML techniques we employ are not novel, the volume of current research that intersects psychology and state-of-the-art Machine Learning is still sparse. By publicizing ML results in a field like developmental psychology, we hope to open the door for researchers to continue to investigate and derive results from the intersection of the two disciplines.

2 Literature Review

In this section we discuss relevant prior literature and field-specific research tools relevant to our ML prediction objective.

2.1 RIFL Scores, CLASS Scores

RIFL scores are a long-standing tool in the field of family psychology used to evaluate the quality of parent-child and inter-sibling interaction using a five-minute witnessed or recorded interaction [3]. They evaluate the interaction over a number of categories related to the quality and content of the interaction, which can be averaged to produce a RIFL score for the interaction. Some examples of the individual prompts that make up the composite score are “This educator uses constructive non-verbal communication”, “This educator is responsive to what children know and understand”, and “This educator is warm and affectionate”, where the educator is the parent. The final value lies between 1 and 5, inclusive. RIFL scores have been shown to correlate well with other independently developed parent-child interaction

metrics, and also long-term outcome data including literacy, social wellbeing, and classroom performance, among others [3].

The Classroom Assessment Scoring System (CLASS) score is a quantitative measurement designed for observation of interactions between students and staff in schools. This metric is thought to be an important indicator of children's outcomes [4]. The CLASS score is sufficiently similar in nature to RIFL scores that machine learning techniques and models employed in predicting CLASS scores might reasonably be effective on RIFL labeled datasets.

2.2 Multimodal Machine Learning Approaches in the Field of Developmental Psychology

At the beginning of 2023, a paper titled Toward Automated Classroom Observation: Multimodal Machine Learning to Estimate CLASS Positive Climate and Negative Climate by Ramakrishnan et al. was published in the IEEE Transactions on Affective Computing journal [5]. This paper uses ML techniques on audio and video data to predict CLASS scores based on a labeled dataset of about 300 15-minute video segments. Similarly, we are working to develop predictive models on a labeled dataset of 5 minute video segments. There are a few important techniques pioneered for this application in the paper. First, the researchers achieve a reasonably high efficacy on audio-only analysis using the PyAudio library [6] to generate features over a temporally divided audio file passed through an RNN. Second, the researchers develop an approach to video analysis where pre-trained facial sentiment feature vectors are concatenated with pre-trained generic visual feature information, and again passed through an RNN. Finally, the researchers show that an unweighted ensemble of the CLASS predictions between the two models is a valid improvement over either model by itself. In this thesis we experiment with similar approaches to Ramakrishnan et al. in the RIFL domain.

In the paper Improving Patient Safety Event Report Classification with Machine Learning and Contextual Text Representation by Chen et al., the researchers describe a now common approach to training ML models on natural language without the need for traditional Natural Language Processing [7]. Instead, the researchers use a pre-trained text embedding model (Roberta-Base in this case) to create representative feature vectors for each text document, and are able to achieve good classification performance on these feature vectors using standard classifiers (a support vector machine performed well). In this thesis, transcript embeddings of parent child

interaction are used to train classifiers and regressors, attempting to achieve a similar result.

3 Methods and Approaches

In this section we discuss the data, methods, and approaches used to address our research objective. We learned a lot along the way, and were prompted by many of the results to pivot throughout the course of the research project.

3.1 Data Description

The dataset consists of 253 labeled samples. The samples were taken from recordings of parent-child interactions conducted in a research lab. Each parent child interaction was centered around one of two objectives. For the first objective, parents were instructed to read a book to their child. For the second objective, parents were instructed to help their child complete a puzzle involving block placement. The activities were designed to elicit natural interaction between the parent and child, in order to generate reasonably meaningful or accurate data. Each sample comprises a recording of the interaction between three and six minutes, and a text transcript of the recording. The video interaction was shown to up to 11 independent RIFL assessors, who each generated a RIFL score for the sample. Finally, the label was generated by taking the average RIFL score between the assessors for a given sample. The distribution of RIFL labels ended up heavily favouring the middle values, likely as a result of both the actual demographic distribution and because the score is an average over several assesments of a composite metric. Low frequency tails lie on either end for very high and very low RIFL scores in the distribution as in Figure 1.

Each sample of data had both an audio component, stored as a .wav or .mp4, and transcription component, stored as either a .txt or .docx file. The transcription files contained were line-separated sentences, phrases, and exclamations spoken by both the parent and child. The line separation was somewhat arbitrary and did not correspond to a change of speaker. The audio files contained raw audio from the original video recording.

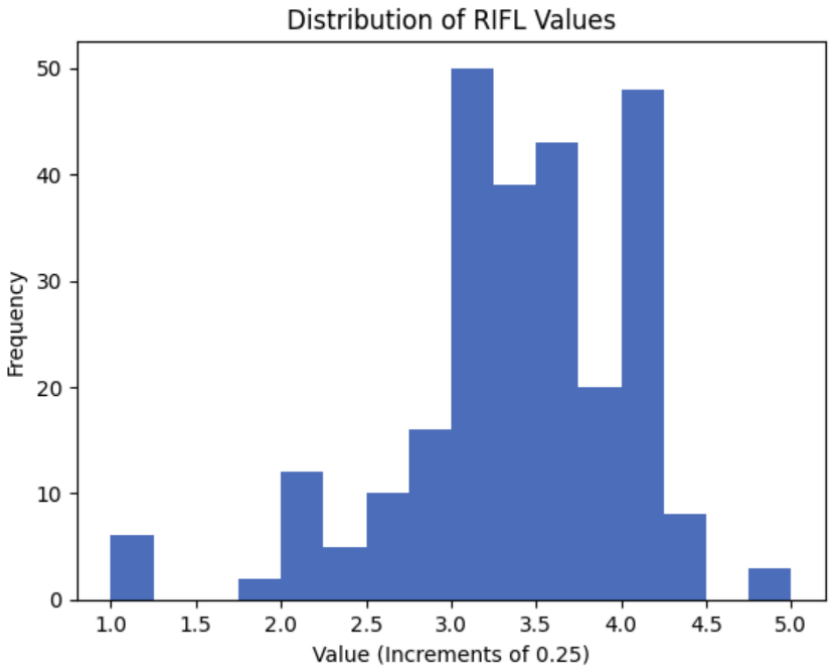


Figure 1: RIFL score distribution over dataset

3.2 Continuous Label Regression on Transcripts

3.2.1 Explanation and Justification for Approach

The first approach taken was to predict the continuous RIFL value for a given sample using the text transcript. A low MSE or high Pearson Correlation for RIFL score prediction using the transcript only would confirm that the written content of the parent-child interaction contained sufficient information to predict scores. Due to the small size of the dataset, we opted to use pre-trained language embeddings to enrich the meaning of the text data before performing the regression.

3.2.2 Data Pre-Processing and Embedding-Based Approach

The transcript data were converted all to the .txt format. We used the xlm-roberta-base [8] model to embed each transcript, in its entirety, into a 768-dimensional vector with semantic meaning.

