Many-to-Many Prediction for Effective Modeling of Frequent Label Transitions in Time Series

Alexander Katrompas
amk181@txstate.edu
Texas State University
San Marcos, Texas, USA

Vangelis Metsis
vmetsis@txstate.edu
Texas State University
San Marcos, Texas, USA

ABSTRACT

Time-series classification is vital in health monitoring and human activity recognition, as well as in areas such as financial forecasting, process control, and a wide array of forecasting tasks. Traditional time-series models segment data into windows and assign one label per window, often missing label transitions within those windows. This paper presents a novel many-to-many time-series model and post-processing using hybrid recurrent neural networks with attention mechanisms, which more effectively captures label transitions over traditional many-to-one models. Further, unlike typical other many-to-many models, our approach doesn't require a decoder. Instead, it employs an RNN, generating a label for every input time step. During inference, a weighted voting scheme consolidates overlapping predictions into one label per time step. Experiments show our model remains effective on time-series with sparse label shifts, but particularly excels in detecting frequent transitions. This model is ideal for tasks demanding accurate pinpointing of rapid label changes in time-series data, such as gesture recognition, making it ideal for fast-paced human activity recognition. [1]

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; **Neural networks**; *Supervised learning*; *Learning to rank*; Structured outputs; **Machine learning algorithms**.

KEYWORDS

Machine Learning, Neural Networks, Time-Series, Attention Mechanisms, Self Attention, Recurrent Neural Networks, Deep Learning, Human Activity Recognition

ACM Reference Format:
Alexander Katrompas and Vangelis Metsis. 2024. Many-to-Many Prediction for Effective Modeling of Frequent Label Transitions in Time Series. In *The PErvasive Technologies Related to Assistive Environments (PETRA) conference (PETRA '24), June 26–28, 2024, Crete, Greece.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3652037.3652049

[1]Project source code.

PETRA '24, June 26–28, 2024, Crete, Greece

ACM ISBN 979-8-4007-1760-4/24/06
https://doi.org/10.1145/3652037.3652049

1 INTRODUCTION

Time-series classification, the task of classifying sequential data over time, is indispensable in various assistive technology applications, ranging from human activity recognition to personalized health monitoring to human-machine interaction. These data, often complex and multidimensional, become particularly challenging when considering variations in length and sampling frequencies, resulting in datasets with frequent and irregular label transitions [3, 5, 19].

A common approach for training machine learning models on time-series data is to segment the continuous time-series into windows of fixed or variable size, and consider each window as a labelable sequence of data. Each window contains a set of time-steps, commonly representing a single label [7, 27]. Classical machine learning typically relies on feature engineering, where a set of features describing the properties of the signal are extracted from each window and used by the learning algorithm [1, 24]. On the other hand, deep learning techniques enable direct input of raw measurements into neural networks, bypassing manual feature definition [9, 21].

Deep learning models for sequence modeling typically appear in two forms: an encoder-only form and an encoder-decoder form. In time-series classification applications, the most common task is to map a set of input time steps to one of a known set of class labels. An encoder-only architecture (i.e., many-to-one) is adequate for such tasks and is the common case. The window size and overlap between windows can be tuned during training and inference [12].

Many-to-one time-series modeling approaches often fail to capture class transitions within each window, as they assign a single label to every window. This limitation becomes evident in applications with frequent label transitions, such as gesture recognition from motion tracking data [11, 19]. To address this issue, we introduce a distinctive many-to-many modeling approach, reminiscent of the encoder-decoder architecture, but without the need for a decoder. Each data input window of dimension $R^{T \times F}$, with T as the length of the sequence and F as the number of channels, is assigned a label per time-step instead of a label per window. The label window corresponding to the data window is also of dimension R^{T}, or $R^{T \times C}$ when the classes are one-hot encoded, where C is the number of known classes.

While our many-to-many architecture shares some similarities with encoder-decoders, our approach deviates in that it eschews the decoder entirely. Instead, we rely on a recurrent neural network (RNN) — specifically, a Long Short-Term Memory (LSTM) enhanced with an attention mechanism, as detailed in Section 4. During the training phase, we feed the model with continuous time-series data, segmented into fixed-size windows, each having multiple labels,

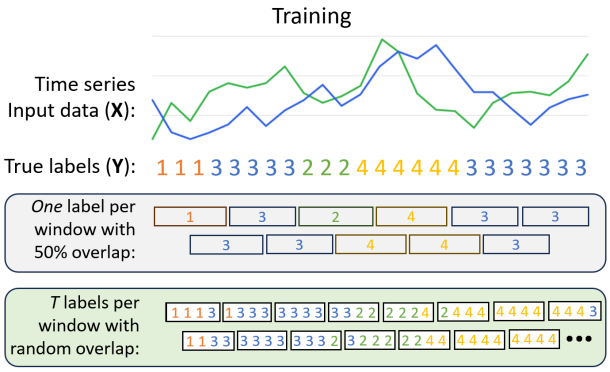

Figure 1: At training time, using the many-to-one approach, the time-series is segmented into (potentially overlapping) windows and a single label is assigned to each window. Using our many-to-many approach, a label is assigned to each time-step of the window.

