# Pose-Based Pedestrian Crossing Intention Prediction with Recurrent Graph Convolutions and Explainability-Driven Data Augmentation

Abel García Romera

**Abstract**

Accurately predicting pedestrian intentions to cross the street is crucial for the safety of both pedestrians and passengers. This paper presents three main contributions to the field of autonomous driving. Firstly, we propose a novel Graph Neural Network architecture for pedestrian crossing intention prediction, defined as a binary classification between the crossing and not-crossing labels. The input of the proposed classifier consists of the skeletons or pose of the pedestrians appearing in the JAAD dataset videos. We estimated pose using OpenPose and AlphaPose, and using AlphaPose skeletons results in better validation metrics due to improved joint detection. The proposed architecture for crossing intention utilizes Recurrent Graph Convolutional Layers (RGCL) to capture both spatial and temporal dependencies in the estimated pedestrian skeletons. We evaluated several RGCLs, including GConvGRU, GConvLSTM, TGCN, and GCLSTM, with different amount of input frames for the network. The optimal architecture utilizes the TGCN layer with 87 input skeletons for each pedestrian, representing the preceding 87 frames. Our proposed architecture outperforms state-of-the-art methods in terms of f1-score, precision, and recall. Additionally, we performed an explainability analysis implementing Grad-CAM on the RGCL, which identified the Neck, Right Knee, and Right Elbow as the most critical joints for the classification. Finally, based on these findings, we propose an explainability-driven data augmentation method that generates new skeletons by modifying the critical joints. Specifically, applying random horizontal displacements to both the Right Knee and its counterpart, the Left Knee, slightly enhances the validation and test metrics, compared to not using data augmentation.

**Index Terms**

Advanced Driver Assistance Systems, Data augmentation, Explainability, Graph Neural Networks, Pedestrian Crossing Intention

## I. Introduction

Accurately predicting pedestrian intentions to cross the street is a fundamental task that autonomous vehicles must accomplish, as it is critical for ensuring the safety of both pedestrians and passengers, and for improving the efficiency of traffic. This task consists of a binary classification between the crossing and not-crossing labels, for each pedestrian that appears in a video or image. Recent advancements in deep learning have shown promising results in predicting pedestrian crossing intentions with several different approaches.

An interesting approach consists on the extraction of the coordinates of the skeleton joints of the pedestrians using a Pose Estimation network. Those coordinates are then usually used to classify the crossing intention using simple networks such as Multi-layer Perceptrons (MLPs). However, with the use of Graph Neural Networks (GNNs), we can leverage the complex relationships among the joints to make accurate predictions about pedestrian behavior, by treating the joints as nodes in a graph and modeling their relationships as edges. In this way, using GNNs to process this graph, we can more effectively learn the influence of joints on pedestrian intention, leading to improved prediction accuracy.

Considering this perspective, this paper presents three key contributions to the field of pedestrian crossing intention prediction. Firstly, we propose a novel GNN architecture specifically designed for this task. It consists of a Spatial-Temporal Graph Neural Network that classifies the crossing intention of pedestrians based on their 2D skeleton coordinates, capturing the complex relationships among pedestrian joints.

Additionally, since the effectiveness of GNNs in this application is highly dependent on the quality of the data, we also propose an approach for enhancing the quality of the training data for pedestrian crossing intention prediction through the use of explainability techniques to improve the dataset of extracted skeletons. Specifically, we present an explainability analysis of

Author: Abel García Romera
Advisor: Dr. Antonio Manuel Lopez Peña
Advisor department: Dpt. Ciències de la Computació and the Computer Vision Center
Advisor university: Autonomous University of Barcelona
Thesis dissertation submitted: July 2023
Code of the project available: github.com/abel-gr/PedestrianCrossingIntention

the proposed model, and subsequently, we introduce an explainability-driven data augmentation technique that generates new skeletons through modifications of the dataset, based on the insights of that explainability analysis. We demonstrate that this approach leads to slightly improvements in the classification metrics of pedestrian crossing intention prediction, as compared to not using data augmentation, and significant improvements, as compared to using completely random-based data augmentation. This is a double contribution, taking into account that there are almost no known works that apply explainability techniques to the crossing intention problem, and that there are also no papers that use data augmentation of the estimated skeletons for this task. Moreover, in this paper both ideas are combined, an approach that could have potential applications in other 2D or 3D skeleton-based tasks where deep learning models are employed.