The xlm-roberta-base model was trained in a self-supervised fashion on 2.5TB of filtered CommonCrawl data from around the internet, and is a reasonable choice both to embed semantic language analysis and for sentiment analysis applications. We had no prior knowledge of whether semantic language meaning or sentiment

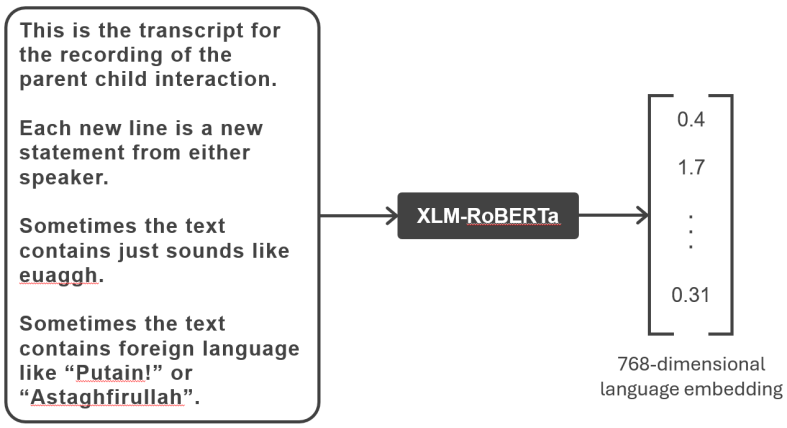


Figure 2: Visual depiction of embedding process

analysis would play a greater role in determining the RIFL outcomes. Additionally, xlm-roberta-base was trained on a multilingual dataset, and can meaningfully embed multiple languages. We hypothesized that this might be a useful feature of the model as there were examples in some of the parent-child interactions of a parent or child reverting to a non-english language, often in moments of tension or emotion. The 768-feature embedded vectors were paired with their labels and split using a 0.7, 0.15, 0.15 ratio into a training, validation, and testing set. A smaller validation and testing set were used to give the model as much data to learn from as possible over the small dataset. Although the validation and test results were volatile due to small partitions, we repeated our shuffled split 10 times and retrained for major experiments, taking the average of the 10 results, in order to regress to the mean result for the limited sample sizes. Because we did not ultimately find a model that performed with high efficacy during our experimentation, and as a result were able to verify the important characteristics of their underperformance on the validation set, we show results on the validation data instead of the test data for the models and methods below.

3.2.3 Linear Regression

Standard Linear Regression was used to fit a linear model to the training embeddings. In validation, the MSE was found to be 1.15, while the MAE was 0.85. The Pearson correlation was 0, indicating that linear regression produced no statistically significant result. Linear Regression failed to capture the patterns in the data as shown in Figure 3.

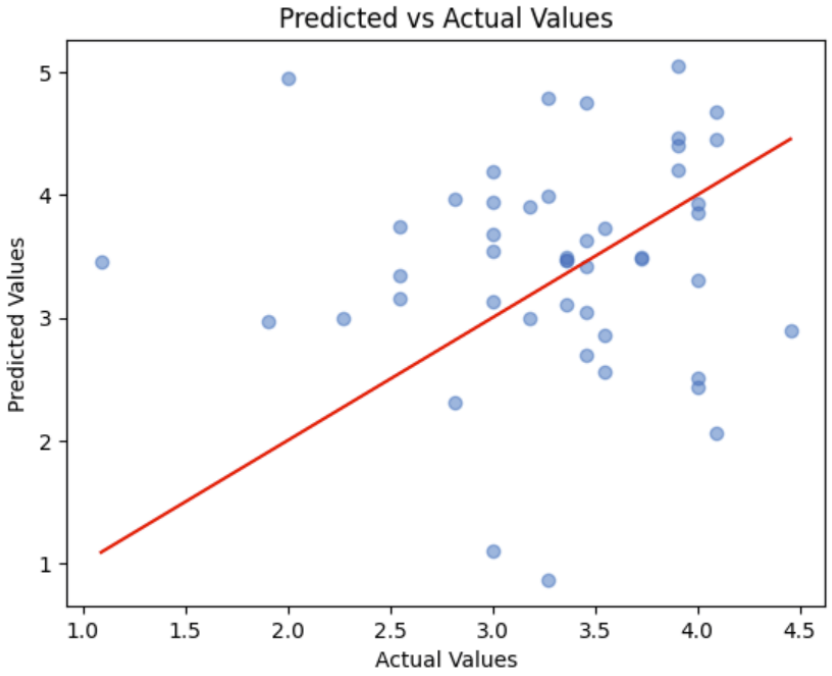


Figure 3: Predicted vs. Actual Values for Linear Regression

3.2.4 XGBoost + Support Vector Regression

An unweighted ensemble of XGBoost Regression and Support Vector Regression was found to work slightly better. XGBoost (Extreme Gradient Boosting) Regression is an implementation of gradient boosting, utilizing decision trees as base learners in an ensemble method that sequentially corrects the mistakes of prior trees to improve prediction accuracy. SVR is an adaptation of Support Vector Machines (SVM) for regression tasks, where the goal is to find the best-fitting hyperplane in a high-dimensional space that deviates from the target values by a margin no greater than epsilon, effectively trying to fit as many data points as possible within this margin while minimizing the error. The best use of these two techniques was found empirically to be ensembling them together, which achieved an MSE of 0.51, MAE of 0.50, but a Pearson Correlation of just 0.03. This model failed to capture the pattern of outliers in the data, and generally predicted a value between 3 and 4 as shown in Figure 4. This resulted in a relatively low MSE strategy given the distribution of the data, but does not meaningfully generalize to unseen data.

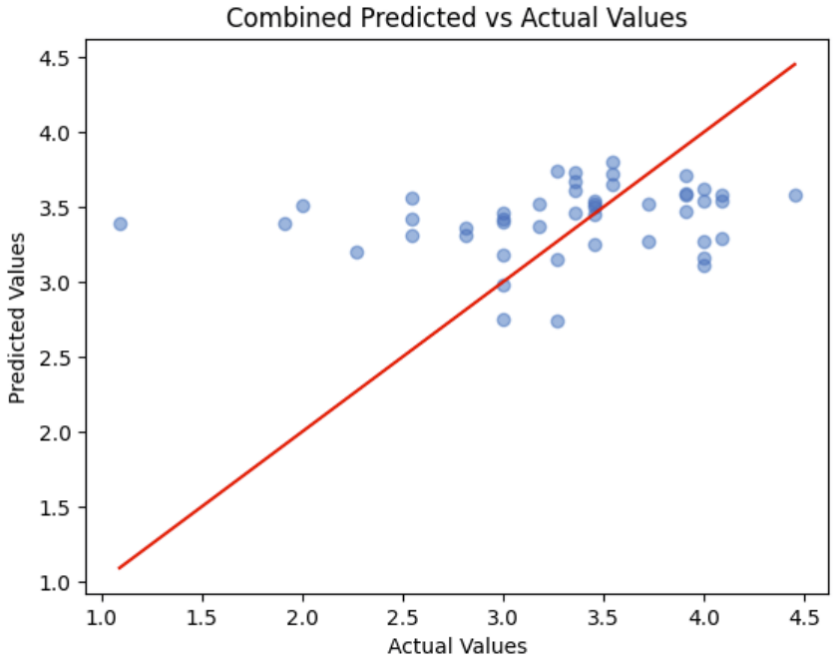


Figure 4: Predicted vs. Actual Values for XGBoost + Support Vector Regression

3.2.5 XGBoost Ranking

XGBoost Ranking was also used in experimentation. XGBoost Ranker is a variant of XGBoost designed specifically for learning-to-rank tasks, which are common in areas like information retrieval and recommendation systems. Unlike traditional regression or classification models, the goal of a ranking model is to predict the correct order of items rather than predicting their exact values or classes. In the context of predicting the correct ranking of an ordered set of vector embeddings where the label is a continuous value, XGBoost Ranker was used to learn the relative ordering among the items based on their vector embeddings and their associated continuous labels. The intuition for this was that although predicting RIFL values directly may be difficult, especially given the unevenly distributed dataset, it may be feasible to learn to compare documents between each other and build a valid ranking.