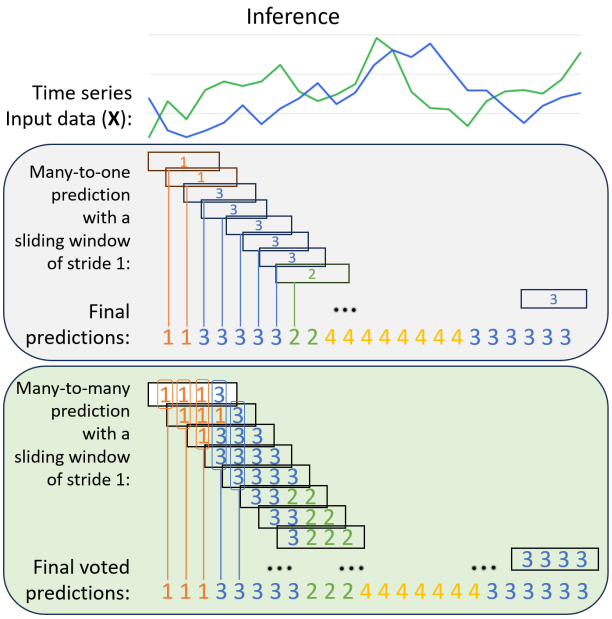

Figure 2: At inference time, using the many-to-one approach, a label per time-step can be predicted using a sliding window with a stride of 1. Using our many-to-many approach, multiple overlapping labels are predicted for each time-step, and a voting scheme is used to consolidate the predictions.

one per time step. This captures diverse class transitions within a window. For inference, we have formulated a weighted voting mechanism, leveraging prediction probabilities and attention scores to refine overlapping predictions into a definitive output for each time step.

Figures 1 and 2 offer a visual contrast between the prevalent many-to-one time-series modeling and our proposed many-to-many paradigm. Parts of the figure with a gray background color depict the common many-to-one approach, whereas the parts with green background represent our proposed many-to-many approach

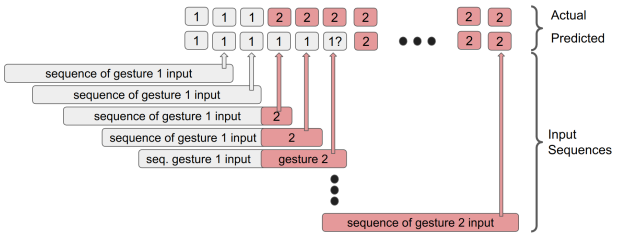

Figure 3: Many-to-one time-series input-output of a hypothetical gesture recognition model. The sliding window fails to correctly predict the transition from gesture 1 to gesture 2.

at training time (1) and at inference time (2). Our method utilizes the LSTM's inherent many-to-many processing capability to yield multiple labels per sequence. After this, as expanded upon in Section 5, a voting mechanism derives a consolidated label for each sequence, in line with the conventional many-to-one prediction paradigm. Furthermore, we introduce a novel weighted training mechanism to handle class imbalance for the many-to-many prediction task.

To encapsulate, this work introduces a novel many-to-many attention-based framework for time-series classification, tailored for frequent label transition scenarios. Our attention-enhanced LSTM coupled with a voting mechanism not only efficiently identifies transitions within data windows but also offers a simpler and more adaptable solution than existing models, as elucidated in Section 2. Empirical results across diverse datasets attest to our model's robust performance, especially excelling in datasets marked by volatility.

2 RELATED WORK

The conventional many-to-one approach to time-series classification is well-documented and broadly adopted [1, 9, 21, 24]. As illustrated in Figure 3, this method demonstrates a limitation in capturing swift label transitions. While the onset of a label change initially remains dominated by its predecessor, the subsequent inputs progressively favor the new label. Despite this, a transitional latency persists [30]. For datasets with infrequent transitions, such delays might be negligible. However, in domains characterized by frequent label changes, like human gesture recognition or financial analytics, this can lead to significant prediction errors [19, 32].

The many-to-many, or sequence-to-sequence, modeling, traditionally employed in tasks like machine translation, remains relatively untapped for label prediction sequences. Even though this paradigm shows promise in terms of accuracy and robustness over its many-to-one counterpart [6, 20, 28], it is predominantly lauded for predicting entire sequences rather than individual labels [4, 33]. Attention-based augmentations have further enhanced its efficiency [15].

Our exploration pivots towards harnessing many-to-many modeling for data exhibiting high label transition rates, a realm often deemed challenging for deep learning. Classical methods necessitate intensive pre-/post-processing or resort to traditional techniques [19, 22, 26, 30]. We postulate that a streamlined many-to-many deep learning approach can adeptly navigate such erratic data with minimal processing overhead.

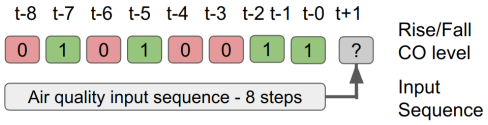

Figure 4: Air Quality Example Sequence.

While there have been ventures into this domain, our approach distinguishes itself in several respects. For instance, while many-to-many models have found applications in medical diagnostics, these models often necessitate significant further post-processing to yield a singular prediction [17]. Others have argued the computational simplicity and comparable accuracy of many-to-many over many-to-one modeling [28]. We advance this claim by demonstrating the superiority of many-to-many models, particularly when supplemented with attention mechanisms and our novel voting process, especially for high volatility datasets. Furthermore, a related study applied many-to-many models to volatile financial datasets [32]. Although it bears similarity to our work, it is tailor-made for autoregressive problems and demands extensive feature engineering. In contrast, our method offers a more versatile solution, negating the need for intricate feature engineering and transcending the confines of autoregression.