In summary, the three proposed contributions provide an accurate method for classifying pedestrian intention to cross the street while enhancing the quality of the pedestrian skeletons used for training the network. These advancements improve the accuracy of pedestrian crossing intention prediction, which could have significant implications for autonomous vehicles and pedestrian safety.

## II. State of the art

This section is divided in two subsections. In subsection II-A, an overview of the methodologies employed by state-of-the-art papers to address the pedestrian crossing intention prediction task is provided. Subsequently, as this paper also proposes an explainability-driven data augmentation method for enhancing pedestrian skeletons, subsection II-B discusses the most relevant state-of-the-art explainability and data augmentation methods employed for the task of crossing intention prediction.

### *A. Pedestrian Crossing Intention*

Most previous works in pedestrian crossing intention use algorithms based on the visual information captured by cameras to predict the intention of pedestrians to cross or not cross the street. In [1] they propose to use only image sequences from a monocular RGB camera, integrating components such as YOLOv3 for detection and tracking, the unscented Kalman filter (UKF) for motion estimation, and the spatio-temporal DenseNet (ST-DenseNet) for intent action prediction.

In contrast, an alternative approach emerged. Rather than relying solely on visual-based information, several papers [2] [3] utilize a pose estimation network to estimate the skeleton joints of pedestrians. These skeleton joints are then fed into a Support Vector Machine (SVM) or Random Forest (RF) classifier to make the prediction. By leveraging the information provided by the skeleton joints, these methods improved the accuracy and reliability of the action prediction for pedestrians.

Another category of approaches that do not rely only on the visual information, instead of using the estimated pose, these methods incorporates the utilization of pedestrian velocities and positions to enhance the prediction accuracy. By incorporating this temporal information, these techniques can leverage the dynamic aspects of pedestrian motion, being easier to infer the intended actions. Specifically, Bouhsain et. al. [4] propose the method PV-LSTM, based on a multi-task sequence to sequence LSTM model that takes as input the velocities and coordinates of observed past bounding boxes and outputs the predicted velocities of the future bounding boxes of the pedestrian. The future bounding box coordinates and the pedestrian's state (crossing or not crossing) are computed using the LSTM decoder part. Similarly, [5] proposed also a multitask framework, in this case to simultaneously predict pedestrian trajectory, action and final location, hence also leveraging the dynamic aspects of the motion of the pedestrians.

Different studies have indicated that there are many factors that determine whether or not a person is going to cross the street, such as the age of the pedestrians, their speed, their trajectory, the distance they are from the crosswalk, the speed of the car and the distance at which the vehicle is also from the intersection [6]. However, most experts agree that the trajectory of the pedestrians and the previously mentioned information of interest is not enough, and much better results are achieved also using the key points of the human body and their coordinates, estimated by pose estimation techniques [6].

For this reason, [7] proposed to fuse different sources of information such as sequences of RGB images, semantic segmentation masks, pedestrian bounding boxes coordinates, estimated pose keypoints and the speed of the ego-vehicle, using attention mechanisms and a stack of Recurrent Neural Networks (RNNs). They utilized a pre-trained OpenPose model as the pose estimation network.

Nevertheless, recent papers [8] [9] have introduced lightweight methods that surpass the previously mentioned complex approaches in terms of classification metrics. These techniques prioritize the extraction of long-term temporal features from the estimated pose using Recurrent Neural Networks. By leveraging RNNs, these methods can effectively incorporate temporal information from several seconds leading up to the moment of crossing. This emphasis on capturing temporal context enables

more accurate intention prediction, based on the relevant features that networks extract from pedestrian motion, instead of specifically computing velocities or trajectories as the previous methods.

Specifically, TrouSPI-Net [8] extracts spatio-temporal features by encoding pseudo-images sequences of skeletal joints' positions and processes them with attention modules and atrous convolutions. They also enhance the proposed approach by combining the relative distances of skeletal joints, the bounding box positions and the ego-vehicle speed, but it is a lightweight method that outperformed the complex previous state-of-the-art techniques.

Similarly, Kotseruba et. al. proposed PCPA [9], a model for pedestrian crossing prediction with attention that uses individual RNN branches to process in parallel the estimated pose, the location of pedestrians, and the ego-vehicle speed. But PCPA also incorporates a 3D convolutional branch for encoding visual information. The reason behind this approach is that Kotseruba et. al. conducted a comprehensive evaluation of state-of-the-art models, that also revealed that the top-performing models extracted temporal features with the use of RNN and 3D convolutions [9], so they used both. Subsequently, those encoded features are combined using self-attention modules before computing the final prediction.