Additionally, from an objective-based point of view, most applications of categorizing RIFL scores would actually be centered around categorizing RIFL scores relative to one another. For example, when determining which parent-child interactions may require an intervention out of a cohort of many samples, you probably just want to look at the bottom x%. On the other hand, for finding exemplary parent-child

interactions with high RIFL scores, the top x% could be just as easily extracted. However, comparing the Pearson correlation of the XGBoost Rank-ordered document list using pairwise comparisons to the actual labeled data showed no statistical significance at 0.08. No MSE or MAE can be reported as scores were not predicted.

3.2.6 Reformulation for RNN

The next thing we tried was to reformulate the problem as a sequence regression problem for an RNN. For this, we no longer could use just a single vector embedding to represent the transcription. Instead, we embedded each piece of text line-by-line, again using xlm-roberta-base, resulting in a sequence of tokens representing each transcript. The token sequences were padded using all-0 vectors by convention to a length matching that of the longest sequence (203).

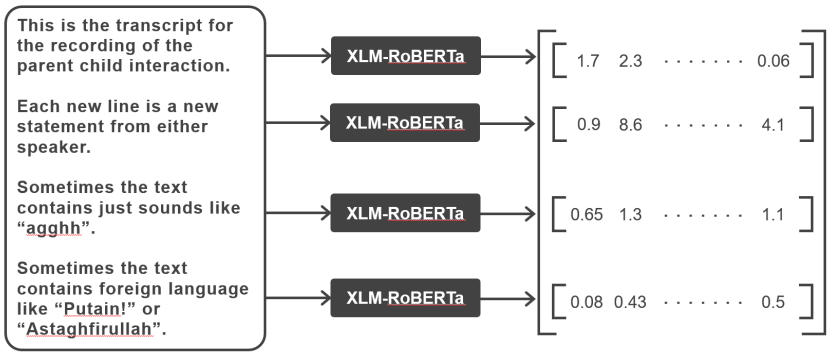


Figure 5: Visual Depiction of Sequence Embedding

3.2.7 RNN with GRU For Regression Training

A GRU architecture was chosen somewhat arbitrarily. According to a paper by Cahuantzi et al., GRUs tend to perform better on low-complexity sequences whereas LSTMs perform better on high-complexity sequences [9]. Because of our small dataset and pre-trained feature mapping, we determined that a high-complexity sequence may not be learnable, and chose the GRU for improved performance in the low-complexity domain. During validation, a two-layer bi-directional GRU with an attention mechanism followed by three fully connected MLP layers with ReLU activation functions between the non-output layers resulting in a single value was found to be sufficient for learning. The hidden dimension for the GRU was 368, and a dropout probability of 0.07 was used to combat overfitting. We used a Mean Squared Error loss criterion on the single output of the model where the target

was the discrete value of the RIFL score of the sample, between 1 and 5, inclusive. The learning rate was initialized to be 0.01. We used the ReduceLROnPlateau [10] learning rate scheduler to modify the learning rate over time, reducing it with a patience of 8 epochs for which the learning rate did not decrease by more than 10e-3. Finally, we chose the Adam optimizer for its popularity and robustness.

3.2.8 RNN with GRU For Regression Result

Initially, the GRU was found not to be effective at regressing to a RIFL value. Due to the unevenly distributed nature of the data, it consistently learned to minimize its loss by picking a value close to the center of the RIFL distribution (about 3.4), which was very close to the average value of the dataset. In doing so, it basically ignored the specific feature data of the sample it was making the prediction for. To combat this, an artificial weighting scheme was implemented in which samples were bucketed into 5 categories, and weighted in the loss function inversely proportionally to their frequency in the training data. Re-training with this change, the model achieved a relatively high accuracy on the training set with little improvement from random on the validation set. The result showed that with this amount of training data, training using features extracted through sentence embeddings sequentially, it was unlikely that we would be able to train a valid regression model.

3.3 Bucketed Label Classification On Transcripts

3.3.1 Explanation and Justification for Approach

Following our low validation accuracy in solving the continuous regression problem over the integer RIFL labels, we opted to reformulate the targets as bucketed classes, and attempt classification. A positive result in the bucketed case would satisfy our research objective as long as the buckets were meaningful, and could be used to effectively communicate samples that were distinctly problematic, very good, or somewhere along the average. We collaborated with the researchers at OISE to determine a bucketing split of [0, 2.5), [2.5, 4.0), and [4.0, 5.0]. Based on the distribution of the dataset and the real-world interpretation of RIFL scores, these would be sufficient buckets to derive meaningful information from new samples.

3.3.2 Data Pre-Processing

Data pre-processing for the bucketed case was simple. We were able to take the features as extracted for the continuous case and re-label the values as 0, 1, or 2, to match the RIFL values' corresponding bucket.

3.3.3 SMOTE

For the following Logistic Regression and XGB Classifier classification methods, the input data was the vector embedding resulting from passing the original transcript sample through xlm-roberta-base. Now with bucketed labels, we were able to use SMOTE to artificially generate more data points for the minority classes and make up for the poorly distributed data, where the "high" RIFL label and "low" RIFL label were underrepresented as compared to the "Middle" RIFL label category. SMOTE generates new features in a given category by linearly interpolating between two points in that class from the original dataset. Unfortunately, there is no simple analog to SMOTE in the sequential data case, which is the data format for our GRU. Therefore, we were unable to artificially generate additional samples to improve the efficacy of the ML model.

3.3.4 Logistic Regression

The basic logistic regression algorithm on the SMOTE'd transcript data was not particularly effective. It achieved a 64% validation accuracy on a relatively small validation set, but still learned to classify the majority of samples in the centre class, and struggled to detect outliers as shown in Figures 6 and 7.

3.3.5 XGB Classifier

The Extreme Gradient Boosted Classifier method uses gradient boosting and ensemble decision trees to classify data. XGB methods are historically performant in machine learning. On the SMOTE'd transcript data, XGB worked slightly better than Logistic Regression, achieving a validation accuracy of about 68% on the small validation dataset. XGB too, however, tended to regress to classifying in the center label, skewing the distribution of the prediction as shown in Figure 8.

3.3.6 Reformulation to RNN with GRU for Classification

We were able to use a GRU for classification as in the regression case. The architecture and hyperparameters were adjusted as we fine-tuned the model for the discrete

