Learning Deep Social Interactions to Identify Positive Classroom Climate

by

Przemek Gardias

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

May 2021

APPROVED:

Professor Jacob Whitehill, Major Thesis Advisor

Professor Neil Heffernan, Thesis Reader

Professor Craig E. Wills, Head of Department

Abstract

Recent work on automatically estimating the level of Positive Climate (PC) in school classrooms, as defined by the Classroom Assessment Scoring System, has demonstrated success in using deep neural networks to model the scene of a classroom as a social network graph. We theorize that by tracking participants within a social graph over time, we can attain higher CLASS prediction accuracy compared to previous work which ignored students' identities [1]. In this thesis, we (1) propose a process for constructing an ordered social network graph data structure over time. We then (2) conduct two experiments on simulated classroom observations to evaluate the effect of tracking people in order to utilize interactions between individuals when fitting a Graph Neural Network (GNN). Our findings suggest an improvement in classification accuracy when harnessing the feature interactions using the proposed tracking-based approach. Next, in an effort to improve the accuracy of tracking faces over time, we (3) analyze the latent embedding space of pre-trained face embedding networks and find suboptimal discriminability of faces in real-world classroom videos with highly non-frontal pose and very young children. Finally, with the aim of improving the discriminability of these embedding models, we (4) explore the viability of fine-tuning a pre-trained face embedding network on classroom videos, where the labels are extracted in a self-supervised manner. Experiments on classroom videos from YouTube and the UVA Toddler dataset suggest this can be effective: fine-tuning the pre-trained FaceNet, we adjust the embedding network to be better suited for a classroom setting, improving from a test ROC AUC (distinguishing same vs. different face) of 0.95 to 0.98 on unseen classroom observation videos [2] [3] [4].

Acknowledgements

My sincere thanks to Professor Jacob Whitehill for his invaluable insights and assistance guiding this work, and Professor Neil Heffernan for his patience and understanding. Special thanks to Anand Ramakrishnan for his valuable suggestions, and whose work inspired the direction of this thesis. Thanks to the rest of my colleagues at the Whitehill research group for their feedback and support along the way. A warm thanks for Professor Carolina Ruiz, for her wisdom, advice, and patience over many years of uncertainty.

The computational resources on which the fine-tuning component of this work was performed on are supported by the Academic & Research Computing group at Worcester Polytechnic Institute. Many thanks for their swift assistance on many occasions.

Finally, thanks to Petar Veličković and Jure Leskovec, who have provided so many freely available educational resources on the topic of Graph Neural Networks in the recent years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A widely-used classroom observation protocol used by educational researchers is the Classroom Assessment Scoring System, CLASS [5], which requires trained human annotators to examine the state of the classroom and students for qualities that exhibit social, organizational, and instructional support. CLASS is a valuable tool for teachers and educational researchers—but human annotation is slow, expensive, and requires weeks to months of training. Typical CLASS annotation sessions require annotators to examine specific characteristics of the states, actions, and interactions among the students and teachers during either live observation or video recordings.

A 2017 study from Chile found costs of individual classroom annotation to be in the order of $100 [6]. The magnitude of these costs makes it difficult to provide raw data for teacher feedback, a useful mechanisms for providing resources for teachers to adjust their teaching methods to effectively support all students. We seek to improve on initial work towards learning classroom climate classification by focusing on CLASS-defined social interactions in a weighted social graph representation of the classroom scene. By definition we focus on identifying positive interactions between student-teacher pairs to verify social support structures within the classroom

environment.

Chapter 2

Background

The first work towards automating aspects of CLASS annotation makes strides to estimate 3 minute clips of classroom observation videos which were most relevant to CLASS annotators to code manually [5] [7]. However, more recently, there have been a number of efforts to analyze the dynamics of a classroom, some focused on an aggregate measure, such as [1] and [8], while others focused on individual students [9]. Although there are many approaches for harnessing deep learning for measuring educational metrics such as CLASS, many prior efforts focus on analyzing student engagement and emotions [10]. While there are many different approaches to labeling classroom observations using CLASS metrics, we aim to distinguish short video clips with high or low positive climate (PC). We follow the intuition behind CLASS climate labeling methods by learning interactions between individuals encoded in a social network graph representing the social scene of a classroom [5].

2.1 Graph Neural Networks

Graph Neural Networks (GNN) are broadly applied to graph classification, node classification, and edge predictions tasks. We focus on these former of these tasks

where we are interested in using node and edge features which can assist with identifying positive climate within a social graph representation of the classroom. GNN layers can be categorized into three types of layers: convolutional, attentional, and message-passing [11].

2.1.1 Convolutional GNN

Convolutional GNNs are a category of recent deep learning architecture applicable for problems modeled as graphs. Although there are two notably different approaches to applying a transform on a graph data structure, spectral and spatial convolution layers, they are similar in their outcome: features are aggregated across neighbors with explicit graph-based regularization, as shown in Eq. (2.1) [11] [12]. Recent examples of convolutional GNNs are Graph Convolutional Networks (GCNs) [12], Chebyshev Networks (ChebNet) [13], and Simple Graph Convolution (SGC) [14].

$$h_i = \phi(v_i, \bigoplus_{j \in N_i} c_{ij}\psi(x_j)) \tag{2.1}$$

This transform is useful to transfer information from neighboring nodes, in a manner similar to a 2D convolution layer, while considering the topology of the graph. It is especially useful for homophilous graphs when edges encode label similarity [11].

2.1.2 Attention GNN

Attention GNNs use implicit weights between neighboring nodes via attention $\alpha_{ij} = a(v_i, v_j)$, as shown in Eq. (2.2) [11]. They differ from convolutional GNNs with these learnable, more complicated combinational attention weights between nodes.

$$h_i = \phi(v_i, \bigoplus_{j \in N_i} a(v_i, vj)\psi(v_j)) \tag{2.2}$$

Recent examples of attentional GNNs are Graph Attention Networks (GATs) [15], Gated Attention Networks (GaAN) [16], and Mixture Model Networks (MoNet) [17].

2.1.3 Message-passing GNN

Message passing GNNs use the sender and received nodes v_i and v_j to compute messages $m_{ij} = \psi(v_i, v_j)$ to be sent across edges, as shown in Eq. (2.3) [11]. This requires an entire weight vector for each edge in the graph.

$$h_i = \phi(v_i, \bigoplus_{j \in N_i} \psi(v_i, v_j)) \tag{2.3}$$

This variants of GNNs are often used for computational chemistry, reasoning, and simulations. Recent examples of message-passing GNNs are applications in quantum chemistry by Gilmer *et al.* [18], Interaction Networks [19], and GraphNet (GN) [20].

2.2 Social Network Graphs

Recent work in identifying classroom climate, ACORN [1], achieves significant inter-coder reliability results with respect to expert labels using a multi-modal deep learning ensemble. In an experiment using a temporal uniform normalized Laplacian matrix, where all other weights are $\frac{1}{d}$, such that the graph is a clique, ACORN establishes that graph topology, or *who is where, and interacting with whom, and when* is important for estimating classroom PC, achieving an average of $AUC = 0.70$ across 10-folds.

Traditional deep learning methods, particularly convolutional neural networks (CNN) have been shown to perform poorly on data with underlying graph structures, such as social network graphs. Some methods explore extending CNN components

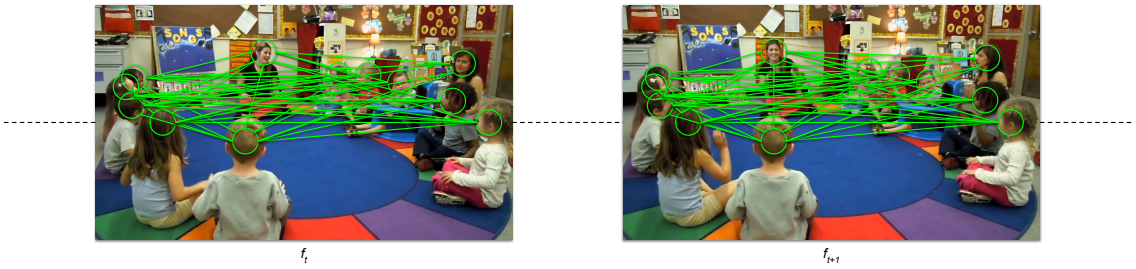


Figure 2.1: Example of social graph structure for encoding relations between entities on two frames of classroom observation video.

to graphs such as graph signal processing (GSP), which demonstrates graphs to be perfect for capturing node interactions, which is of interest to our application, especially on non-Euclidean data domains [21]. A recent application of GCNs on a problem modeled as a spatio-temporal social network graph achieves state of the art results and shows the model is able to capture behavior expected in humans [22]. The same work notably achieves these results with significantly less parameters are and a fraction of the training data used by previously comparable methods which did not model the scene as a graph.