Finally, PedFormer [10] improves the previously discussed approach by proposing a cross-modal Transformer architecture to capture dependencies between the different data sources, and a gated hybrid LSTM decoder mechanism for producing the predictions.

### *B. Explainability and data augmentation for pedestrian crossing intention*

In recent years, explainability for deep learning has gained popularity as an area of research but its application is still very uncommon across the different fields, including Pedestrian Crossing Intention. Only a few papers have explored the use of explainability for this task. Yu Yao et. al. [11] proposes a model for predicting the crossing intention, that obtains certain interpretable results that relate the detected traffic objects in the urban scene and the pedestrian's crossing intention. This is achieved with the use of an Attentive Relation Network (ARN) that extracts information from the detected traffic objects. Therefore, it is partially explainable as it is based only on the detected objects in the scene and not on the pedestrian's movement or pose. This is due to the fact that Yu Yao et. al. were not using estimated pedestrian poses, and as noted in the previous subsection, using them can potentially increase the classification metrics and generalization capabilities of the model.

Alternatively, Chen et. al. [12], propose a model that produces text-based explanations of the driver reasoning process when estimating pedestrian intention, with sentences such as "Pedestrian is in road facing other sidewalk with intent to cross". This explainability is the easiest for a human to interpret, but it is very computationally expensive.

Similarly, there are also very few papers that use data augmentation in the crossing intention prediction task, and those works that apply it, do not perform it on the estimated skeletons of pedestrians. On the one hand, some articles apply data augmentation directly to images by applying random rotations or changes in brightness, contrast, saturation, and hue [13]. On the other hand, other papers perform data augmentation to the pedestrian trajectories, such as random rotations to each trajectory, mirror the trajectory in the x-axis or the y-axis, or apply Gaussian noise to them [14].

Up to the best of our knowledge, there are currently no existing papers that study the use of explainability analysis or data augmentation techniques for crossing intention prediction using the estimated pose of pedestrians. Hence, our proposal can be considered a novelty in the field, specially considering that it combines both ideas. Our explainability and data augmentation approaches are explained in sections VI-C and VI-D, respectively.

## III. Proposed architecture

In Fig. 1 you can see our architecture proposal that will be used in this article to solve the task of predicting whether pedestrians cross the street or not. As it has been previously mentioned, many papers agree on the importance of using pose estimation models to extract the coordinates of the skeletons of the pedestrians and then use them as input for a classifier, that outputs the crossing intention [3]. Accordingly, our approach starts from videos in which these pedestrians appear, in which they may or may not be performing the action of crossing in each frame. The skeletons of those pedestrians are extracted from these videos using a Pose Estimation neural network such as those described below in subsection III-A.

Then, the coordinates of the joints of that skeleton form the input of another neural network, this time a binary classifier that classifies between Crossing and Not-crossing based on the action performed by each pedestrian. Subsection III-B presents the architecture that we propose as classifier.

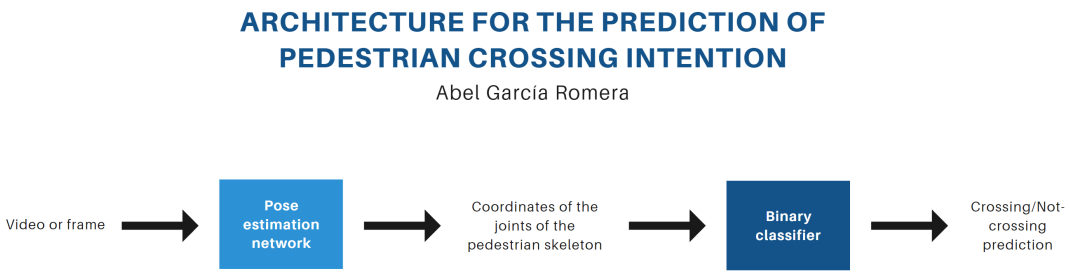



Fig. 1: Proposed architecture for the prediction of pedestrian crossing intention.