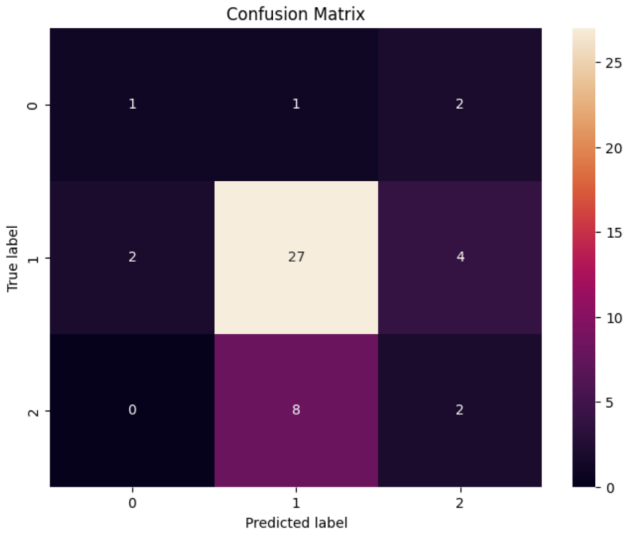
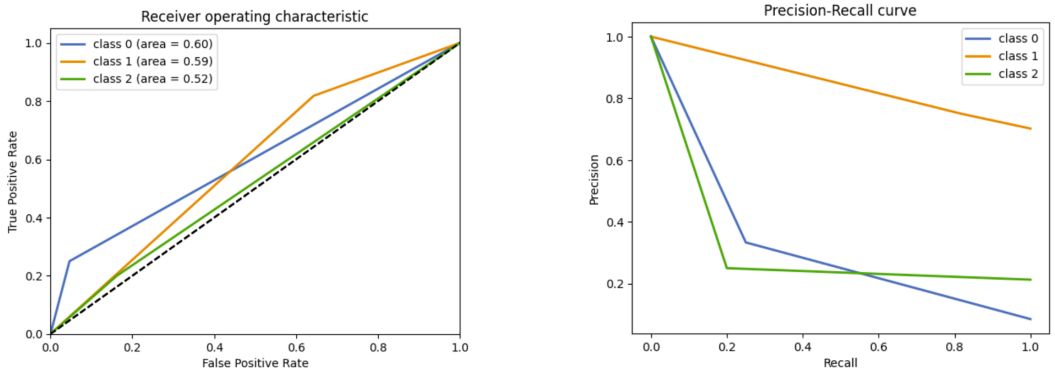


Figure 6: Confusion Matrix for Logistic Regression on Transcript Data



(a) ROC Curve

(b) Precision Recall Curve

Figure 7: Receiver Operating Characteristic and Precision-Recall Curve for Logistic Regression on Transcript Data

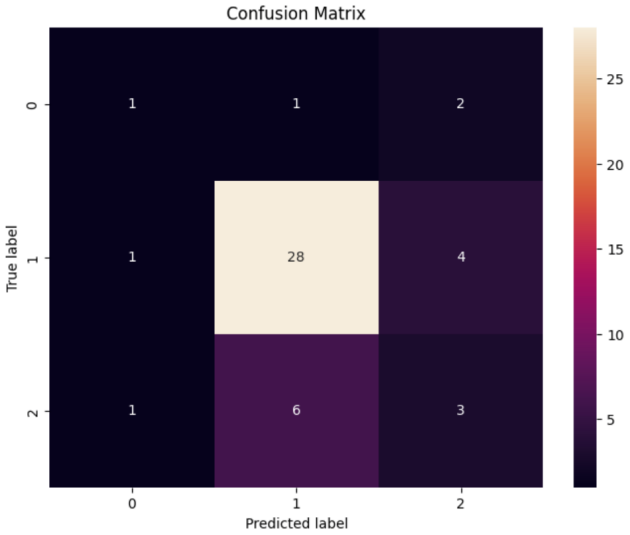


Figure 8: Confusion Matrix for XGBoost Classifier on Bucketed Transcript Data

classification task. We used 2-layer bi-directional GRU with a hidden dimension of 768, followed by a single linear layer with an output dimension of 3, to match the number of classes. The softmax function was not used at output because it is performed at the Cross Entropy Loss stage. Dropout was not used; any amount of dropout seemed to strongly affect the training process. Because we used Cross Entropy Loss and had reframed the task as a classification problem, we were able to organically reweight the classes inversely proportionally to their actual frequency in the training set. This forced the model to treat the classes pseudo-equally and prevented it from honing in on consistently predicting the middle RIFL score class. It was effective as a substitute for SMOTE or similar techniques. Again we used the Adam Optimizer and ReduceLROnPlateau. We started the learning rate at 0.1, but found that it decayed quickly.

3.3.7 RNN with GRU for Classification Result

After training was complete, the GRU achieved a relatively low validation accuracy of about 45.7% on the small validation dataset. Although we had succeeded in preventing the GRU from overpredicting the majority class, it struggled to differentiate samples and generalize to the validation set, especially in the medium and high RIFL domains as shown in Figure 9.

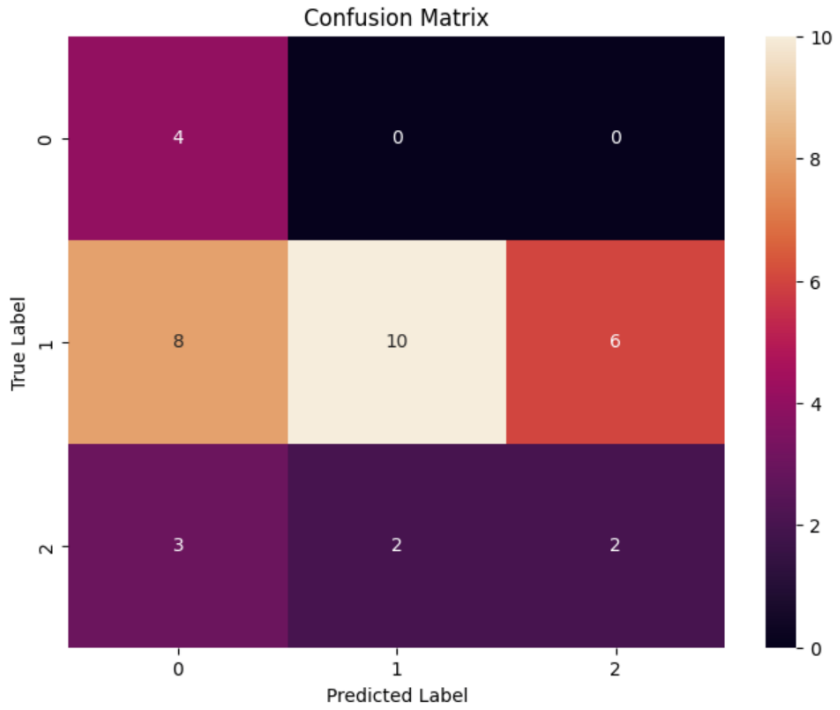


Figure 9: Confusion Matrix for GRU Classifier on Bucketed Transcript Data

3.3.8 Migration to Transformer for Transcript Classification

Following the poor result for the GRU over the validation dataset, we transitioned to a transformer-encoder architecture. Transformers were introduced in the seminal 2017 paper "Attention is All You Need," by Vaswani et al. [11], and revolutionized natural language processing by enabling models to process sequences in parallel, unlike the sequential processing in recurrent architectures. Transformers enable us to better capture complex dependencies and contextual information from the input data using self-attention mechanisms. This architecture is known to generally outperform RNNs (including GRUs) in most tasks by efficiently handling long-range dependencies. This applies especially to datasets that contain long sequences, which is true in our case.

We chose to migrate to a transformer-encoder architecture for this task to achieve a better test and validation accuracy, with the hope that the model gained a better intrinsic understanding of the long-term dependencies in the sequential data. The encoder outputs an encoded sequence of data from which we extract the CLS token and use it to make the class prediction.

3.3.9 Transformer for Classification on Bucketed Transcript Data Training

The transformer-encoder architecture is as follows. A learnable CLS token is concatenated on the beginning of the sequence, and learnable padding parameters are concatenated on the end to regularize the batch sequence length. The input sequence is then encoded using learned positional embeddings to give the model a sense of relative positioning for the sequence of input features. The transformer input dimension is 512. There are 4 attention heads and 4 transformer layers. The encoded CLS token is connected to a single linear layer which outputs three softmax logits corresponding to the three output classes. The data is pre-scaled using a standard scaler. For text classification, a dropout as high as 50% was found to positively contribute to the validation accuracy. A relatively small learning rate of 10e-5 was found to be an optimal starting point. Cosine Annealing was used to schedule the learning rate, where T-max was set to be equal to the number of epochs. Cross Entropy Loss was chosen for the loss function, and the Adam Optimizer was again chosen as an optimizer. As before, we also used class weighting in the loss function proportional to the frequency of each class in the training data to learn a balanced representation of the imbalanced dataset. A batch size of 64 was chosen, and training was performed in 300 epochs.