3 DATA

In time-series modeling, the continuous data are commonly segmented into windows of multiple time-steps. Each window forms a training instance. The data characteristics of interest to this work are high label transition rate time-series data, defined as data for which, within a sequence window, there are two or more labels for numerous such windows. Figure 4 shows an example of such a sequence from the air quality data used in this study. As shown, the optimal window size for modeling this type of data is 8 time-steps (derived through grid-search) and has within it multiple labels, making the prediction of t+1 more difficult. The following data provide such data for the study, as well as being sufficiently disparate to demonstrate the generalizability of the technique and results. Three data sets are presented, used across four prediction/classification tasks. Data sets are listed in order of the percentage of sequences (windows) which contain multiple labels per sequence.

- *Hand Gestures*: Predicting gesture phases (rest, gesture, retraction) [19]. Multi-label prediction. 67% transition sequences. Sequence length: 44.
- *Air Quality*: Predicting CO levels will rise/fall within 24 hours [5]. Binary prediction. 84% transition sequences. Sequence length: 8.
- *Metro Interstate Traffic Volume*: Binary prediction traffic will increase/decrease in the next hour [10]. 99% transition sequences. Sequence length: 12.
- *Metro Interstate Traffic Volume*: Multi-label classification traffic by very-low, low, medium, high, very-high volume [10].

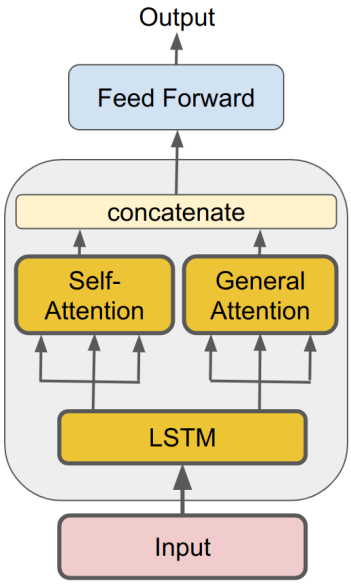

Figure 5: LSTM with Temporal Attention Model.

99% transition sequences. Sequence length: 12.

4 MODEL

4.1 Rational and Architecture

The proposed model (see Figure 5) is the culmination of a series of approaches to deep learning time-series modeling, each facilitating superior performance in time-series prediction [13]. Firstly, our backbone of choice is the LSTM, historically shown to produce state-of-the-art results in time-series modeling [17, 34]. Attention mechanisms, shown to improve LSTM performance, are added in two forms for two purposes. Self-attention has been shown to generally improve time-series accuracy [12, 23, 34]. General attention is uniquely suited for detailed input-output analysis, which is of direct interest to this work. In particular, the general attention layer can be extracted and analyzed, providing insight as to what the model finds important during classification [13].

The LSTM architecture inherently produces a many-to-many output, which when combined with general attention, gives us a multistep output suitable for processing with our unique voting process, ultimately producing a single output label with higher accuracy as compared to many-to-one time-series modeling. This is accomplished by leveraging the fact that the general attention layer is weighting the input-output relationship, stressing the more relevant time-steps for the desired output [13]. By processing the multistep output, as modified by attention, we should expect that we can find the "important" inputs-outputs within the sequence relative to the desired output(s).

For example, consider Figure 4, and assume the ?-output is 0 (i.e. falling CO). Even though 50% of the time-steps represent the input associated with the opposite case (i.e. rising CO), temporal attention allows the model to stress the importance of label 0 (in this sequence) in order to correctly predict 0 [13]. Given this behavior,

we should be able to analyze the output activations in a many-to-many model and build a more accurate and higher confidence model (see Section 5).

General Attention equations as implemented within this work.

$$e_i = tanh(W_a H^T + b_a) \quad [\text{ similarity score }] \tag{1}$$

$$a_i = softmax(e_i) \quad [\text{ attention weights }] \tag{2}$$

$$y = \Sigma_i a_i H^T \quad [\text{ output vector }] \tag{3}$$

Self-Attention equations as implemented within this work.

$$E = tanh(W_a H^T + b_a) \quad [\text{ similarity score }] \tag{4}$$

$$A = softmax(W_e E) \quad [\text{ attention weights }] \tag{5}$$

$$Y = AH \quad [\text{ output matrix }] \tag{6}$$

4.2 Temporal Attention

Originally designed for text-centric sequence-to-sequence models, attention mechanisms allow a network to emphasize specific segments of an input sequence, thus refining sequence representation [2, 18]. Essentially, attention offers a contextual weighting, delineating importance within a sequence. Two primary forms, general attention and self-attention, have been integrated into our time-series context, collectively termed as *temporal attention* [2, 29].

General attention creates a context vector, determining the significance of each RNN output step concerning prediction. This vector, in turn, guides the model to focus on pertinent segments of the input sequence [2, 18]. In contrast, self-attention measures sequence segments' significance concerning one another, culminating in a context matrix. This matrix subsequently facilitates better modeling between input and target output [16, 29, 35].

General attention, as applied to time-series, operates by capturing the full LSTM output (i.e., "hidden layer"), typically denoted H_t^T, from within a sequence and training a separate layer to "attend" to some parts of the LSTM output more than others. Equations 1 through 3 define temporal general attention, closely following the accepted global soft-attention [2, 18], where H is the output of the LSTM layer (i.e., input to the attention layer), and x is model input ((i.e., input to the LSTM layer).