2.2.1 Classroom Observations as Graphs

There are recent methods which successfully leverage computer vision for automating certain aspects of CLASS that identify relevant segments of classroom video important for coders rather than predicting a label [7]. Other methods use dedicated hardware to unify a multitude of contemporary machine learning techniques [9]. Tapaswi *et al.* use character face tracks for facial embedding clustering and achieve success resolving a large number of individuals through a combination of facial (hair) and non-facial (clothing, spatial locations) cues [23], building off of previous work which uses a Siamese network trained on tuple sample with a boolean indicator denoting if the samples are of the same or different faces [24].

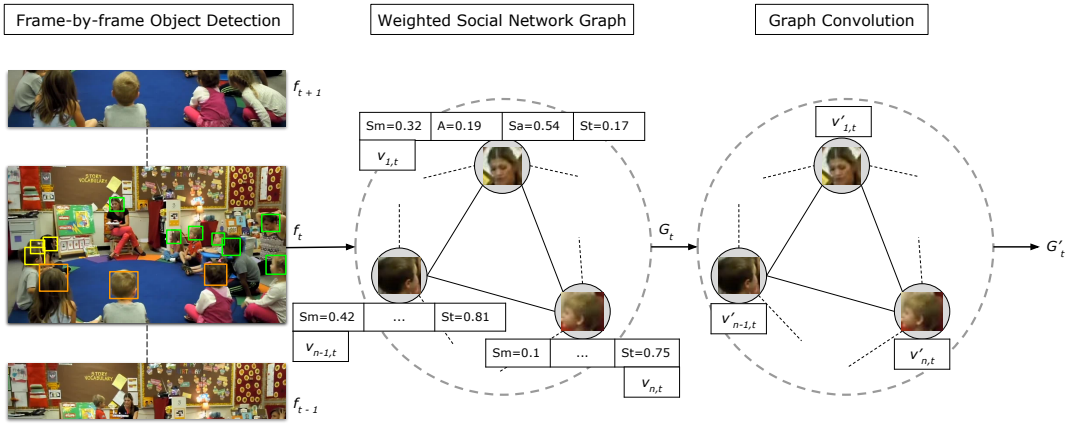


Figure 2.2: Process of converting a sequence of classroom observation video frames to social network graph.

Due to the requirements of our approach, we consider off-the-shelf object recognition and tracking tools for our processing pipeline. Visual perception frameworks such as OpenPose [25] and OpenFace [26] offer comparable utility as cloud services– and are successful applied as low-level feature engines, processed to estimate higher-level features. A concern when applying these tools is our niche classroom environment recognition task. Often, researchers build bespoke perception systems specific to student learning environments [27]. Similar toolkits are available for object tracking [28], but only some address multi-object tracking (MOT) [29] [30]. These techniques are applied as part of our scene to graph pipeline shown in Figure 2.2, where an example sequence of frames f_{t-1}, f_t, f_{t+1} is used to show the creation of a weighted social network graph $G_t = (V_t, A_t)$ from frame f_t, where each node $v_{n,t} \in V_t$ contains feature vector consists of emotion and age information as proposed in [1].

2.3 Embedding Networks

Suppose we want to determine whether two objects are of the same class, even in scenarios where the classes seen during testing are not the same during training. One approach which can provide us information about the object is an embedding, a latent interpretation of the object. Embedding networks used to create these object embeddings can be trained to place the embedding in a high-dimensional latent space with meaningful locations to other objects which we might be interested in comparing to, such that similarly labelled objects are clustered in close proximity. We then can use these embeddings to compare the objects they represent. The Euclidean distance between the two embeddings is a measure of difference, while the cosine similarity can be interpreted as a measure of similarity.

Embedding networks are commonly combined with object detection techniques as a method of identifying and comparing objects within the same or different scenes. Recent embedding network such as FaceNet apply this technique to facial detections, which are pre-processed by an alignment step performed by Multi-task Cascaded Convolutional Neural Network (MTCNN) [2] [31]. FaceNet embeds images into a 512 dimension latent space, and is trained on CASIA-WebFace [32] and VGGFace2 [3] separately, the network trained on the latter achieving a state of the art performance of $99.63\% \pm 0.09$ mean classification accuracy on Labeled Faces in the Wild (LFW). FaceNet minimizes triplet loss which, inspired by nearest-neighbor classification, aims to ensure that an image of the same person x_i^a (*anchor*) is closer to all other images of the same person x_i^p (*positive*) than it is to any image of any other person x_i^n (*negative*) as shown in Fig. 2.3, is defined as

$$\sum_i^N \Big[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \Big]_+, \tag{2.4}$$

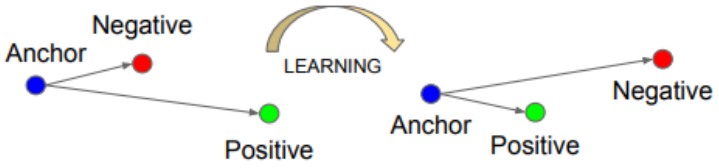


Figure 2.3: Triplet loss example before and after learning, where the distance between the *anchor* and a *positive* is minimized while maximizing the distance between the *anchor* and a *negative* [2].

where α is a hyperparameter defining margin that is enforced between positive and negative pairs [2]. Tapaswi *et al.* improves upon this strict definition of the hypersphere with a supervised approach for defining the layout of this latent space [23]. Notably, the training procedure of FaceNet emphasizes the necessity of selecting hard triplets, which are triplets that do not already easily fulfill the constraint with the current weights. Done naïvely, following this procedure requires forward propagating each of the training samples for an estimation of similarity to inform these pickings. Therefore, the training procedure avoids selecting the hardest negatives to mitigate entering a poor local minima during training and instead selects *semi-hard* x_i^n where the norm between x_i^a and x_i^p is less than that of x_i^a and x_i^n. An effective method of selection is crucial for fast convergence, especially combined with small mini-batch sizes used for the same reason during Stochastic Gradient Descent (SGD) [33].

The datasets which FaceNet is trained on consist of semi-automatically collected face images from the internet, often overwhelmingly consisting of celebrities and other widely photographed individuals. Therefore, the resulting networks are inherently biased to perform better on this population which they have been exposed to, and poorly on younger individuals' faces, which we are interested in embedding. For niche tasks such as our own, a common technique to increase performance is known as fine-tuning, where a pre-trained model is further trained on a dataset which is

better representative of the task in consideration, starting with a small learning rate and annealed as necessary.

Siamese Neural Networks consist of two networks with shared weights that are joined at their output with a feed-forward network. They were first introduced for signature verification by Bromley *et al.* which trained the parallel networks and used only one for a signature to embedding representation task which was then evaluated with a comparison function between a stored representation for the signer's original embedding [34]. These types of networks are similar to the embedding networks which are relevant to our task in their learning of a latent representation for similarity necessitating task. In effect, a Siamese Neural Network can be used with a similar binary classification task, where the network learns to differentiate positive and negative matching faces rather than including both a positive and negative example in each training sample. Taigman *et al.* experiment with Siamese networks which optimizes L_1-distance between face features to achieve 97.35% on LFW using an ensemble of three networks combined using a non-linear SVM [35]. The latent representation used at the output of each of the parallel networks are similar to the embeddings which FaceNet outputs, as the learned high-dimensional hypersphere representing the latent space are structured in similar manners, where clustered samples are aimed to be easily distinguishable from others.