3.3.10 Transformer for Classification on Bucketed Transcript Data Result

The transformer was able to sufficiently fit itself to the training data, reaching a training loss of about 0.2, and a training accuracy of 100%. After adding dropout iteratively to raise the validation accuracy, the peak training accuracy was about 90%. The validation loss reached its minimum at about 1.0, and the peak validation accuracy was about 70%. The final validation accuracy settled around 67%. The result can be interpreted as classic overfitting. Although the transformer was able to gain an understanding of indicative features in the training set, they didn't seem to effectively generalize to the validation set as shown in Figures 10 and 11.

3.3.11 Transformer for Classification on Bucketed Transcript Data Results

The clear evidence of overfitting means one of two things. First, it could mean that there is no function (linear or nonlinear) that can achieve a high prediction accuracy on sequential text embeddings generated from parent-child interaction audio, because the data simply isn't meaningful. The second option is that the training set contained insufficient data to provide good generalization to unseen

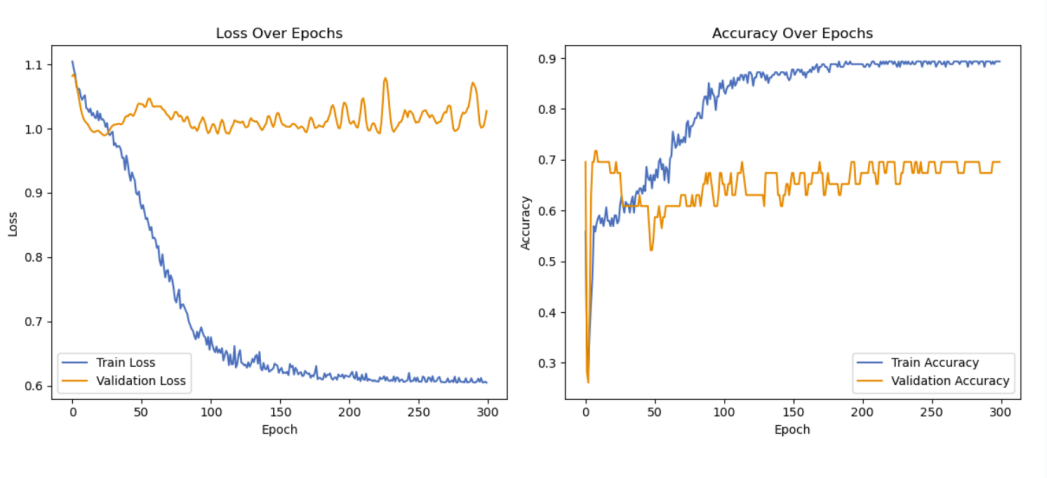


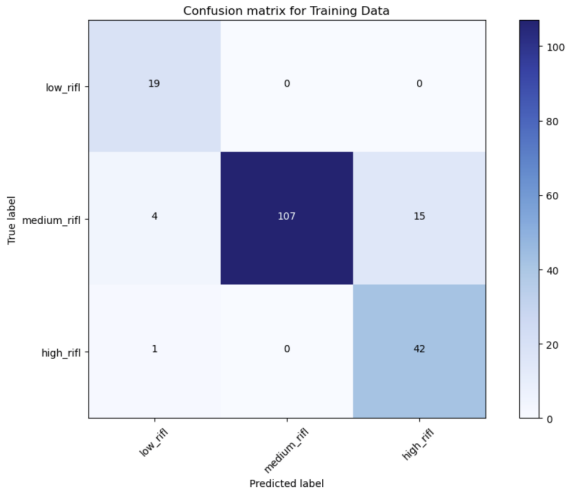
Figure 10: Training and Validation Loss and Accuracy for Transform Classifier for Bucketed Transcript Data

data. Finally, it may be a combination of both; if the features in the training set contained fewer pieces of meaningful information than a different representation of the features or utilization of media from the audio-video data, we may be able to extract features from the data that allow us to generalize to the validation set with fewer data samples, where the sequential transcript embedding data did not. With this in mind, we turn to audio-based classification.

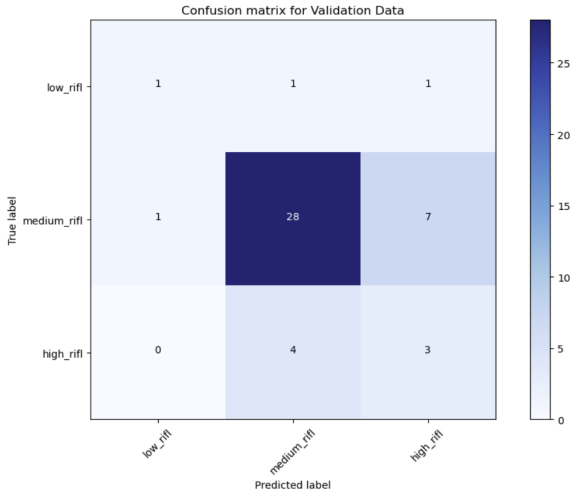
3.4 Audio-Based Classification

3.4.1 Explanation and Justification For Approach

Because our data comprised text transcripts, audio, and video data, the natural next step was to use the audio data as features for our analysis. The method we chose was heavily influenced by the CLASS paper [5], which achieved a reasonable result by compiling distinct audio features over short time slices and processing these as sequential data. It makes intuitive sense that audio data may carry some latent information that text transcripts do not. Humans, after all, communicate as through tone and inflection in addition words themselves. There is an argument to be made that this is even more true for children and parent-child interactions than other types of social interactions.



(a) Prediction on Training Data



(b) Prediction on Validation Data

Figure 11: Confusion Matrices for Transformer Classifier Trained on Bucketed Transcript Training Data

3.4.2 Audio Feature Extraction

First, the audio files were converted from .mp3 and .m4a format to .WAV format to be interpretable for audio analysis. The library we chose was PyAudioAnalysis [6], which included several pre-built functions for extracting time-slice audio features from audio data. We opted to experiment on both short-term features and mid-term features to see what worked best. In the case of short-term feature vectors, the audio was represented as an array of feature vectors taken over a window-size of 100ms and a corresponding step size of 50ms. For example, a 5 minute video would be converted into 3000 feature vectors. We also tried building short-term vectors with window sizes of 50ms and 25ms, but these turned out to be prohibitively difficult to train on our available hardware, as they resulted in sequences that were tens of thousands of features long. The features themselves comprised various classical audio features such as Zero Crossing Rate, Energy, Spectral Centroid, etc., and are detailed in Table 1.

The audio representation using medium-term feature vectors was similar. However, instead of building the feature vectors from the audio features themselves, the feature vectors were constructed from feature statistics over several short-term feature vectors for each mid-term window. We chose the mid-term windows and sub-windows steps each to be 1 second and the short-term vectors to be taken over windows of 50ms with a sub-window step size of 25ms. Therefore each mid-term window comprised statistics for a set of 20 short-term vectors. The statistics were averages, standard deviations, medians, and other operations over the short-term features (ex. average Zero Crossing Rate).

3.4.3 Audio RNN with GRU Regression

We started by using our original sequence-to-regression GRU code to make a RIFL score prediction based on the Audio Features. We suffered again from our imbalanced dataset and our inability to organically account for the proportionally small tails. Our model once again tended to predict an similar value for each sample, about 3.5, regardless of the actual input.