Self-attention, as applied to time-series, follows traditional self-attention (typically used with text processing) [16, 29], with minor but notable changes. In temporal self-attention, the "value matrix," a projection of input, is used as-is and not transformed. This is done because in the case of quantitative time-series data, there is no purpose to a transformation of the input data. A second difference is that dot-product attention is used, rather than scaled dot-product. The purpose of scaling is to avoid the vanishing gradient problem, a redundant feature in the presence of an LSTM, which avoids this inherently [21, 34]. Equations 4 through 6 describe temporal self-attention.

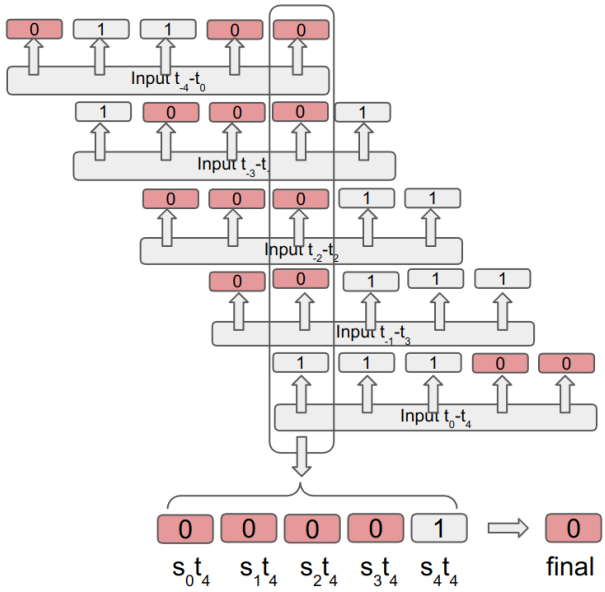

Figure 6: Single Prediction, Hard Vote.

5 EXPERIMENTS

5.1 Methodology

For each dataset, we constructed two distinct models: a conventional many-to-one LSTM model for deep learning time-series and our proposed many-to-many model (refer to Figure 5). The key architectural distinction between these models is the inclusion of a flatten layer in the feed-forward segment of the many-to-one model. This incorporation yields a standard many-to-one LSTM with attention for time-series modeling. However, both models incorporate an attention layer, which has been demonstrated to enhance LSTM time-series performance significantly [13].

The many-to-one model is fed with training data using a time-series generator approach, utilizing a fixed sliding window with a stride of 1. In contrast, the many-to-many model segments data into fixed-size windows, applying a sliding window of random stride ranging from 1 to $T/2$. This approach is designed to encapsulate diverse class transitions. Notably, both models maintain an identical sequence size, denoted as T. We determined the optimal sequence sizes and other hyperparameters through a comprehensive grid search, with specifics available in the accompanying code.

To ensure robustness, each model underwent training and evaluation ten times, leveraging 5-fold cross-validation on the training, test, and validation datasets, using a 70/15/15 split. During inference, both models employed a $T-1$ overlap between successive windows to assess performance across all potential sequences. In the many-to-one model, each sequence yields one label. Conversely, in the many-to-many approach, each sequence results in T labels. These labels are subsequently refined using various weighted voting mechanisms, as illustrated in Figure 6 and detailed in Section 5.2).

5.2 Voting Schemes

The process of generating a single prediction from a many-to-many model, as visualized in Figure 6, involves producing T predictions based on the input sequences ranging from t_{i-T} to t_{i-1}, where i spans from 1 to T. In this context, t_0 is the targeted prediction, which could either predict a future step or classify the current step—both scenarios are explored and analyzed in this paper. This process yields a prediction/classification vector, denoted as $s_{i-1}t_{T-1}$, where each sequence signifies a "vote" for t_0. To amalgamate these votes into a singular prediction, we introduce four voting mechanisms:

(1) *Hard-vote*: Here, the labels (one per time-step) are predicted, tabulated, and the majority label is designated as the prediction (see Figure 6).

(2) *Soft-vote*: Instead of relying on discrete labels, this scheme evaluates the probabilistic prediction for each class. Let p_{ij} be the predicted probability of the i^{th} time-step for the j^{th} class for a given sequence. For soft voting, the predicted probability for the j^{th} class (for a given time-step) is the average of the predicted probabilities for that class across all overlapping sequences. $v_j = \frac{1}{N}\sum_{i=1}^{N} p_{ij}$ for $j = 1, 2, \ldots, M$. The class with the highest average probability after soft voting is selected as the final class of the time-step. Mathematically, this is: prediction = $\arg\max_j(v_j)$

(3) *Attention-vote*: The attention mechanism employed in our model assigns a different attention weight to each time-step of the sequence, indicating the importance of that time-step to the prediction outcome. In the attention-vote, we incorporate the attention weight w in the soft voting process, thus producing a weighted soft voting. Assuming weights w_i for each overlapping sequence are normalized such that $\sum_{i=1}^{N} w_i = 1$, the predicted probability for the j^{th} class using weighted soft voting is: $v_j = \sum_{i=1}^{N} w_i \cdot p_{ij}$ for $j = 1, 2, \ldots, M$. The final prediction after weighted soft voting with normalized weights is: prediction = $\arg\max_j(v_j)$. This mirrors the soft-vote mechanism but incorporates an additional factor: $w = \alpha_{i,j}$, where α represents the attention score of the output i at time-step j.

(4) *Stacking-vote*: A Gradient Boosting Classifier (GBC) is first trained on test set predictions. Subsequently, it's leveraged to transform multi-label outputs into single-label predictions during inference. The GBC operates as a meta-estimator, fitting multiple decision tree classifiers to an input sequence (i.e., the many-to-many network's output) to ultimately predict a singular label.