### *A. Pose estimation network*

As it can be deduced, Pose Estimation is a type of Computer Vision task in which the objective is to detect the position and orientation of a person. In general this is done by predicting the coordinates of specific key points such as shoulders, eyes, knees and many other parts of the body that are called joints [15]. Figure 2 illustrates an example skeleton with the eighteen joints corresponding to the COCO 18-points format.

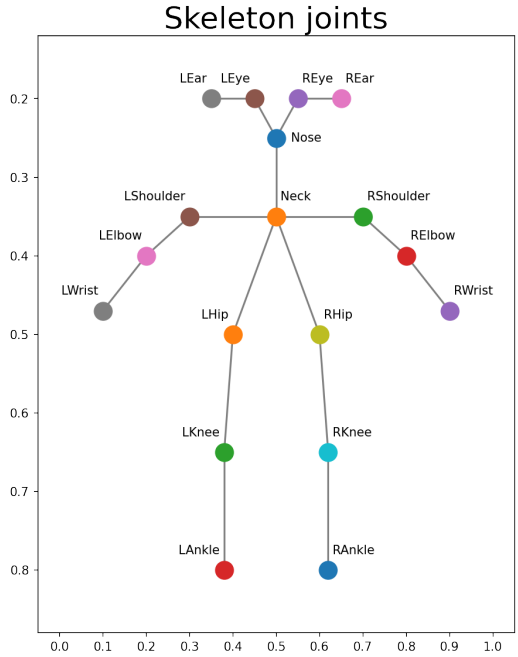



Fig. 2: Example of the eighteen skeleton joints of the COCO 18-points format.

There are chiefly two types of approaches to solve this task. The first approach is to employ a person detector that allows to locate the people in each frame, and then for each of these people a single-person pose estimator is executed. Clearly here we can see that this type of method, called top-down, suffers from the problem that its execution time is proportional to the number of people that appear in the image. Another problem with this approach is that if the person detector fails, the Pose Estimation system will not be able to extract any skeleton of the person [15].

On the contrary, the second type of approach, called bottom-up, no longer suffers from either of those two problems, since it does not rely on any person detection. That is, they offer greater robustness and eliminate the increase in execution time depending on the number of people [15]. However, these methods tend to be slower to execute, since their final stages usually require longer inference times, even taking several minutes per image in the first algorithms developed around the year 2016, such as DeeperCut [16]. This implies that, although the execution time of the method was independent of the number of people present in the image, its slower inference times led to a longer processing time compared to the aforementioned top-down approaches even when the number of persons appearing in the image was not relatively small.

However, one year later, a new bottom-up Pose Estimation method appeared, capable of extracting the skeletons of multiple

people with a time performance comparable to top-down approaches [17]. In 2018 the authors presented a subsequent paper providing further details and improvements to the original framework introduced in 2017, and they explicitly denoted their proposed approach as OpenPose [18]. Notably, OpenPose is the first real-time multi-person system to detect key points of the human body, the hand, the face, and the foot. As previously mentioned, it performs an efficient inference with less computational complexity compared to previous bottom-up algorithms, while obtaining very good results [15]. Consequently, it stands as one of the most utilized options in many papers when using Pose Estimation. Considering all these factors, OpenPose is one of the methods being contemplated for use in this article.

Nevertheless, we must also analyze some of the algorithms belonging to the top-down approach, remembering that despite the drawbacks that were previously mentioned, they provide fast results when the number of people present in the image is not large, which is useful in the context of autonomous driving in which it is necessary to provide predictions in real time to ensure that the vehicle has sufficient time to stop and prevent collisions with pedestrians. The top-down algorithm that became most famous for its versatility when it comes to solving multiple tasks such as instance segmentation, bounding box object detection and person pose estimation is Mask R-CNN [19]. At that time, around 2017, it outperformed all other existing models in all the mentioned tasks [20], including bottom-up approaches such as OpenPose, as you may see in Table I.

On the other hand, at about the same time, another architecture was being developed, also based on the top-down approach. It was called Regional Multi-Person Pose Estimation (RMPE) [21], which soon after became popular as AlphaPose [22], a new version based on RMPE, that achieved even better results than Mask R-CNN in the Pose Estimation task for the COCO 2015 dataset [22]. Therefore, AlphaPose surpassed both the OpenPose and Mask-RCNN metrics for that dataset, as you can compare in Table I.

| Method | AP@0.5:0.95 | AP@0.5 | AP@0.75 | AP medium | AP large |
|---|---|---|---|---|---|
| OpenPose (CMU-Pose) | 61.8 | 84.9 | 67.5 | 57.1 | 68.2 |
| Detectron (Mask R-CNN) | 67.0 | 88.0 | 73.1 | 62.2 | 75.6 |
| **AlphaPose** | **73.3** | **89.2** | **79.1** | **69.0** | **78.6** |

TABLE I: Pose Estimation results on COCO test-dev 2015, extracted from AlphaPose GitHub repository for academic reasons [23].