3.4.4 Bucketed Audio RNN with GRU Classification

We transitioned to classification for audio feature data with labels that had been bucketed into three categories, as before. We started again with the GRU model, following the technique of using a sequential model for audio as in the CLASS paper.

Index	Name	Description
1	Zero Crossing Rate	The rate of sign-changes of the signal during the duration of a particular frame.
2	Energy	The sum of squares of the signal values, normalized by the respective frame length.
3	Entropy of Energy	The entropy of sub-frames' normalized energies. It can be interpreted as a measure of abrupt changes.
4	Spectral Centroid	The center of gravity of the spectrum.
5	Spectral Spread	The second central moment of the spectrum.
6	Spectral Entropy	Entropy of the normalized spectral energies for a set of sub-frames.
7	Spectral Flux	The squared difference between the normalized magnitudes of the spectra of the two successive frames.
8	Spectral Rolloff	The frequency below which 90% of the magnitude distribution of the spectrum is concentrated.
9–21	MFCCs	Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
22–33	Chroma Vector	A 12-element representation of the spectral energy where the bins represent the 12 equal-tempered pitch classes of western-type music (semitone spacing).
34	Chroma Deviation	The standard deviation of the 12 chroma coefficients.

Table 1: Audio Features for Short-Term Window [1]

Unfortunately, the GRU failed to sufficiently train on the dataset, likely unable to capture the long-term dependencies contained in the audio features. The audio feature sequence lengths ranged from hundreds in the case of the medium-term feature vectors to thousands in the case of our short-term feature vectors. This sequence length tends to be too long for sequential models, so it was not a surprising result. The CLASS audio samples were seconds long compared to our multi-minute data, which may account for the disparity between results.

3.4.5 Transformer Encoder Classification for Medium-Term Audio Feature Data

Finally, we opted again for the transformer encoder model, as it has greater proficiency for learning long-term dependencies in sequential data. We used the same transformer encoder as described for the transcript data application. During training, the transformer was less receptive to dropout than it was using the transcript feature set. Any dropout over 0 negatively impacted the models ability to fit the training set and did not positively influence the validation set, so dropout was not used when training the transformer on the audio feature data. The training results for the medium-term audio feature data are shown in Figures 12 and 13.

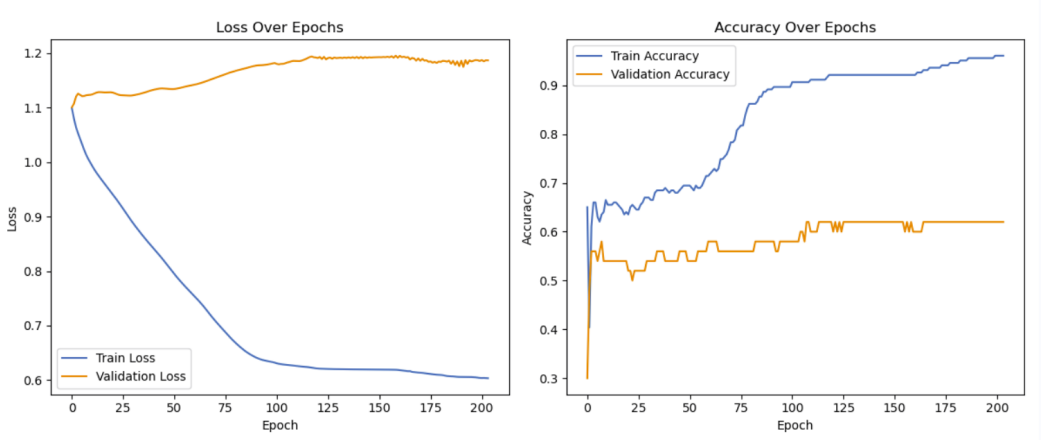
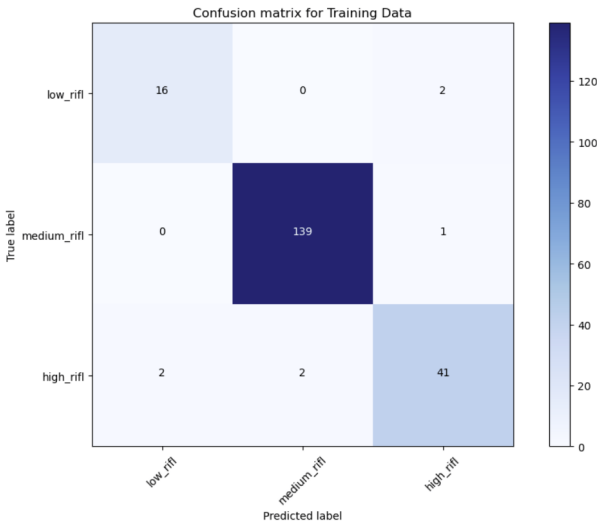


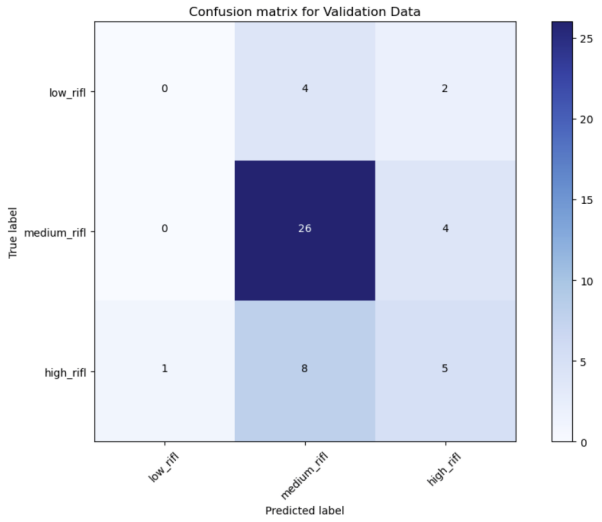
Figure 12: Training and Validation Loss and Accuracy for Transform Classifier for Bucketed Medium-Term Audio Feature Data

3.4.6 Transformer Encoder Classification for Short-Term Audio Feature Data

Similarly, we trained the transformer on the short-term audio feature data. The hyperparameters were not changed during this training process. The result is shown in Figures 14 and 15.



(a) Prediction on Training Data



(b) Prediction on Validation Data

Figure 13: Confusion Matrix for Transformer Classifier Trained on Bucketed Medium-Term Training Data