The intuition behind the four voting processes is simple; hard-voting represents the absolute "opinion" of the T sequence's input-output, which provides a base-line. Soft-voting is a modification of hard-voting, which gives a stronger/weaker vote to neurons that are more/less "sure" of their classification based on the strength of their output. We should expect soft-voting to outperform hard-voting, as it is a relative measure based on the strength of the output. Attention-voting is a modification of soft-voting by weighing the strength of the output neuron by the network's attention scoring. Since we are combining the relative confidence of the output neuron with the network's overall confidence of that input-output, we should expect attention-voting to outperform soft-voting. Stacking is presented as a meta-classifier for comparison purposes, and at the time of the study, we have no preconceived notion as to its performance relative to other schemes.

Table 1: Gesture Data: Multi-label classification, classifying hand gestures into three phases; rest, gesture, retraction.

Gesture	m-to-1	hardv	softv	attnv	stacking
accuracy	0.717	0.787	**0.806**	0.805	0.638
precision	0.732	0.794	**0.810**	0.809	0.652
recall	0.717	0.787	**0.806**	0.802	0.637
f1	0.698	0.781	**0.795**	0.794	0.610

Table 2: Air Quality Data: Binary prediction, predicting CO levels will rise or fall within 24 hours.

Air Quality	m-to-1	hardv	softv	attnv	stacking
accuracy	0.777	0.825	**0.828**	**0.828**	0.801
precision	0.789	**0.844**	0.834	0.834	0.811
recall	0.772	0.824	**0.827**	**0.827**	0.809
f1	0.753	**0.812**	0.811	0.811	0.798

Table 3: Traffic Volume Data: Binary prediction, predicting traffic levels will rise or fall within 1 hour.

Traffic 1	m-to-1	hardv	softv	attnv	stacking
accuracy	0.775	0.787	0.796	**0.799**	0.791
precision	0.774	0.785	0.790	**0.805**	0.800
recall	0.775	0.787	0.792	**0.801**	0.793
f1	0.768	0.786	0.781	**0.793**	0.787

Table 4: Traffic Volume Data: Multi-label classification, classifying current traffic into 5 levels.

Traffic 2	m-to-1	hardv	softv	attnv	stacking
accuracy	0.664	0.775	**0.787**	0.786	0.672
precision	0.661	0.806	**0.808**	0.802	0.668
recall	0.664	0.775	**0.787**	**0.787**	0.670
f1	0.649	0.776	**0.789**	0.788	0.642

5.3 Dealing with Class Imbalance in Many-to-Many Modeling

In the course of obtaining data, it was noted that very often high-transition rate data is also imbalanced data. While the air quality set was balanced, with each class representing near 50% of the data, the other two sets were quite imbalanced. For example, the rest phase of the Gesture data represents less than 20% of the time steps. Similarly, very high and very low traffic in the traffic volume data was less than 10% each. The challenges of imbalanced data are well-documented [8, 14], and typically require some kind of remediation, such as up-/down-sampling, or a weighted loss. These remediations can be readily applied in the common case where

each classification instance (sequence window) is associated with a single label.

In the case of sequences in which each time-step is associated with a label, up-/down-sampling entire sequences does not solve the problem as the labels associated with each time-step will be up-/down-samples by the same rate, keeping the class proportions unchanged. Also, up-/down sampling specific time steps is not possible because they alter the characteristics of the sequence.

To overcome the class imbalance in the case of many-to-many prediction, we compute a custom weight for each sequence. Given an imbalanced multi-class time-series dataset, the algorithm calculates sample weights based on the uniqueness and frequency of class transitions within each sequence instance (window).

Let's denote our multi-class time-series dataset by Y_{train} where $Y_{\text{train}} \in \mathbb{Z}^{n_{\text{sequences}} \times n_{\text{timesteps}}}$ contains sequences with one label per time-step. The goal is to compute weights for each sequence to mitigate class imbalance. The proposed algorithm focuses on computing weights based on the uniqueness and frequency of label transitions in each sequence window. The steps are as follows:

(1) **Input:** Y_{train}

(2) **Identify Transitions:** For each sequence s in Y_{train}, identify label transitions between consecutive time-steps. Let T_s be the set of transitions for sequence s, i.e.,

$$T_s = \{(y_t, y_{t+1}) \mid y_t, y_{t+1} \in s, t = 1, \ldots, n_{\text{timesteps}} - 1\}$$

(3) **Compute Transition Frequencies:** Let F denote the frequency map. For each unique transition t across all sequences, compute its frequency:

$$F(t) = \sum_{s \in Y_{\text{train}}} \mathbb{I}(t \in T_s)$$

where $\mathbb{I}$ is the indicator function.

(4) **Calculate Transition Weights:** The weight of each transition t is the inverse of its frequency:

$$W(t) = \frac{1}{F(t)}$$

(5) **Assign Sequence Weights:** The weight for each sequence s is computed based on the transitions it contains:

$$W_s = \frac{1}{|T_s|} \sum_{t \in T_s} W(t)$$

where $|T_s|$ is the number of transitions in seq. s.

(6) **Output:** Sequence weights $\{W_s\}_{s=1}^{n_{\text{sequences}}}$

These sequence weights can then be used as sample weights when training the model, enabling a weighted loss computation that takes into account the class imbalance.