2.4 UVA Toddler

The University of Virginia (UVA) Toddler dataset consists of 192 CLASS-coded videos, each approximately 45-60 minutes long. The videos are from 61 early childhood care centers, where the students are toddlers 2-3 years old. All videos were recorded from classrooms in a Mid-Atlantic state of the USA. For our purpose, each

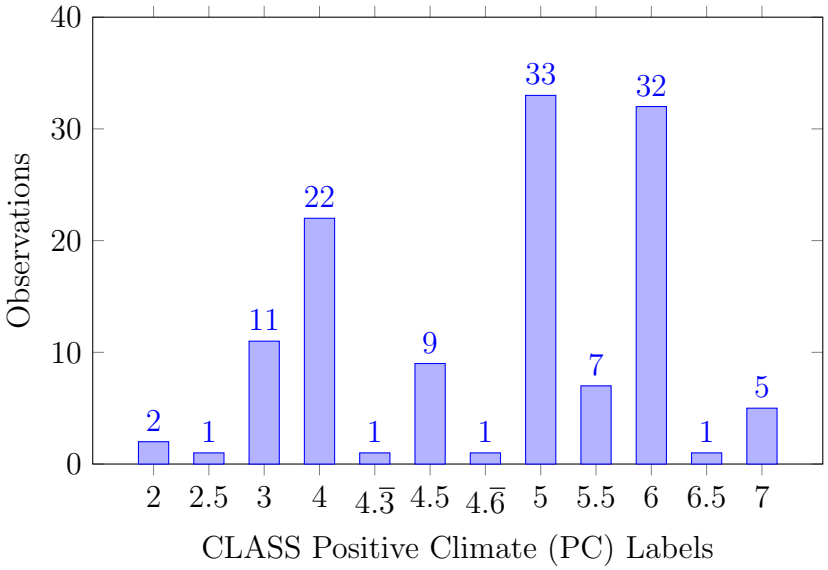


Table 2.1: Distribution of positive climate labels of 125 classroom observation videos in the UVA Toddler dataset.

of the videos is split into short clips with high or low PC. To lessen the effect of erroneous social network construction, and to simplify our tracking problem, we cap the number of face detections in each frame to 22, the maximum number of participants in any of the UVA Toddler classroom videos. Given the non-uniform distribution of PC labels shown in Table 2.1, it is clear why a regression approach is necessary, in combination with efforts to counter the effects of an unbalanced dataset on model fit.

In a similar fashion as ACORN [1], we can investigate our ability to generalize to elementary and middle school students, as well as compare climate classification accuracy using the Measures of Effective Training (MET) dataset, which contains thousands of videos and is similarly CLASS-coded.

2.5 Research Questions

We propose a number of questions to focus our experimentation towards the goal of verifying whether the intuition behind human annotation of CLASS climate is applicable deep learning classification techniques.

RQ1. Is the traditional graph data structure sufficient to encode temporal social interactions such that a deep network can exploit spatial information to inform climate prediction?

RQ2. What is the effect of erroneous tracking on our classification?

RQ3. Can we predict climate more accurately, even with sources of error in the construction of the graph data structure?

RQ4. Can we improve tracking accuracy by fine-tuning existing facial embedding techniques, given poor performance on children in our classroom setting?

Chapter 3

Simulated GNN

We theorize that if we apply a graph convolution network on social graphs representation of our classroom observations, we can capture key interactions between participants which are central to distinguishing between high and low PC. In this section, we introduce GCN models with node tracking capabilities which are designed to learn our simulated feature-based thresholds for classification. This network architecture combines graph convolution and shared-weight LSTM layers which processes the graph time series node-wise and then aggregate the hidden states for each node via a graph pooling mechanism to then classify classroom positive climate. Figure 3.1 compares this architecture to a similar network which ignores participant identities by pooling before the time series processing step instead. We conduct a series of experiments to compare the ability to predict CLASS climate by tracking participants in classroom videos with increasing complexity of node interactions.

3.1 Model Architecture

We define a social network graph as the graph representation of individuals in each frame of classroom observations, where the feature vectors of length v for each

person contain float elements indicating smiling, anger, sadness, and probability the individual is a student, $P(student)$ (equivalent to $1 - P(teacher)$), as established in previous related work [1]. Our social network graph follows a standard graph data structure for each time step: $G_t = (V_t, A_t)$ with n nodes, each with 4 features per node, such that $V_t \in \mathbb{R}^{n \times 4}$, where each row is the feature vector for a particular person, and $A_t \in \mathbb{R}^{n \times n}$. Notably, we do not define an adjacency as $a \in V_t \times V_t$. Instead our adjacency matrix is weighted by the inverse pixel distance between nodes such that it depicts a *full graph*, in which each adjacency a_{ij} is

$$a_{ij} = 1 - \frac{d(v_i, v_j)}{\sqrt{w^2 + h^2}} \tag{3.1}$$

for each $(v_i, v_j) \in V_t \times V_t$, where d a function of the Euclidean distance between v_i and v_j, and w, h are the width and height of the simulated video frame, respectively. This process results in an adjacency matrix between 0 and 1 which more heavily weights closer neighbor nodes. In our simulated environment, the number of individuals represented by nodes n is held constant over the course of the time series.

Given this full graph structure, *all* nodes are convolved in each graph convolution layer, therefore we must be careful to avoid over-convolving towards a fully entropic graph state. Inspired by a simple formulaic depiction of a feed-forward neural network, which omits the bias term, we can define our graph convolution layer as follows: for layer l, our spatial graph convolution layer activation output $H_t^{(l)}$ is

$$\begin{aligned} H_t^{(0)} &= V_t \\ H_t^{(l+1)} &= \sigma(L_t H_t^{(l)} W^{(l)}) \end{aligned} \tag{3.2}$$

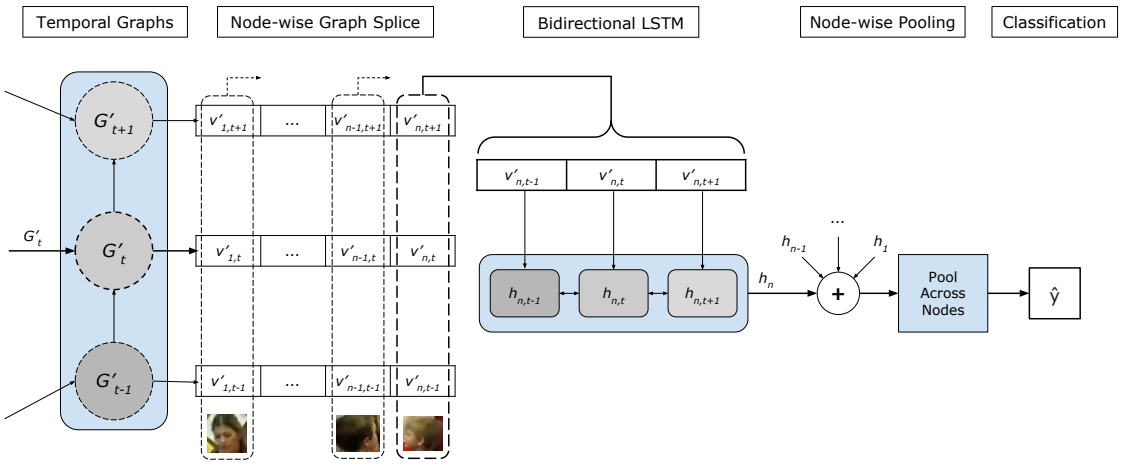
where the symmetric normalized Laplacian L_t is

$$L_t = I_t - D_t^{-\frac{1}{2}} A_t D_t^{-\frac{1}{2}}, \tag{3.3}$$

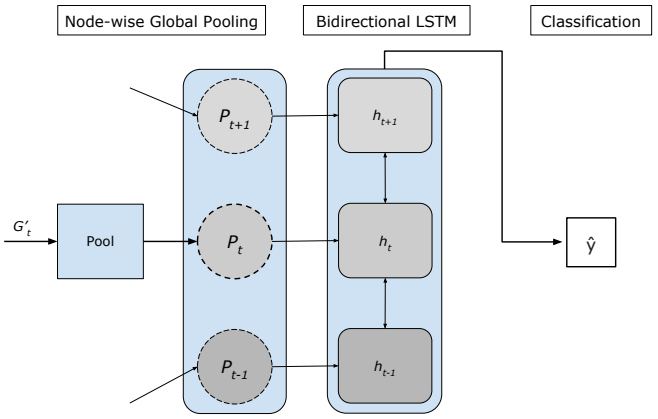
in which $I_t \in \mathbb{R}^{n \times n}$ is an identity matrix, $W^{(l)}$ is the convolution kernel weights of the given layer, D is the degree matrix with diagonal $D_{ii} = \sum_j A_{ij}$, and σ is the non-linear activation function ReLU. We follow each graph convolution layer with layer normalization to address an exploding gradient [36].