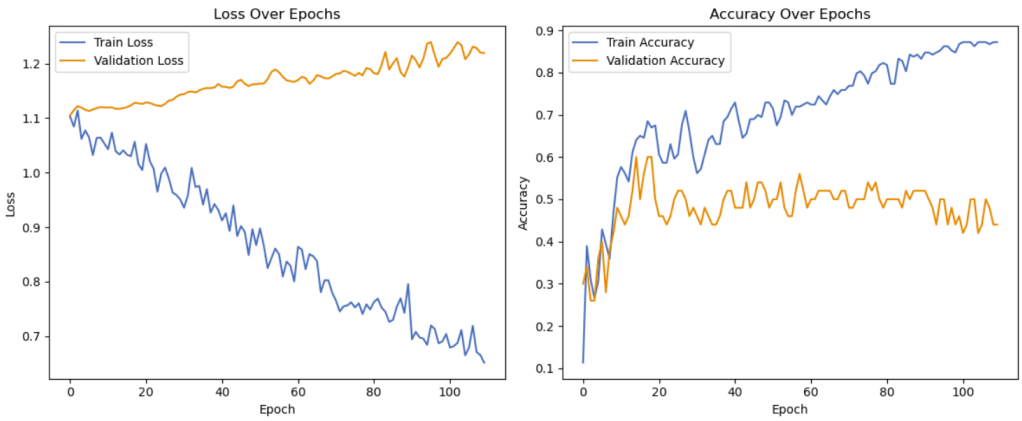


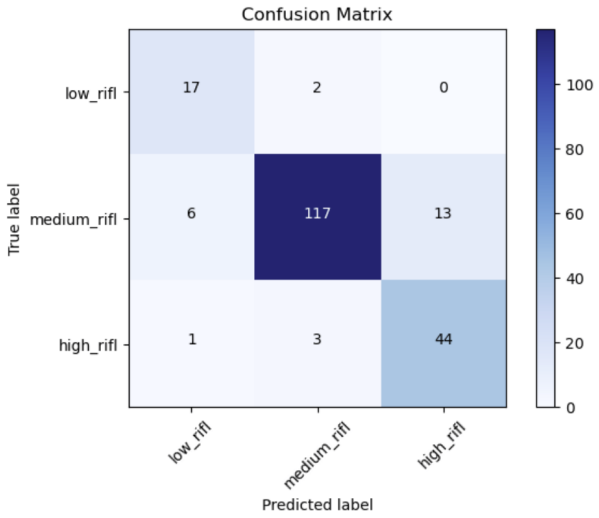
Figure 14: Training and Validation Loss and Accuracy for Transform Classifier for Bucketed Short-Term Audio Feature Data

Although the model was able to adequately fit to the training samples, we again observe what looks to be textbook overfitting with respect to the validation set. The model seems especially reluctant to classify any validation samples into the low-RIFL class which, despite the small validation dataset, might indicate that the meaningful low-RIFL features in the training set simply are not present in the low-RIFL validation samples. Because our transformer so effectively fits the data, the issue is probably an undersized dataset for the complexity of the information the features are attempting to capture. This result indicates that a larger dataset may enable us to train a high accuracy transformer encoder model to classify RIFL scores into discrete categories.

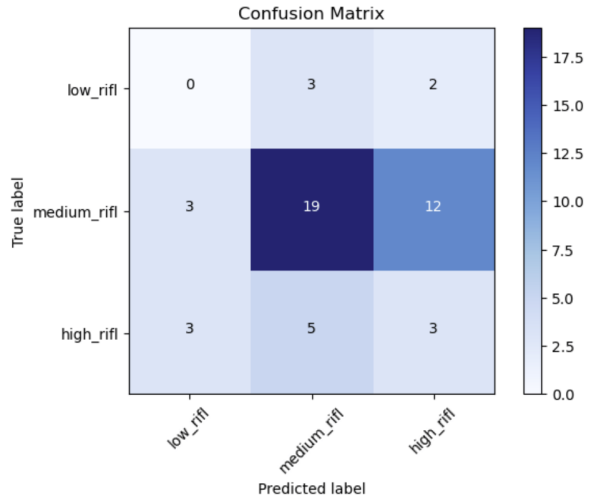
3.5 ROCKET Classification for Short-Term Audio Feature Data

3.6 Explanation of ROCKET Method

ROCKET (RandOm Convolutional KErnel Transform) is a machine learning method specifically designed for time-series classification. It takes a completely different approach from traditional time-series classification methods, which tend to rely on complex and computationally expensive models like the Transformers and GRUs in the previous sections. ROCKET transforms time-series data in such a way that it becomes amenable to linear classifiers, which are typically much faster than other models used for time-series data. To transform the data, ROCKET first generates a set of random convolutional kernels to convolve the sequential data. These kernels have varying lengths, weights, biases, and dilation factors. We are using the default



(a) Prediction on Training Data



(b) Prediction on Validation Data

Figure 15: Confusion Matrix for Transformer Classifier Trained on Bucketed Short-Term Training Data

number, 1000 kernels. The output of the kernel convolution over our original sequence of features are transformed feature maps that can be linearly classified using straightforward methods like Logistic Regression or an XGB Classifier. The intuition is that the kernels randomly map the time-series data to high-dimensional vectors in a way that may contain meaningful separations of data that are easier to learn for a classification algorithm.

3.6.1 Data Preparation for ROCKET Classification

We use the original time series data as an input to the ROCKET algorithm. SMOTE is used on the resulting convolved features in order to generate more samples for the minority classes. An XGB Classifier was found to be most performant at classifying the generated features. We used a gamma of 15, the multi:softprob objective, and weighted samples to account for our imbalanced dataset.

3.6.2 Results of ROCKET Classification

The ROCKET transformation with XGB Classification scored similarly to the transformer. The training accuracy was close to 100% and could be forced to a perfect 100% by adjusting the gamma value. For the medium-term feature vectors, the validation accuracy was 64.1%. For the short-term feature vectors, the validation accuracy was 60.9%. This is not a great result, but it is interesting that the ROCKET was similarly performant to the transformer on the audio data. With sufficient data to train the transformer, it would be interesting to see if the ROCKET achieves a similar accuracy.

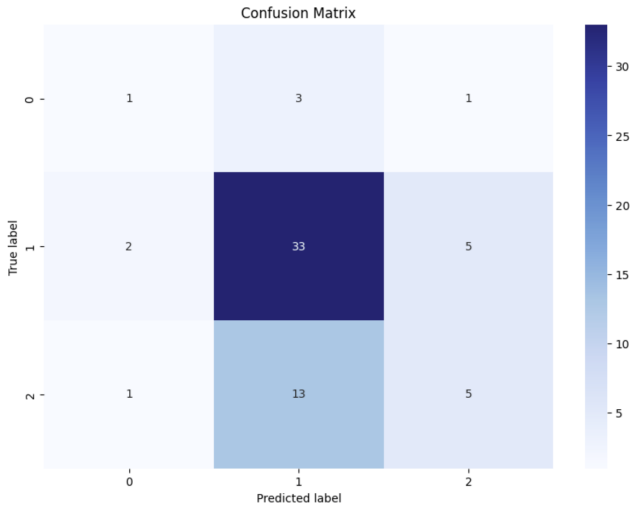
The confusion matrices for the validation performance on the medium-term and short-term validation data are shown in Figure 16

4 Results and Future Work

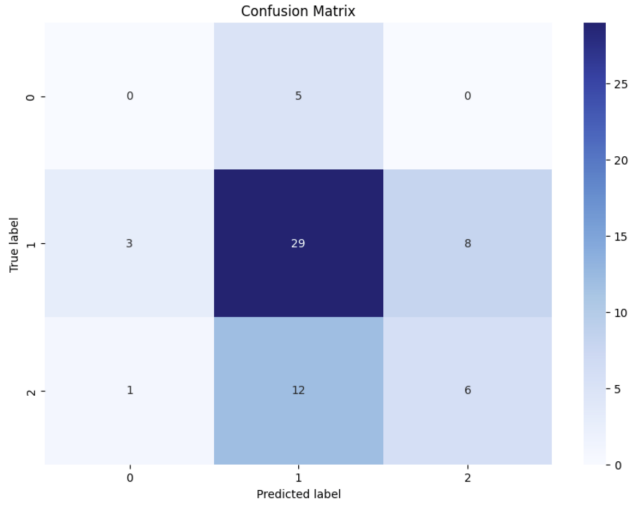
In this section we discuss the meaning and implication of the results from the transcript and audio domains. We also discuss pertinent future work based on our results and limitations throughout the project.