6 RESULTS AND ANALYSIS

6.1 General Analysis

Upon training, each model undergoes evaluation on its respective test set. The results pertaining to the four datasets, as described in Section 3, are detailed in Tables 2 through 4. These tables enumerate the performance metrics for many-to-one (m-to-1) modeling alongside the various many-to-many voting schemes: hard voting (hardv), soft voting (softv), attention-based voting (atnv), and stacking.

A consistent observation across the datasets is that many-to-many modeling outperforms the many-to-one paradigm across all voting strategies. In alignment with our anticipations, hard voting exhibits superior performance when juxtaposed with the conventional many-to-one time-series model, thus reinforcing the validity of our proposed many-to-many time-series modeling approach. Additionally, soft voting, which accords weighted votes based on the confidence of the output neurons, surpasses hard voting.

Intriguingly, while one might predict enhanced performance from soft voting when integrated with weights extracted from the attention mechanisms, our results indicate that soft voting on its own and weighted soft voting offer comparable outcomes. A meticulous examination of Equations 3 and 6 reveals the underlying rationale: the output is inherently weighted by attention values. Thus, the subsequent application of these weights yields no discernible difference, as the attention scores are intrinsically embedded within the output activations. This insight not only became evident post-analysis, but also underscores the fundamental operation of attention mechanisms, affirming their role in amplifying relevant features or time steps to bolster prediction accuracy.

Lastly, the stacking method, incorporated as a comparative benchmark devoid of any voting mechanism, presented ambiguous results when contrasted with the many-to-one approach and exhibited a consistently subdued performance against the soft-voting many-to-many modeling paradigm. Given the added computational cost compared to voting, stacking seems like a less attractive option.

6.2 Gesture Data Analysis

The gesture data set is selected for detailed examination as it is the lowest transition rate (67%) of the presented high-transition rate data, showing more easily human-identifiable patterns, making for an effective and interpretable demonstration of the efficacy of many-to-many modeling, especially within the realm of human activity recognition. Figures 7 and 8 show actual (red) versus predicted (blue) for the gesture test data, highlighting the transitions from rest (0), to gesture (1), to retraction (2). Figure 7 demonstrates the traditional many-to-one model fails for many of the transitions to and from retraction (2). This results from the relatively short duration of retraction, causing it to be "lost" in the relatively long sequence lengths (44). Conversely, as shown in Figure 8, the many-to-many model effectively captures retraction (2). This is a direct result of the attention mechanisms weighing the retraction-relevant time-steps heavier when retraction is both present and relevant in the sequence.

The gesture data set also provides a comparative study to validate the study results [19]. Previous work has shown this data set

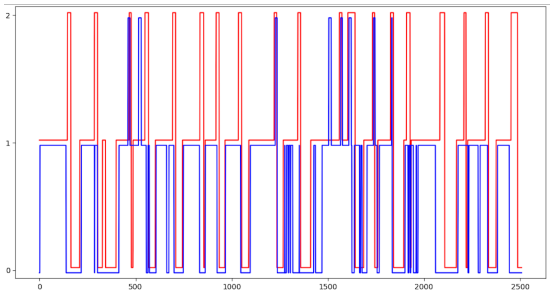

Figure 7: Gesture classification, many-to-one, actual (red) versus predicted (blue).

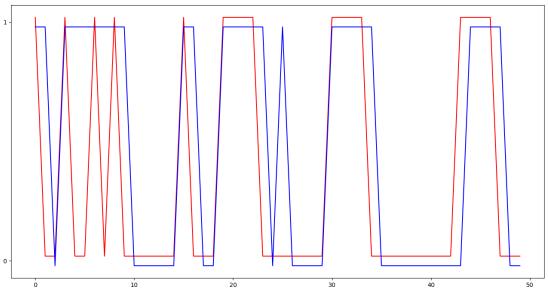

Figure 9: Air Quality, many-to-one, actual (red) versus predicted (blue); random sample of 50 time-steps.

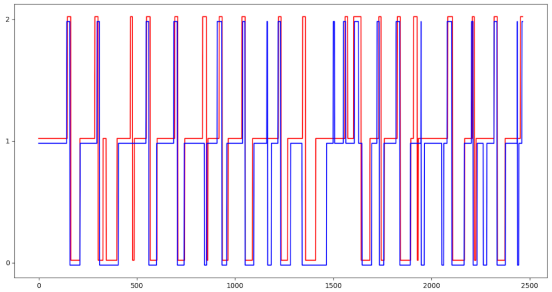

Figure 8: Gesture classification, many-to-many, actual (red) versus predicted (blue), soft vote.

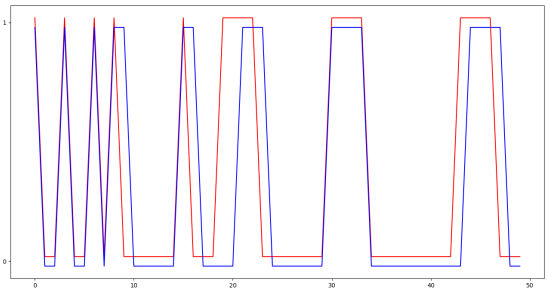

Figure 10: Air Quality, many-to-many, actual (red) versus predicted (blue); random sample of 50 time-steps.

is particularly challenging, and does not lend itself well to multi-label, continuous time-series, deep learning models, despite being continuous time-series data, and this is clearly shown to be due to transition errors. In this previous work, to overcome the transition errors, the original data were greatly simplified into binary classification problems of the form *label/not-label*, such as rest/not-rest. Using Support Vector Machines, averaged classification precision of ~84% was achieved, and that metric was itself compared to previous work, using similar simplifications, and achieving similar results [25, 31]. While theoretically effective, these simplifications, it could be argued, are over-simplifications as they do not yield useful real-time, continuous data insights, they simply prove that transition errors are highly problematic in high-transition rate data.