Despite these good results reported in Table I, the disadvantages of the top-down approaches must be kept in mind. One of these problems is clearly represented in Fig. 3. In this figure, it can be seen both for AlphaPose and for Mask R-CNN, that the execution times grow linearly with the number of people to estimate their pose. Instead, OpenPose's inference time is invariant to that variable. The reason this happens has already been explained earlier in this section: OpenPose is a bottom-up method, while Mask R-CNN and AlphaPose follow a top-down approach. This experiment was carried out by the creators of OpenPose in their paper and it has been included here for academic reasons to compare these algorithms [18].

Therefore, both types of approaches are interesting. That said, one of each will be used in this paper: as a top-down algorithm the modern AlphaPose will be utilized, and as a bottom-up method the selected option is the popular OpenPose. Both techniques will be used in section V to generate two datasets of pedestrian skeleton coordinates from videos of traffic environments. Those skeletons will be utilized in the final stage of the pipeline, the classifier, to produce the final output of the crossing intention. The purpose of using the two Pose Estimation networks, OpenPose and AlphaPose, is to compare the classification metrics. The goal is to determine which set of skeletons, obtained from each approach, yields the highest validation accuracy when used as input to the classifier, thus determining which method, OpenPose or AlphaPose, estimates the most useful, robust or accurate skeletons for the classification task. This experiment is done in section VI-B.

### *B. Binary classifier*

The coordinates of the skeletons of the pedestrians could be used to classify their crossing intention using a simple Multi-layer Perceptron (MLP). However, notice that we can model the body skeletons as a graph, by representing the joints as nodes and modeling their relationships as edges, allowing for a more robust capture of the relationships between the joints, since graphs are very useful for representing information that contains values that are related or connected, such as pedestrian skeletons in this case. The problem is that they are usually complicated to process through Deep Learning algorithms. For simple graphs, indeed a Multi-layer Perceptron already provides good results, but there are specific networks to work with graphs: Graph Neural Networks (GNN) [24], that offer a promising approach to accurately predict values, in this case pedestrian crossing intentions, based on graphs.

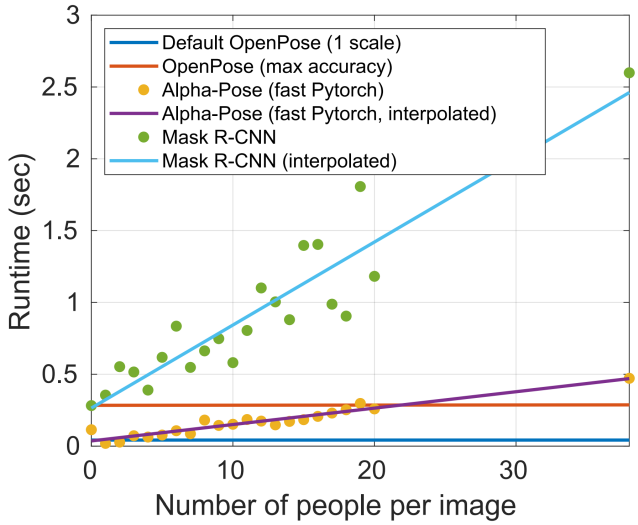



Fig. 3: Inference time comparison between OpenPose, Mask R-CNN, and AlphaPose. Plot image extracted from the OpenPose paper for academic reasons [18].

Specifically, the progress in Convolutional Neural Networks (CNN) raised the possibility of using convolutions also in graphs, due to the properties of local connectivity and shift invariance of convolutions, and how useful these properties are in a graph. In spite of that, due to the non-grid-like structure inherent in graphs, Convolutional Neural Networks can not be directly applied to process them. Therefore, two alternatives emerged to allow the use of convolutions with graphs: the spectral based graph convolutions, and the spatial based graph convolutions. The first ones are based on signal processing theory, specifically in the frequency domain rather than the spatial domain. However, instead of using the Discrete Fourier Transform they use the eigenvectors of the Laplacian graph, which needs to be calculated in each iteration, making it a computationally expensive approach [24]. Nonetheless, some more efficient frequency domain graph convolutional operators appeared such as the Chebyshev spectral graph convolutional operator [25].