3.2 Tracking-based GNN

Using this structure for our graph convolution layer, we define a tracking-based and tracking-free pair of models for PC classification, shown in Figure 3.1. For our purposes in this section, the tracking-based GNN aims to process the timeseries belonging to each of the individuals within a classroom observation separately. The tracking-based model includes a graph convolution layer for each time step t, after which each of the the convolved feature vectors $v'_{n,t}$ are concatenated temporally to once again represent a spatio-temporal social network. We splice the vertex set node-wise such that each tracked node has an individual time series, and then aggregate using a bidirectional LSTM with 8 neurons, which is used for each node time series but shares weights across all nodes. We then pool across the output hidden states, such that we retain information from all nodes. On the other hand, the tracking-free model does not assume tracked features are retained across the time steps of the temporal social network graph, and includes a global pooling layer across nodes such that $pool(G_t) \in \mathbb{R}^{v \times 1}$, also followed with a bidirectional LSTM layer. Both models include a final fully-connected layer for binary classification, which is the PC classification task we focus on for these experiments.



a) Tracking network



b) Tracking-free network

Figure 3.1: Architectural differences between our two proposed graph convolution models.

To evaluate the design choices of our architecture model, we define a tracking problem: given a plausible classroom scenario, where *tracking the state and proximity of nodes to each other is causally related to the PC label*, we construct a randomly generated time series with length t seconds of features for n nodes. We seek to replicate the process of identifying key classroom moments within the larger duration of the video with a threshold decision function. We sample our input features $V \in \mathbb{R}^{n \times t}$ such that each feature $v_{n,t} \sim \mathcal{U}(0, 1)$, and evaluate for y as follows: a threshold function must be met for a percentage of features $v_{n,t}$ and again by a percentage of the nodes for a positive label. We aim to introduce a requirement which pairs of nodes must meet simultaneously that is unable to be captured with an architecture that does not perform node-wise processing. However, we expect a real world data set, with higher dimension features and a more complex ground truth label function, might perform differently and therefore comparisons of tracking methods would be difficult to evaluate consistently.

3.3 Feature-based Interactions

We continue our experimentation by expanding the method with which we determine our ground truth label: we consider proximity-based interactions with features which we sample from $\mathcal{U}(0, 1)$. The final identity features of teachers and students are sampled from uniform distributions $\mathcal{U}(0, 0.5)$ and $\mathcal{U}(0, 0.5)$, respectively. Additionally, we simulate participant movement across a frame by random initializing participants across our standard scene size with height h and width w and sampling

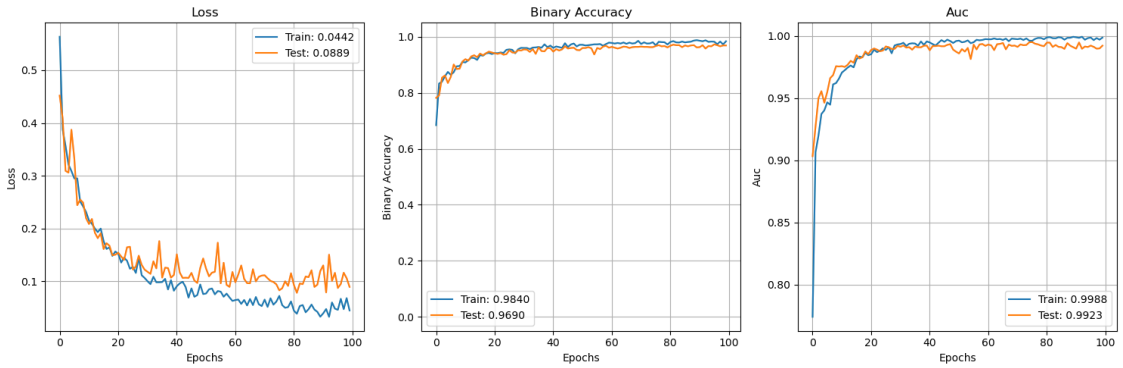
movement vectors of each participant:

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \gamma * \begin{bmatrix} \mathcal{U}(0, w) \\ \mathcal{U}(0, h) \end{bmatrix}, \tag{3.4}$$

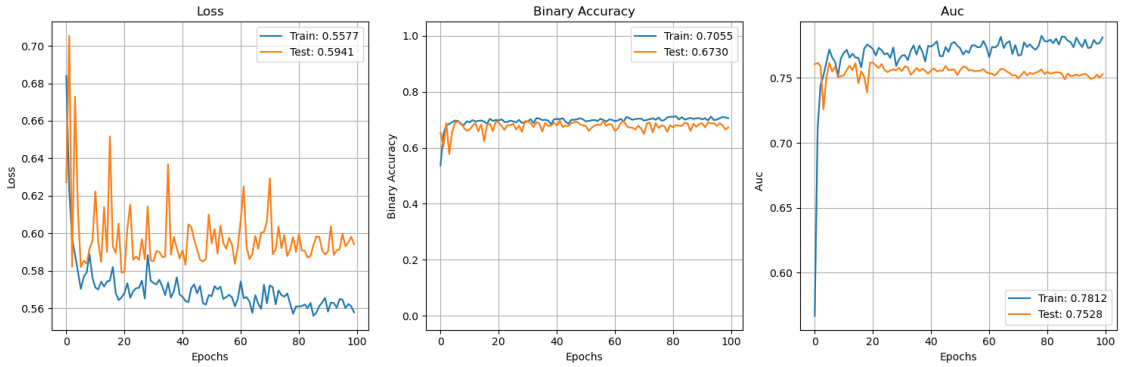
where γ is a small hyperparameter to mimic movement. We clip the final positions of participants such that it cannot exceed the bounds of our simulated frame, and prevent sparse graph representations due to occlusion or exit of scene events, which are common in real world data. In addition to these changes, we adjust our evaluation of y by defining a proximity threshold as a proportion of the diagonal of the frame to count a student-teacher interaction as positive in our simulated dataset construction.

3.4 Tracking Error

As mentioned in RQ2, we are interested in the effects of erroneous matches when tracking individuals over the course of a classroom observation. Therefore we additionally conduct an experiment in which we simulate datasets of increasing length in frames. For each frame simulated, we introduce a probability p of an erroneous swap occurring, during which the features $v_{n,p}$ of two random individuals are switched and therefore are matched to their new identifies in the next processed frames. We investigate both how a given probability p of a swap occurring and the corresponding length of the simulated timeseries correspond to the AUC ROC.



a) Tracking model



b) Tracking-free model

Figure 3.2: Training results of a simple single feature simulation for the tracking-based network compared to the tracking-adverse model. Both models were trained for 100 epochs, using $lr = 1e-2$ and a dataset simulated with $n = 4$ nodes and $t = 10$ timesteps.

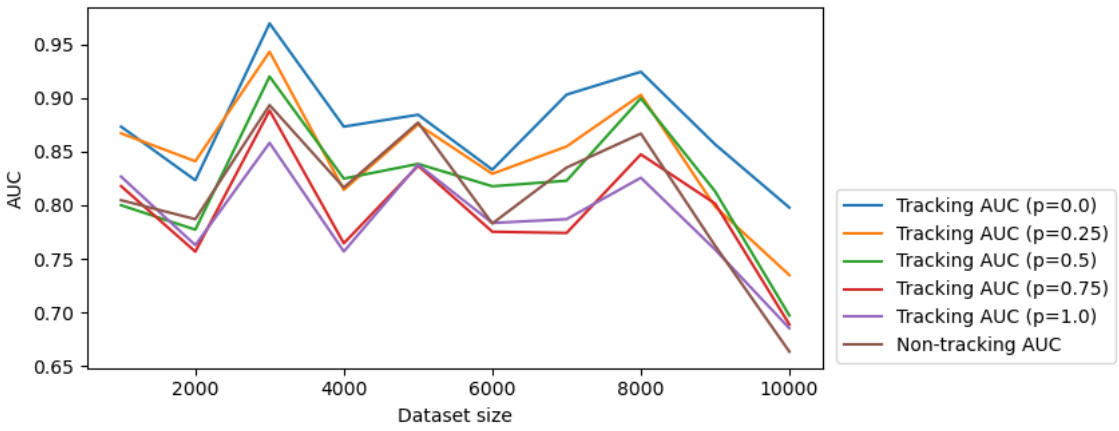


Figure 3.3: ROC AUC of simulated models given increasing probabilities of a swap error occurring during the construction of the dataset.