Through this study, we are able to improve upon previous work by treating the data as continuous and whole time-series data, accurately identifying rest, gesture, and retraction in a continuous stream of data, a far more complex proposition, while achieving comparable precision, ~81%, but with a more practical and useful approach, paving the way for real-time multi-class classification and analysis, something previous approaches could not achieve.

6.3 Performance Analysis in Low Class Transition Rate Data

An examination of Figures 7 and 8 reveals that the many-to-many modeling approach is comparable to the many-to-one model in regions characterized by low label transition rates. Notably, in areas with elevated transition rates, the many-to-many approach exhibits superior performance. In prolonged segments where labels predominantly represent either rest (0) or gesture (1), both modeling paradigms deliver satisfactory and equivalent results. Nonetheless, during rapid transitions, especially evident in swift gesture retractions, the many-to-one model's performance diminishes, as illustrated in Figure 7. In contrast, Figure 8 confirms that the many-to-many model's efficiency remains consistent in regions of infrequent transitions, and it excels remarkably during frequent transitions, outpacing the many-to-one model.

For a more comprehensive understanding, Figures 9 and 10 provide an actual-versus-predicted comparison over a randomly selected window of 50 time-steps for the air quality dataset. Analogous to the gesture dataset observations, these figures underline that the many-to-one model struggles during periods of rapid transitions (early segments of the window shown) but aligns closely with the many-to-many model in segments marked by sporadic transitions (later in the sequence). This reaffirms the premise that the many-to-many modeling approach, while retaining its efficacy in areas of low transition rates, offers a distinct advantage in high-transition-rate scenarios. Additional examinations of the air quality data and traffic volume data, which are not included here for conciseness, corroborate these findings.

7 CONCLUSION

Our research underscores the efficacy of many-to-many time-series modeling, especially for data characterized by high label transition rates, outperforming the conventional many-to-one modeling

approach. This novel enhancement is largely attributed to the capability of many-to-many attention-based modeling to accurately discern transition states, leading to superior metrics in terms of accuracy, precision, recall, and F1 score. It is noteworthy that while excelling in high-transition scenarios, our novel attention-based many-to-many voting model doesn't compromise performance in non-transition data, positioning it as a versatile tool for general time-series classification or prediction, irrespective of the transition rate intensity.

Through our experiments, soft voting emerged as the most impactful mechanism for accuracy enhancement. While attention voting did present some improvements, they were mostly minimal. This observation aligns with both theoretical foundations and intuitive reasoning: the attention scores are implicitly embedded within the output activations in soft-voting. The prominence of soft-voting, seamlessly integrating attention, further validates the potency of attention mechanisms when coupled with RNNs for time-series modeling. The attention mechanism refines output activations to better resonate with the input context.

In closing, our research indicates that integrating sample weighting can further refine time-series models. This approach not only addresses challenges posed by high-transition rate data but also compensates for data imbalances. The culmination of these strategies paves the way for robust and precise modeling of data that is both volatile and skewed.

REFERENCES

[1] Oliver Anderson. 1995. More effective time-series analysis and forecasting. *Journal of Computational and Applied Mathematics - J COMPUT APPL MATH* 64 (11 1995), 117–147. https://doi.org/10.1016/0377-0427(95)00011-9

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *ArXiv* 1409 (09 2014).

[3] Oresti Banos, Claudia Villalonga, Rafael García, Alejandro Saez, Miguel Damas, Juan Holgado-Terriza, Sungyong Lee, Hector Pomares, and Ignacio Rojas. 2015. Design, implementation and validation of a novel open framework for agile development of mobile health applications. *BioMedical Engineering OnLine* 14 (08 2015), S6. https://doi.org/10.1186/1475-925X-14-S2-S6

[4] Ananth Reddy Bhimireddy, Priyanshu Sinha, Bolu Oluwalade, Judy Wawira Gichoya, and Saptarshi Purkayastha. 2020. Blood Glucose Level Prediction as Time-Series Modeling using Sequence-to-Sequence Neural Networks. In *KDH@ECAI*.

[5] Saverio DeVito, Ettore Massera, M Piga, L Martinotto, and G Francia. 2008. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B Chemical* 129 (02 2008), 750–757. https://doi.org/10.1016/j.snb.2007.09.060

[6] Shengdong Du and Horng. 2018. Time Series Forecasting Using Sequence-to-Sequence Deep Learning Framework. In *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*. 171–176. https://doi.org/10.1109/PAAP.2018.00037

[7] Ary Goldberger, Luís Amaral, Leon Glass, Jeffrey Hausdorff, Plamen Ivanov, Roger Mark, Joseph Mietus, George Moody, Chung-Kang Peng, and H. Stanley. 2000. PhysioBank, PhysioToolkit, and PhysioNet : Components of a New Research Resource for Complex Physiologic Signals. *Circulation* 101 (07 2000), E215–20. https://doi.org/10.1161/01.CIR.101.23.e215

[8] Haibo He and Edwardo A. Garcia. 2009. Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering* 21, 9 (2009), 1263–1284. https://doi.org/10.1109/TKDE.2008.239

[9] Hansika Hewamalage. 2020. Recurrent Neural Networks for Time Series Forecasting: Current Status and Future Directions. *International Journal of Forecasting* 37 (08 2020). https://doi.org/10.1016/j.ijforecast.2020.06.008

[10] John Hogue. 2019. Metro Interstate Traffic Volume. https://doi.org/10.24432/C5X60B

[11] Michael Husken and Peter Stagge. 2003. Recurrent neural networks for time series classification. *Neurocomputing* 50 (01 2003), 223–235. https://doi.org/10.1016/S0925-2312(01)00706-8

[12] Alexander Katrompas and Vangelis Metsis. 2022. Recurrence and Self-attention vs the Transformer for Time-Series Classification: A Comparative Study. In *Artificial Intelligence in Medicine*. Springer International Publishing, Cham, 99–109.