On the other hand, the popular Graph Convolutional Networks (GCN) [26] work in a similar way to image convolutions, but in this case combining the information of a node and its neighbors to generate a new node as the output. Therefore, they work in the spatial domain instead of the frequency domain, being a fast process compared to the spectral method.

Nevertheless, several studies have indicated that one of the determining factors in deciding whether or not a pedestrian intends to cross is the speed with which people are moving, and what their trajectory is [3]. That information could be directly computed and added as input to the network, but that would require a change in the architecture, increasing its complexity, and the computation of the trajectory of pedestrians is not straightforward as it would require a tracking method. Alternatively, an interesting approach to include temporal information in the proposed model is, instead of using Graph Neural Networks based just on the spatial or spectral domain, to use Spatial-Temporal Graph Neural Networks (ST-GNNs) [27], whose input are now a sequence of graphs, where each graph corresponds to a different time step. The objective is to process the graphs using the same GNN operations previously described, but now also considering the time dimension. This allows the model to learn representations that capture both the spatial properties of the graphs and the temporal evolution of the data.

Mainly there are two strategies to capture the temporal dependencies of multiple graphs, either use Convolutional Neural Networks to add temporal convolutional layers or use Recurrent Neural Networks (RNN) [24]. The first approach basically consists in adding a convolutional layer to also take account temporal information, but this is usually a very simple approximation [24]. Another more advanced strategy is the use of Recurrent Neural Networks, which allow modeling sequences that vary over time. Specifically, the Recurrent Graph Convolutional Layers (RGCL) are very common, which are layers that implement the convolutional operators of graphs but in the form of a RNN, that is, they are perfect for modeling temporal sequences of graphs. For that reason, the Recurrent Graph Convolutional Layers are the type of GNNs that are used in this paper to extract the features of the pedestrian skeletons and then feed those features to a fully-connected layer to make the final prediction.

There are many types of Recurrent Graph Convolutional Layers [28], and as previously stated, all of them combine the filtering capabilities of the graph convolutional operators, implemented either in the spatial or spectral domain, with the recurrent and temporal processing capability of a RNN. An example of a RGCL that uses graph spectral filtering combined with the recurrent

update mechanism of a RNN, specifically a Gated Recurrent Unit Cell, is the Chebyshev Graph Convolutional Gated Recurrent Unit Cell (GConvGRU) [29]. Notice that it has been previously mentioned that spectral-based approaches are computationally expensive, but GConvGRU uses the Chebyshev graph convolution operator that is computationally efficient compared to the majority of spectral operators. Other examples of Recurrent Graph Convolutional Layers that implement convolutions in their own adaptation of the spectral or spatial domain, while simultaneously leveraging the capabilities of Recurrent Neural Networks, are the Temporal Graph Convolutional Gated Recurrent Cell (TGCN) [30], the Diffusion Convolutional Gated Recurrent Unit (DCRNN) [31] and the Graph Convolutional Long Short Term Memory Cell (GCLSTM) [32], to name a few. In section VI-B several layers are tested to determine which one produces the best validation results.

In Table II you can see the different layers of our architecture proposal for the Pedestrian Crossing Intention binary classifier, with the input and output shapes of each layer. The input of the network is a batch of shape $(B_s, N_g, N_n, N_f)$. $B_s$ is the batch size. $N_g$ means the number of graphs and it is the amount of $N$ skeletons of each pedestrian you want to use to make the prediction. That is, the amount of previous frames desired to be taken into consideration to generate the classification output. $N_n$ is the number of nodes, that is always 18 since it was the amount of joints when we extracted the skeletons using the Pose Estimation networks. $N_f$ is the number of node features. It is always three since each input node contains three values: the $x$ and $y$ coordinates of that pedestrian joint, and the confidence score $c$ of that joint.