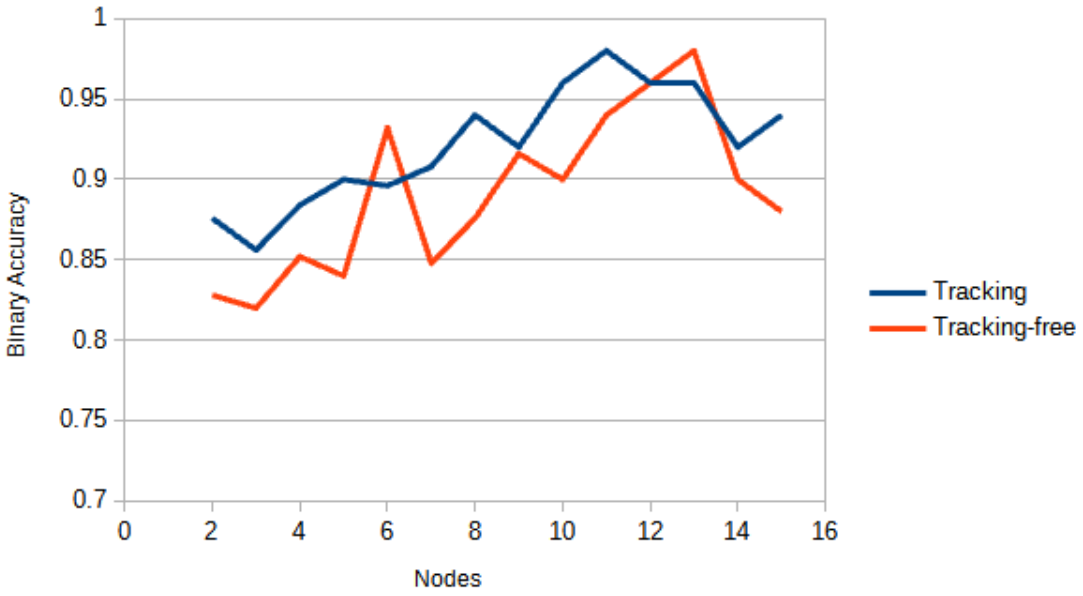


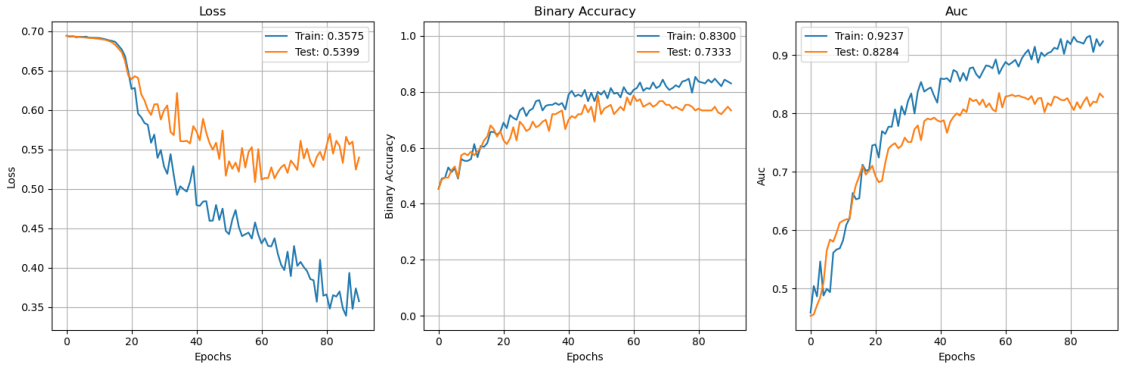
Figure 3.4: Tracking vs. tracking-free test binary accuracy for n nodes. Models were trained for 100 epochs, with early stopping and optimal weight restoration to maximize validation set metrics for each simulation configuration.

3.5 Results

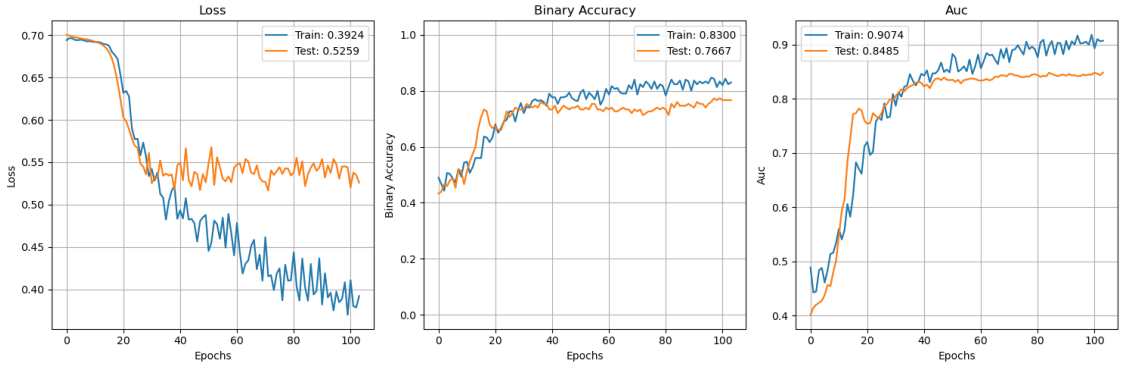
As described in Section 3.1, we compare two architectures for classification, where a bidirectional LSTM layer is either followed by a pooling layer, as in the tracking-based network, or preceded by a pooling layer, as in the tracking-adverse network. This distinction is visualized in Figure 3.1. We examine the effect which node-level data loss contributes to the network's ability to fit to a correlation defined for each individual time series by comparing test classification accuracy. Furthermore, as shown in Figure 3.3, training on longer datasets with a higher swap probability p resulted in lower ROC AUC, although the results on shorter datasets are less conclusive and fluctuate significantly. In order for a tracking-based graph convolution network to work better than a tracking-free variant, the discriminability of face embeddings must be good to minimize the swapping of individuals.

As shown in Figure 3.2, we identify an accuracy improvement when tracking participants, most likely due to loss of meaningful features required for the interaction learning we seek to accomplish. These results affirm two key points: our tracking-based architecture seem to be able to capture the node interactions, and the tracking-adverse model is able to perform seemingly well even in the absence of learning interactions. Identifying the performance difference provides us a baseline comparison differential of node-level learning, which is especially useful in the context of real-world data. This supports further investigation of expanding our simulation to include more participants additionally validated by Figure 3.4, which presents a performance difference in a trend of increasing accuracy given increasingly larger graphs. We hypothesize, following the assumption that CLASS climate labels are based upon node interactions, that given a sufficiently deep node feature capture method, we can identify a method which employs graph convolution for social network feature propagation and tracking-based time series processing, that captures this interaction sufficiently enough to perform better than a tracking-adverse network. In this way, we seek to identify a method which is able to understand the causal mechanism of how the node-wise interactions result in CLASS climate.

Results of further experimentation on the efficacy of the proposed architecture on our problem, described in Section 3.3, are shown in Figure 3.5. By increasing the simulated participant features available, and expanding the logic of our ground truth labels to account for complex interactions which are reliant on more of the available features, we create a significantly more difficult to capture differentiation between labels. Although most notably the differences between model accuracy are not clearly identifiable, this lack of an accuracy difference may be indicative of our models ability to capture very nuanced interactions which are reliant on several feature requirements in tandem. This introduces an opportunity for node



a) Tracking model



b) Tracking-free model

Figure 3.5: Training results of a simulation for the tracking-based network compared to the tracking-adverse model. Both models were trained for 100 epochs, using $lr = 1e - 3$, with our dataset simulation configured to $n = 22$ nodes and $t = 10$ seconds.

attention mechanisms to identify key neighboring nodes for our graph convolution steps, which we expect to be necessary for success considering the initial findings of [1]. Furthermore, we do not concede the ability of a tracking-based network to achieve success on real world data, given the difficulty to represent the true relationship between social interactions and PC labels as exists in the UVA Toddler dataset.

Chapter 4

Tracking

To create the necessary data to train and evaluate our GNN with, we convert the videos available to us to a graph data structure. We aim to improve on techniques demonstrated in ACORN by evaluating an application of our proposed tracking-based graph classification method [1]. To achieve this, we consider methods with which we can track individuals, representing nodes, between sequential temporal frames of video using traditional object tracking methods such as GOTURN [28], or trying to exploit our knowledge of limited nodes with unsupervised node embedding network, such as node2vec [37]. In this section, we aim to establish a node-wise sorted vertex set V for all frames to fulfill the data structure assumptions held by our tracking-based network.