[13] Alexander Katrompas and Vangelis Metsis. 2023. Temporal Attention Signatures for Interpretable Time-Series Prediction. Springer Nature.

[14] Bartosz Krawczyk. 2016. Learning from imbalanced data: Open challenges and future directions. *Progress in Artificial Intelligence* 5 (04 2016). https://doi.org/10.1007/s13748-016-0094-0

[15] Yurui Li, Mingjing Du, and Sheng He. 2022. Attention-Based Sequence-to-Sequence Model for Time Series Imputation. *Entropy* 24 (12 2022), 1798. https://doi.org/10.3390/e24121798

[16] Zhouhan Lin, Minwei Feng, Cicero Dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Y. Bengio. 2017. A Structured Self-attentive Sentence Embedding. (03 2017).

[17] Zachary Lipton, David Kale, Charles Elkan, and Randall Wetzel. 2015. Learning to Diagnose with LSTM Recurrent Neural Networks. (11 2015).

[18] Minh-Thang Luong, Hieu Pham, and Christopher Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. (08 2015).

[19] Renata Madeo, Clodoaldo Lima, and Sarajane Peres. 2013. Gesture unit segmentation using support vector machines: Segmenting gestures from rest positions. *Proceedings of the ACM Symposium on Applied Computing*, 46–52. https://doi.org/10.1145/2480362.2480373

[20] Zelda Mariet and Vitaly Kuznetsov. 2019. Foundations of Sequence-to-Sequence Modeling for Time Series. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, 408–417.

[21] Ryan McClarren. 2021. *Recurrent Neural Networks for Time Series Data*. 175–193.

[22] Lucia Pancikova, Martina Hlinkova, and Lukas Falat. 2015. Prediction Model for High-Volatile Time Series Based on SVM Regression Approach. https://doi.org/10.1109/DT.2015.7222954

[23] Yao Qin, Dongjin Song, Haifeng Cheng, Wei Cheng, Guofei Jiang, and Garrison Cottrell. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. (04 2017).

[24] Adrian E. Raftery. 1985. Time series analysis. *European Journal of Operational Research* 20, 2 (1985), 127–137. https://doi.org/10.1016/0377-2217(85)90052-9

[25] Ajay Ramakrishnan and Michael Neff. 2013. Segmentation of hand gestures using motion capture data. 1249–1250.

[26] Kumar Ranjan, Debesh Tripathy, B Rajanarayan Prusty, and Debashisha Jena. 2021. An improved sliding window prediction-based outlier detection and correction for volatile time-series. *International Journal of Numerical Modelling Electronic Networks Devices and Fields* (01 2021). https://doi.org/10.1002/jnm.2816

[27] Ekaterina Spriggs, Fernando De la Torre, and Martial Hebert. 2009. Temporal segmentation and activity classification from first-person sensing. *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 0 (06 2009), 17–24. https://doi.org/10.1109/CVPRW.2009.5204354

[28] Yi-Fei Tan, Xiaoning Guo, and Soon-Chang Poh. 2021. Time series activity classification using gated recurrent units. *International Journal of Electrical and Computer Engineering (IJECE)* 11 (08 2021), 3551. https://doi.org/10.11591/ijece.v11i4.pp3551-3558

[29] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *31st Conference on Neural Information Processing Systems (NIPS 2017)* (06 2017).

[30] Phillip Wild, John Foster, and Melvin Hinich. 2010. Identifying Nonlinear Serial Dependence In Volatile, High-Frequency Time Series And Its Implications For Volatility Modeling. *Macroeconomic Dynamics* 14 (05 2010), 88–110. https://doi.org/10.1017/S1365100509090543

[31] Andrew Wilson, Aaron Bobick, and Justine Cassell. 1996. Recovering the Temporal Structure of Natural Gesture. (09 1996).

[32] Sibo Yan and Yuechun Gu. 2020. Price Forecast in High-Frequency Stock Market: An Autoregressive Recurrent Neural Network Model with Technical Indicators. https://doi.org/10.1145/3340531.3412738

[33] Chao Yang, Zhongwen Guo, and Lintao Xian. 2019. Time Series Data Prediction Based on Sequence to Sequence Model. *IOP Conference Series: Materials Science and Engineering* 692 (11 2019), 012047. https://doi.org/10.1088/1757-899X/692/1/012047

[34] Xuan Zhang, Xun Liang, Aakas Li, Shusen Zhang, Rui Xu, and Bo Wu. 2019. AT-LSTM: An Attention-based LSTM Model for Financial Time Series Prediction. *IOP Conference Series: Materials Science and Engineering* 569 (08 2019), 052037. https://doi.org/10.1088/1757-899X/569/5/052037

[35] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. 2020. Exploring Self-Attention for Image Recognition. 10073–10082.