| Layer | Input shape | Output shape |
|---|---|---|
| Recurrent part | $[B_s, N_g, 18, 3]$ | - |
| RGCL | $[B_s, 18, 3]$ | $[B_s, 18, 3]$ |
| Dropout | $[B_s, 18, 3]$ | $[B_s, 18, 3]$ |
| ReLU | $[B_s, 18, 3]$ | $[B_s, 18, 3]$ |
| Non-recurrent part | $[B_s, 18, 3]$ | - |
| Reshape | $[B_s, 18, 3]$ | $[B_s, 54]$ |
| Linear | $[B_s, 54]$ | $[B_s, 37]$ |
| Dropout | $[B_s, 37]$ | $[B_s, 37]$ |
| ReLU | $[B_s, 37]$ | $[B_s, 37]$ |
| Linear | $[B_s, 37]$ | $[B_s, 18]$ |
| Dropout | $[B_s, 18]$ | $[B_s, 18]$ |
| ReLU | $[B_s, 18]$ | $[B_s, 18]$ |
| Linear | $[B_s, 18]$ | $[B_s, 2]$ |
| Softmax | $[B_s, 2]$ | $[B_s, 2]$ |

TABLE II: Layer input and output shapes of our architecture proposal for the Pedestrian Crossing Intention binary classifier. RGCL stands for any Recurrent Graph Convolutional Layer of the ones used in section VI-B. $B_s$ is the batch size. $N_g$ is the number of graphs for each pedestrian, that is equivalent to the amount of previous frames you want to take into consideration to generate the classification output.

As it can be seen in Table II, the skeletons are first passed through a recurrent part in which each pedestrian skeleton of each frame is fed recurrently into any of the Recurrent Graph Convolutional Layers used in section VI-B, and each output hidden state is fed to a Dropout layer and a ReLU activation function. Then, that result is fed again to the Recurrent Graph Convolutional Layer for the skeletons of the next frame, and this is done for each skeleton of the pedestrian, corresponding to the last $N_g$ frames, that is $N_g$ skeletons for each pedestrian to be processed in this recurrent part. Then, after the last graph is processed by this recurrent part, the latest hidden state is flattened and passed to a fully-connected network that acts as the classifier. This fully-connected network is not recurrent. It has a final output layer consisting of a Softmax operation, that outputs the final probability of the pedestrian to be crossing the street or not in that specific frame. Remember that the input of the network is a set of skeletons in several frames, not just one frame, since they are the last $N_g$ frames before the actual frame in which we want to perform the prediction of the crossing intention. Note that our proposed network can be trained end-to-end since the skeletons that consist of the input of the model, have been previously estimated through the Pose Estimation networks detailed in subsection III-A.

After training the GNN, we used some explainability techniques in order to enhance the training dataset. Specifically, we generated new skeletons with the aim of further improving the performance of the classifier. This is explained and done in sections VI-C and VI-D.

## IV. Dataset

The dataset used in this work is the Joint Attention in Autonomous Driving dataset, commonly known as JAAD. It is a widely utilized dataset in the field of autonomous driving, particularly in research papers focusing on pedestrian behavior, studying their crossing intention and the factors that influence it [15]. Hence, it is an ideal choice for the task at hand, as it provides ground truth annotations for pedestrian actions (crossing and not-crossing) in each frame, enabling its use in the binary classification task addressed in this article. Furthermore, the great popularity of JAAD allows the comparison of results with those achieved by other researchers in their respective papers.

The JAAD dataset contains 346 short videos between five and ten seconds in length that were extracted from almost 240 hours of driving footage recorded in several locations in North America and Eastern Europe, representing common daily urban driving actions in different climatic conditions [15]. This means that they are not only real videos in real situations and roads, but also that they are very varied scenes to demonstrate that our method is indeed capable of generalizing the prediction of the crossing intention in different kinds of situations that occur in the real world.

In Fig. 4 you can see examples of two frames from two different videos of JAAD dataset.

(a) (b)

Fig. 4: Example of two frames from two videos of the JAAD dataset: (a) Video 231; (b) Video 157

Each video contains annotations with the behavior and actions of pedestrians such as crossing, walking, standing, looking at the car, etc. There is also extra information about pedestrians such as their age and gender, as well as other attributes related to the environment such as weather, traffic signs and so on [15]. The creators of JAAD themselves provide a code through GitHub to parse and extract the *XML* files with the annotations of each video. We have modified that code to convert and save the result to a *csv* file with an easier format, so it can be read quickly and directly through the *pandas* library [33], which is comfortable and efficient when working with datasets. They also provide the identifiers of the videos that belong to each subset (train, test, and validation) so that all the papers can be compared under the same conditions. We therefore used these identifiers to split the *csv* into three other *csv* files for each of the three mentioned subsets. The train subset contains 61805 samples, the test *csv* contains 52966 rows and finally the validation set only 9583.