4.1 Experiments

As discussing in Section 2.3, embeddings can be used to evaluate and then maximize facial similarity between detected individual's faces in sequential frames. After evaluating all combinations of faces, we pick those with the highest similarity until there are no possible remaining matches. Although this greedy method does not

mean that we guarantee a globally optimal solution, it may yield a locally optimal solution that approximates a globally optimal one in a reasonable computational time. Since we are interested in evaluating the classification accuracy between the tracking-based and tracking-free architectures described in Figure 3.1, we want to be able to construct our social network graphs from classroom video that have minimal erroneous swaps between detected individuals. If there are too many erroneous swaps due to incorrect matches, we expect to observe a corresponding decrease in classification accuracy of the tracking-based approach.

We manually annotate faces in matching first and last frames from six clips, each of length 90 second and extracted from publicly available classroom videos, using the VGG Image Annotator (VIA) [38]. The resulting dataset consists of ordered listing of individuals in each of the frames, such that the identifiers matched across the first and last frame tuples.

4.1.1 Matching Individuals

We first compare a few accessible off-the-shelf facial detection tools: FaceNet [2], OpenCV [39], MTCNN [31], YOLOv3 [40], and implementations based off the aforementioned frameworks, controlling for the recognition threshold. In a comparison of the number of faces consistently detected in a clip of video, YOLOv3 resulted in the highest number of detected faces with the lowest amount of observed variance [40].

Using our manually labeled faces, we want to understand how accurately we are able to match detected faces across frame pairs. Our evaluation of similarity between two detected faces as a combination of Euclidean distance and embedding

cosine similarity, where parameter δ determines the weight of each:

$$\delta * \frac{E(A_p) \cdot E(B_p)}{\|E(A_p)\|\|E(B_p)\|} + (1 - \delta) * \|A_{xy} - B_{xy}\|, \tag{4.1}$$

with embedding network $E(x)$, where each individual, here denoted as A and B, have 2-dimensional coordinate vector (A_{xy} for person A and B_{xy} for person B) and facial pixels (A_p for person A and B_p for person B). We could alternatively use the Euclidean difference between embeddings for similar results, as the two methods of similarity are comparable and proportional. Using YOLOv3 as our facial detection component and FaceNet as the embedding network, we evaluate matching faces across 90 second gaps, corresponding to the clips we manually annotated [40] [2].

Results

When using only the embeddings as our evaluation of who is who ($\delta = 1$), the resulting accuracy was inconsistent and far from ideal, ranging from 25% to 62.5%. On the other hand, experimentation with a low δ demonstrated success of videos which previously had low accuracy but did not improve our tracking accuracy across all video clips. However, our task is dependant on success tracking individuals within video frames consistently across sequential frames. Therefore the results found when tracking across first and last frames of video clips are not indicative of a lack of accuracy on such a problem. So we evaluate our pipeline in a different manner, this time establishing an assumption which allows us to evaluate over a large number of comparisons: we can match detected faces across near-sequential frames by greedily picking the matching face based on the Intersection over Union (IoU) of the two bounding boxes. Sampling every 9 frames of video, with the ground truth labels found with this assumption, we find that using only similarity of embeddings ($\delta = 1$)

we are able to match with 89% accuracy, where the average IoU of all the matching frames is 34%. We repeat this experiment, instead sampling every sequential frame and find similar results.

4.1.2 Average Point Drift

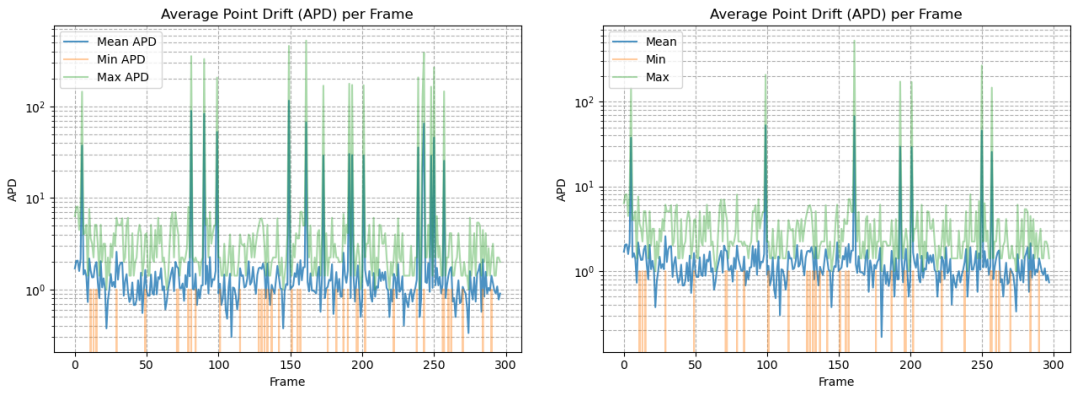
Considering RQ3, we acknowledge a few sources contributing to inaccuracies between the true distribution of individuals within a frame of video and a possible graph representation which we may learn on. We can consider discrepancies between these two as a result of the errors at each of the following steps:

- detection of faces within each frame,
- matching of individuals between frames, and
- interim feature construction.

We can estimate the results of the first two points given the assumption that on average, individuals move fairly consistently over the course of the video. This enables us to evaluate our success in this endeavour over segments of video by analyzing the proximity of our matched faces between frames. We expect that over time, given a small enough error between the two aforementioned components in our video to social graph pipeline, we will observe consistent movement. Errors in this process can be observed by the associated high variability of this metric. We refer to this measure as Average Point Drift (APD).

Results

A comparison of APD over a video clip with different values of the hyperparameter δ, shown in Figure 4.1, indicates lower tracking accuracy when the method of tracking



$\delta = 1$ $\delta = 0.5$

Figure 4.1: Plots of APD across 300 frames of classroom video. Spikes indicate likely erroneous swap occurrences between individuals, moments where the method of greedily matching faces based on the similarity heuristics made a decision which resulted in a large spatial difference between the matches.

solely uses embeddings ($\delta = 1$), instead of when a mix of embeddings and Euclidean distance is used ($\delta = 0.5$). Even with an optimal configuration for each segment of video, this approach would only succeed in tracking all individuals a few seconds at a time. Every few seconds, the tracking pipeline would incorrectly swap the nodes which the features are associated with, and assuming we continue to correctly match individuals, this would continue until a another erroneous swap occurred, likely swapping the feature vectors of two different nodes than before. This means that a temporal graph data structure which we derive from a 90 second video, which we are primarily interested in, would contain anywhere from 100 erroneous swaps of the temporal feature representations between all individuals. Although we believe that at a smaller scale this may be tolerable in a robust network, the frequency of these errors justifies experimentation towards improving our matching accuracy. Since the primary heuristic for the aforementioned greedy matching process is facial embeddings, we focus on fine tuning FaceNet for our embedding task such that we are able to identify an improvement in embeddings on a dataset consisting of

children in classroom observation videos [1].

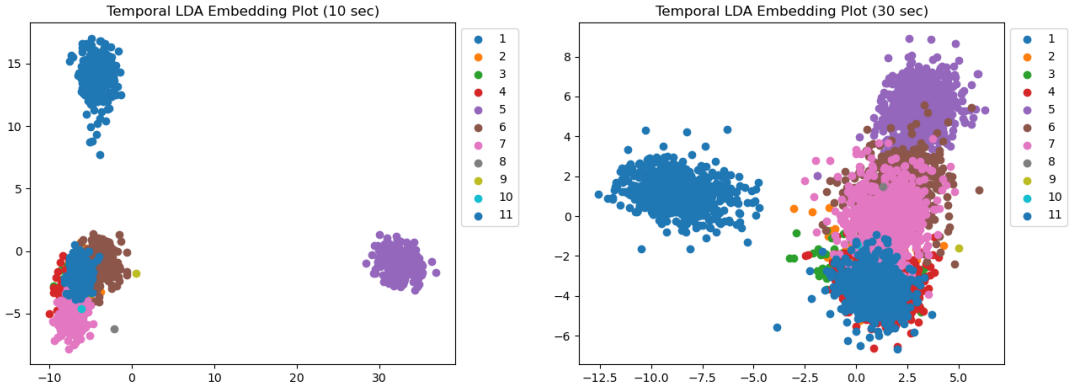
4.1.3 Latent Space Analysis

Linear Discriminant Analaysis (LDA) is a method used to find a linear combination of features that best separates the classes which the features describe. For our task, we are interested in investigating the discriminability of the embeddings of our detected faces. Since this method requires more than one sample for each of the classes, we aggregate the temporal embeddings representing each of our tracked individuals' faces. We apply LDA on the collection of embeddings, with each individual assigned a sequential integer as their class, and compare the progression of the primary feature across a 90 second classroom observation video clip, shown in Figure 4.2. Notably, the collection of data of a tracked individual in this video likely includes erroneous swaps between the features of individuals, contributing to the increasing visual indiscriminability of the plotted embeddings. However, since each plot contains progressively more data points, aggregated since the beginning of the video clip, this observation is not applicable to the individual per-frame pair task of matching but rather indicative of generally poor consistency of embeddings across the course of a video clip.