4.1 Transcript Result

The transformer trained on the transcript data embedded line by line using xlm-roberta achieved the best result of any model at 70% validation accuracy, with misclassifications relatively evenly distributed throughout the three classes. However,



(a) Prediction on Medium-Term Validation Data



(b) Prediction on Short-Term Validation Data

Figure 16: Confusion Matrices for ROCKET Transformation and XGB Classifier for Bucketed Medium-Term and Short-Term Audio Validation Data

the model fails to generalize well to the unseen data in the validation and testing sets. The training loss and accuracy values over the training cycle indicate clearly that the model has overfit the training data. The model learns to fit the training features relatively quickly and efficiently. The validation accuracy and loss, however, plateaus. This may indicate that the features are meaningful and learnable, although the dataset is too small for the model to learn to generalize.

4.2 Audio Result

The transformer trained on the audio data as represented by their short- and medium-term feature vectors also performs well relative to other methods, each achieving a maximum of about 60% validation accuracy, with misclassifications relatively evenly distributed among the three classes. The short term feature set was able to overfit more than the medium term feature set. This makes sense intuitively, as the short term sequences were much longer and contained less compressed information than the medium term feature set. The model would have had more opportunity to isolate more components of sequences for specific samples or sets of samples that did not generalize.

It is interesting that the model performed better when trained on the transcript sequence than either audio sequence. An intuition for this is that while the audio may have had more precise and separable features given sufficient data, the features from the transcription which were extracted using a pre-trained model may have been more meaningful out of the box. When training with a small dataset, this could help prevent overfitting and help the model correctly predict the "easy" ones. I hypothesize that, in accordance with the CLASS paper [5], the audio features would provide higher accuracy and better generalization if the dataset was larger. Within our small data context, the sequential transcript embeddings reign supreme.

4.3 Future Work

There are several avenues that would be interesting to pursue in the continuation of this project. First, although we were unable to experiment with video due to time constraints, using the video component of the recordings to improve the features would be a simple next step. One option, which mirrors the CLASS paper [5], would be to train a separate video model and simply ensemble the result with the audio or transcript model to produce a final prediction. The video model could take advantage of pre-trained models like VGG-16 or automatic facial classifiers to derive its result.

Given enough data, a randomly initialized CNN could also be trained. Other options in this pathway include approximating the video as an average of frames, or using a recurrent architecture to classify a sequence of features extracted from a set of frames. Another option would be to integrate the audio and video models into one model that receives a composite set of features from both media pathways. Although the video data might provide more meaningful features for the network, with the current size of the dataset and given the result of the transcript and audio models, it appears unlikely that more complex features and model-training is going to greatly improve the validation and testing accuracy.

The simplest and most logical continuation of the project is to gather more data. More importantly, more evenly distributed data among all possible RIFL scores, although difficult to target during collection, is crucial. The limited dataset and poor distribution of RIFL values through the data were the two greatest impediments to the success of our models during experimentation.

The ROCKET result will also provide a useful proxy to gather an intuition of whether a larger data set will provide an accuracy improvement on unseen data. The ROCKET features can be trained on a very quick model like logistic regression or an XGB Classifier. Additionally, the ROCKET technique paired with an XGB Classifier was clearly sufficient to overtrain the small dataset and achieve similar validation accuracy to the transformer on unseen data. Therefore, given a larger, fresh set of unseen data, one could use ROCKET to quickly train a model and verify whether it looks like a promising avenue. Then, one could follow up by training a large, compute-heavy model.

Another promising research avenue would be to harness the power of large language models in few-shot classification or regression for RIFL scores. Due to privacy constraints on our data and limitations in accessible hardware for the project, we were unable to take advantage of LLM API's or open-source LLMs to experiment with this. However, the state-of-the-art few-shot capabilities of LLMs make it seem reasonable that they might perform well at the RIFL classification task, as humans are. The RIFL criteria may also be included in the prompt to improve the performance or increase the context. Unfortunately, only the transcribed text features could be functionally used as an input to an LLM. This may, however, be sufficient.

5 Conclusion

We started with the following research question: *To what degree of accuracy can we predict or classify RIFL scores using multimodal supervised ML via text transcript or audio mediums*? By exhaustively experimenting with various mediums, models, and features, we can partially answer this question. RIFL scores cannot be predicted or classified to any reasonable degree of accuracy using text transcripts or audio for the dataset provided. This does not mean that it is impossible to classify RIFL scores to a high degree of accuracy. It doesn't even mean that other techniques such as LLMs or the addition of video features wouldn't improve the accuracy. We can say, however, that with this amount of data, distributed in this manner, state-of-the-art supervised learning or related techniques on various audio- or text-derived features are insufficient to train a model that generalizes well for the task of predicting or classifying RIFL scores in parent-child interactions.

References

[1] Theodoros Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PLoS ONE*, 10(12):e0144610, 2015. doi: 10.1371/journal.pone.0144610. URL https://doi.org/10.1371/journal.pone.0144610.

[2] Samantha Burns, Christine Barron, Sumayya Saleem, Calpanaa Jegatheeswaran, Jennifer Jenkins, and Michal Perlman. Examining interactions between educators and across children: Evaluating the validity of the responsive interactions for learning - educator-child dyad version. *Early Childhood Research Quarterly*, 62:405–416, 2023. ISSN 0885-2006. doi: https://doi.org/10.1016/j.ecresq.2022.10.002. URL https://www.sciencedirect.com/science/article/pii/S0885200622001089.

[3] Michelle Rodrigues Sahar Borairi Jennifer M. Jenkins Nina Sokolovic, Ashley Brunsek and Michal Perlman. Assessing quality quickly: Validation of the responsive interactions for learning - educator (rifl-ed.) measure. *Early Education and Development*, 33(6):1061–1076, 2022. doi: 10.1080/10409289.2021.1922851. URL https://doi.org/10.1080/10409289.2021.1922851.

[4] Michal Perlman, Olesya Falenchuk, Brooke Fletcher, Evelyn McMullen, Joseph Beyene, and Prakesh Shah. A systematic review and meta-analysis of a measure of staff/child interaction quality (the classroom assessment scoring system) in early childhood education and care settings and child outcomes. *PLOS ONE*, 11:e0167660, 12 2016. doi: 10.1371/journal.pone.0167660.

[5] Anand Ramakrishnan, Brian Zylich, Erin Ottmar, Jennifer LoCasale-Crouch, and Jacob Whitehill. Toward automated classroom observation: Multimodal machine learning to estimate class positive climate and negative climate. *IEEE Transactions on Affective Computing*, 14(1):664–679, January 2023. ISSN 2371-9850. doi: 10.1109/taffc.2021.3059209. URL http://dx.doi.org/10.1109/TAFFC.2021.3059209.

[6] Theodoros Giannakopoulos. pyaudioanalysis: An open-source python library for audio signal analysis. *PLoS ONE*, 10, 2015. URL https://api.semanticscholar.org/CorpusID:21762643.

[7] Hongbo Chen, Eldan Cohen, Dulaney Wilson, and Myrtede Alfred. Improving patient safety event report classification with machine learning and contextual

text representation. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, 67, 10 2023. doi: 10.1177/21695067231193645.

[8] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale, 2020.

[9] Roberto Cahuantzi, Xinye Chen, and Stefan Güttel. *A Comparison of LSTM and GRU Networks for Learning Symbolic Sequences*, page 771–785. Springer Nature Switzerland, 2023. ISBN 9783031379635. doi: 10.1007/978-3-031-37963-5_53. URL `http://dx.doi.org/10.1007/978-3-031-37963-5_53`.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. `https://pytorch.org/`, 2019.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Appendix

A. Link to Repository:

`https://github.com/ChrisMountn/Undergraduate-Thesis`