4.2 Classroom Observation Faces

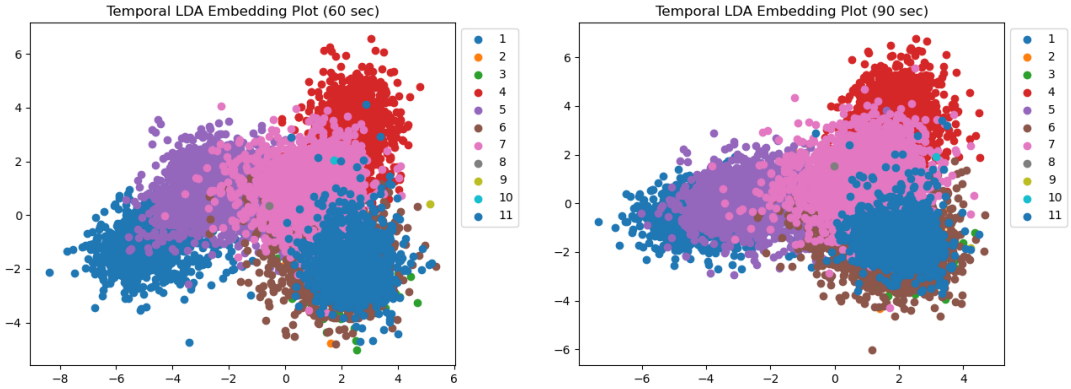
The process of fine tuning FaceNet necessitates a unique approach due to the difficulty of sourcing data suited to our task. Since we aim to improve our embedding accuracy on children in temporal proximity, the best source of data is videos similar to the ones which we aim to generate social network graph structures from. Initially,

[1]We use the open source implementation and training procedures of FaceNet developed and trained by David Sandberg, source at `https://github.com/davidsandberg/facenet` [2].



LDA of embeddings at 10 seconds.

LDA of embeddings at 30 seconds.



LDA of embeddings at 60 seconds.

LDA of embeddings at 90 seconds.

Figure 4.2: Linear Discriminant Analysis (LDA) of tracked individual's face embeddings over the course of a 90 second classroom observation video clip.

we use publicly listed classroom observation videos found on YouTube to find face tuples which we can use to construct the data necessary for training on. However, since our tracking problem is self-supervised, we are free to fine tune on the same data that we apply our network to. Given this additional source of faces for training data, we still partition unseen videos to evaluate our fine tuned network weights with.

As described in detail in Section 2.3, FaceNet optimizes triplet loss, which requires the training data to consist of an anchor, a positive match to the anchor, and a negative which is different from the anchor [2]. Furthermore, for fast convergence, the training process necessitates the selection of hard triplets which do not already fulfill the triplet loss constraint. Since the process of sourcing of our data does not permit the same class-based structure that the datasets which were originally used to train the network follow, we instead we apply the IoU method of identifying matching faces 3 frames apart, and repeat this across the length of the videos available to us. This allows us to maintain a high consistency in correctly matching individuals in separate frames while also often finding faces with sufficient differences in the data contained within their cropped image tuples. This process constructs a triply-nested dataset that does not permit hard triplet mining, and necessitates us to source the negative sample from the same frame pair from which the anchor and matching positive are derived from.

The results of our dataset construction are as follows: from the 20 publicly listed YouTube videos we used as our initial source, we process 33,891 frames of video that can be used to construct triplet (with at least two matching faces) found across the frame pairs. Repeating this process on the UVA Toddler videos, we further process an additional 168,254 frames of video that are usable in training.

4.2.1 Cropping Considerations

The datasets which FaceNet are originally trained on are centered and aligned using MTCNN [2] [31]. This means the original image of an individual, often framed as a wide portrait, has a face detection process applied to it, the result of which is then used to adjust and crop the original image, as defined in [31]. This process results in uniquely adjusted face images, and is noted as necessary to fine tune FaceNet properly [2]. Since we use YOLOv3 as our method of facial detection in our process of dataset construction, we investigate the differences between using the two methods [40].

We compare a small number of faces for which we generate different bounding boxes for using both YOLOv3 and MTCNN [40] [31]. We then find the Manhattan and Zero norm—the sum of the absolute values and the number of elements not equal to zero, respectively—between the two images. We repeat this process for a baseline example image and compare it to itself with a Gaussian blur applied. For context, the Zero norm per pixel of the example image with a Gaussian blur applied is simply 1. With this process, the norms of the Gaussian blur difference are about five and three orders of magnitude greater than than the maximum values observed in our differently cropped faces. Visual, qualitative comparisons of the different face detection methods support these findings. We further explore the differences between these methods by comparing an example detected face to itself cropped by 80%. In this case, the Manhattan norm increases by five orders of magnitude, indicating a similar conclusion that the differences between the original FaceNet alignment method and ours are not sufficiently large enough to warrant re-constructing our dataset with MTCNN as our method of face detection [2] [31].

4.3 Fine-tuning FaceNet

We use the Classroom Observation Faces dataset to fine-tune the FaceNet weights originally trained on VGGFace2 [2] [3]. Since the format of our data does not match the requirements necessary to mine hard triplets, we instead create a procedure to construct our triplets without consideration for the affect of the fitting the network weights to the given sample. From the experiments conducted in Section 4.1.1, we know that the network is able to successfully embed many of our observed faces with sufficient similarity for our matching task, however we aim to fine-tune the network to extend this success, and specifically improve the performance of the network on the faces of children, which are not included in VGGFace2 or CASIA-WebFace [3] [32]. Additionally, since we use triplets which include temporally similar faces, we further fine-tune the network to embed these faces, which already have a low Zero norm, closer in the embedding space. Although we may already expect these to be embedded similarly if we were training an embedding network from scratch, with newly initialized weights, because of the aforementioned pixel similarity, the reality of our embedding matches experiment in Section 4.1.1 shows otherwise.

While fine-tuning FaceNet, we are able to evaluate the tuned network weights with accuracy on Labeled Faces in the Wild (LFW), the dataset and corresponding metric which the network was originally trained to excel on [2] [4]. However, more meaningful is an evaluation of the tuned network weights using a metric that is applicable to our matching task, even if it corresponds with worse LFW accuracy. Therefore, we primarily consider the area under the receiver operating characteristic curve (ROC AUC) *after* evaluating which of the triplet faces match to the anchor using cosine similarity.

Figure 4.3: Training loss curve while fine-tuning FaceNet on the Classroom Observation Faces dataset for 40 epochs with an initial $lr = 1e-5$ that is annealed further by a schedule [2].

4.3.1 Results

We fine-tune the network at the following learning rate schedule:

- Epoch 0: $1e-5$
- Epoch 15: $1e-6$
- Epoch 25: $1e-7$

for 40 epochs total. The training loss curve is shown in Figure 4.3. During training, we observed many triplets that resolved to effectively 0 loss, with sparse cases resulting with significantly higher loss many orders of magnitude larger. This is

likely due to that we do not mine hard triplets when generating them. As a result, the training process is less efficient than if only hard triplets were selected, and we continue iterating over already robustly represented samples for all epochs. The initial face classification (same vs. different face) test ROC AUC was 0.95, while after training this increased to 0.98. After the fine-tuning process, the resulting network weights evaluated to $91.4\% \pm 0.015$ accuracy on LFW with a validation rate (true accepts over number of same pairs) of $46.93\% \pm 0.031$ at $FAR = 0.001$.

Chapter 5

Conclusion

We demonstrated that inter-node requirements can be approximated with tracking-based graph neural networks by processing the timeseries of nodes with a sharing bidirectional LSTM, and identified a necessity for high accuracy tracking to succeed with this requirements. Since the video to ordered social network graph pipeline of state-of-the-art object detection and embeddings networks did not perform as well as we would like, we constructed a novel dataset of classroom faces consisting primarily of children's faces, using the IoU of detected faces in temporally close frame pairs as an indication of ground truth identities. We then fine-tuned FaceNet weights originally trained on VGGFace2 and evaluated on LFW, to improve the discriminability of children's faces in classroom observation videos from the original 0.95 test ROC AUC to 0.98 [2] [3] [4]. Although we do not manage to apply this procedure to convert the UVA Toddler video clips to social network graphs, we are able to improve the applicability of the embedding network on matching faces in classroom observation video.

The pipeline which we use required the loading of two fairly large neural networks in tandem. We designed software to synchronously load networks and batch jobs

through each network as necessary. This may not be the most efficient method of using many CUDA enabled networks in parallel, however performance was not a primary consideration. Rather, we required stability to ensure efficient usage of Turing cluster resources. Fine-tuning FaceNet required a recreation of the triplet loss training procedure, including a new method of loading the dataset. This training procedure on the entire Classroom Observation Faces dataset took just over an hour per epoch on an NVIDIA Tesla P100, and managing GPU memory usage during this process was difficult, often resulting in allocation issues.

5.1 Future Work

The most interesting next component of work is using the system we have developed in the original goal of this thesis, to evaluate a tracking neural network on a binary PC label, or even evaluating on a regression task of predicting the exact granular label. This would additionally be complemented by a comparison to the PC classification AUC of Ramakrishnan *et al.* [1]. This comparison would be best conducted given the same short clip configuration from videos in the UVA Toddler dataset.

Additional promising work includes expanding the capability of using FaceNet as an embedding method of children's faces. This would require sourcing more publicly available classroom observations, which could include 70 videos used as a dataset by Ramakrishnan *et al.* as well as including the MET dataset, and generating an expanded dataset of matching faces to sample triplets from [1]. Furthermore, as discussed in Section 2.3, alternative networks which include similar latent representations of facial pixels can be evaluated for a similar face matching task. Another consideration which we intended to explore when fine-tuning was optimizing both LFW accuracy and test ROC AUC. This would ensure that we do not sacrifice

success of general embeddings for success on our niche pre-K classroom task. Furthermore, although we are unable to mine hard triplets such that we achieve a similar training convergence speed as described by Schroff *et al.*, we can still select the more difficult triplets which are available to us, discarding those which are unnecessary, and as a result saving resources and compute time, allowing a larger dataset containing faces from additional classroom observation videos to be exposed to the network during fine-tuning [2].

Bibliography

[1] A. Ramakrishnan, B. Zylich, E. Ottmar, J. LoCasale-Crouch, and J. Whitehill, "Toward automated classroom observation: Multimodal machine learning to estimate class positive climate and negative climate," *arXiv preprint arXiv:2005.09525*, 2020.

[2] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[3] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, *Vggface2: A dataset for recognising faces across pose and age*, 2018. arXiv: `1710.08092 [cs.CV]`.

[4] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, Oct. 2007.

[5] R. C. Pianta, K. M. La Paro, and B. K. Hamre, *Classroom Assessment Scoring System™: Manual K-3.* Paul H Brookes Publishing, 2008.

[6] W. Bank. (2017). "The classroom assessment scoring system (class)," [Online]. Available: `www.worldbank.org/en/programs/sief-trust-fund/brief/the-classroom-assessment-scoring-system-class`.

[7] Q. Qiao and P. A. Beling, "Classroom video assessment and retrieval via multiple instance learning," in *International Conference on Artificial Intelligence in Education*, Springer, 2011, pp. 272–279.

[8] T.-Y. Yang, R. S. Baker, C. Studer, N. Heffernan, and A. S. Lan, "Active learning for student affect detection.," *International Educational Data Mining Society*, 2019.

[9] K. Ahuja, D. Kim, F. Xhakaj, V. Varga, A. Xie, S. Zhang, J. E. Townsend, C. Harrison, A. Ogan, and Y. Agarwal, "Edusense: Practical classroom sensing at scale," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 3, pp. 1–26, 2019.

[10] K. Holstein, B. M. McLaren, and V. Aleven, "Student learning benefits of a mixed-reality teacher awareness tool in ai-enhanced classrooms," in *International conference on artificial intelligence in education*, Springer, 2018, pp. 154–168.

[11] P. Veličković. (Feb. 17, 2021). "Theoretical foundations of graph neural networks," [Online]. Available: `https://petar-v.com/talks/GNN-Wednesday.pdf` (visited on 02/22/2021).

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[13] M. Defferrard, X. Bresson, and P. Vandergheynst, *Convolutional neural networks on graphs with fast localized spectral filtering*, 2017. arXiv: `1606.09375 [cs.LG]`.

[14] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International conference on machine learning*, PMLR, 2019, pp. 6861–6871.

[15] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, *Graph attention networks*, 2018. arXiv: 1710.10903 [stat.ML].

[16] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "Gaan: Gated attention networks for learning on large and spatiotemporal graphs," *arXiv preprint arXiv:1803.07294*, 2018.

[17] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model cnns," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5115–5124.

[18] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, *Neural message passing for quantum chemistry*, 2017. arXiv: 1704.01212 [cs.LG].

[19] P. W. Battaglia, R. Pascanu, M. Lai, D. Rezende, and K. Kavukcuoglu, *Interaction networks for learning about objects, relations and physics*, 2016. arXiv: 1612.00222 [cs.AI].

[20] P. W. Battaglia, J. B. Hamrick, V. Bapst, A. Sanchez-Gonzalez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. Dahl, A. Vaswani, K. Allen, C. Nash, V. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, *Relational inductive biases, deep learning, and graph networks*, 2018. arXiv: 1806.01261 [cs.LG].

[21] M. Cheung, J. Shi, O. Wright, L. Y. Jiang, X. Liu, and J. M. Moura, "Graph signal processing and deep learning: Convolution, pooling, and topology," *arXiv preprint arXiv:2008.01247*, 2020.

[22] A. Mohamed, K. Qian, M. Elhoseiny, and C. Claudel, "Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 424–14 432.

[23] M. Tapaswi, M. T. Law, and S. Fidler, "Video face clustering with unknown number of clusters," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5027–5036.

[24] V. Sharma, M. Tapaswi, M. S. Sarfraz, and R. Stiefelhagen, "Self-supervised learning of face representations for video face clustering," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, IEEE, 2019, pp. 1–8.

[25] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[26] T. Baltrušaitis, P. Robinson, and L.-P. Morency, "Openface: An open source facial behavior analysis toolkit," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, 2016, pp. 1–10.

[27] J. Whitehill, Z. Serpell, Y.-C. Lin, A. Foster, and J. R. Movellan, "The faces of engagement: Automatic recognition of student engagementfrom facial expressions," *IEEE Transactions on Affective Computing*, vol. 5, no. 1, pp. 86–98, 2014.

[28] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision*, Springer, 2016, pp. 749–765.

[29] G. Ciaparrone, F. L. Sánchez, S. Tabik, L. Troiano, R. Tagliaferri, and F. Herrera, "Deep learning in video multi-object tracking: A survey," *Neurocomputing*, vol. 381, pp. 61–88, 2020.

[30] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," *arXiv preprint arXiv:1909.12605*, 2019.

[31] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, Oct. 2016, ISSN: 1558-2361. DOI: 10.1109/lsp.2016.2603342. [Online]. Available: http://dx.doi.org/10.1109/LSP.2016.2603342.

[32] D. Yi, Z. Lei, S. Liao, and S. Z. Li, *Learning face representation from scratch*, 2014. arXiv: 1411.7923 [cs.CV].

[33] D. R. Wilson and T. R. Martinez, "The general inefficiency of batch training for gradient descent learning," *Neural networks*, vol. 16, no. 10, pp. 1429–1451, 2003.

[34] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 04, pp. 669–688, 1993.

[35] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[36] J. L. Ba, J. R. Kiros, and G. E. Hinton, *Layer normalization*, 2016. arXiv: 1607.06450 [stat.ML].

[37] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 2016, pp. 855–864.

[38] A. Dutta and A. Zisserman, "The VIA annotation software for images, audio and video," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19, Nice, France: ACM, 2019. DOI: 10.1145/3343031.3350535. [Online]. Available: https://doi.org/10.1145/3343031.3350535.

[39] Itseez, *Open source computer vision library*, https://github.com/itseez/opencv, 2015.

[40] J. Redmon and A. Farhadi, *Yolov3: An incremental improvement*, 2018. arXiv: 1804.02767 [cs.CV].