In Table III you can find some basic statistics about the JAAD dataset that they themselves provide so that people can get an idea of its magnitude [15]. Conversely, Table IV displays the number of pedestrians, among those with crossing annotations, who either cross or do not cross the street. There is a significant imbalance, with a larger number of pedestrians crossing compared to those who do not, a fact that will imply using various metrics when evaluating the result of the crossing classification, and not only the accuracy.

Fig. 5 and Table V show the total number of pedestrian annotations for each subset, as well as for each class (crossing and not-crossing) separately. It can be seen that this imbalance between classes appears in the three subsets, with more samples of pedestrian annotations that cross in each frame than pedestrians who do not cross. For this reason, as previously mentioned,

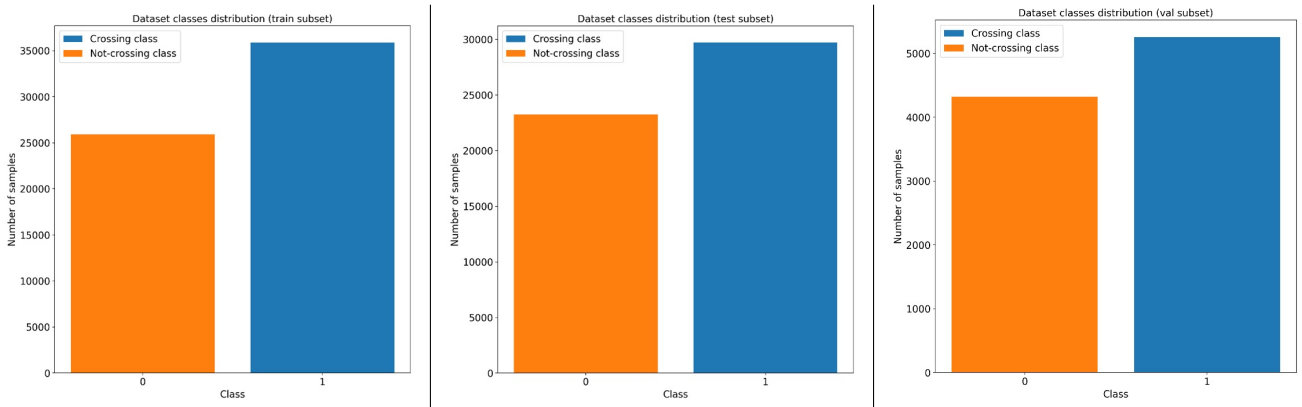
| | |
|---|---|
| Total number of frames | 82,032 |
| Total number of annotated frames | 82,032 |
| Total number of pedestrians | 2,786 |
| Number of pedestrian bounding boxes | 378,643 |
| Number of pedestrians with behavior annotations | 686 |
| Average length of pedestrian track in frames | 121 |

TABLE III: JAAD dataset stats [15].

| | |
|---|---|
| Number of pedestrians who cross the street | 495 |
| Number of pedestrians who do not cross the street | 191 |

TABLE IV: JAAD dataset pedestrian counts [15].

several metrics will be used in section VI-B to evaluate the classification results of this crossing variable, and not only accuracy, as is usually done.




(a) (b) (c)

Fig. 5: Distribution of the classes using the crossing annotations as the class target: (a) Train set; (b) Test set; (c) Validation set

| | Train | Test | Val |
|---|---|---|---|
| Crossing | 35,889 | 29,723 | 5,258 |
| Not-Crossing | 25,916 | 23,243 | 4,325 |
| Total | 61,805 | 52,966 | 9,583 |

TABLE V: Number of samples of each class for each of the three subsets.

The videos of this dataset are the input to the pipeline of the architecture proposed in section III.

## V. Pose Estimation

As it has been mentioned at the end of section IV, the videos of the JAAD dataset form the input of the pipeline proposed in section III. They are therefore the input to the Pose Estimation network that extracts the skeleton of each pedestrian from the videos. In this section, the process of utilizing these networks to estimate pedestrian skeletons is initially explained in subsection V-A. Subsequently, in subsection V-B, the method of combining these skeletons with the crossing and not-crossing ground truth from the dataset of section IV is described. This combination is required to use the estimated skeletons to train the crossing intention classifier in section VI.

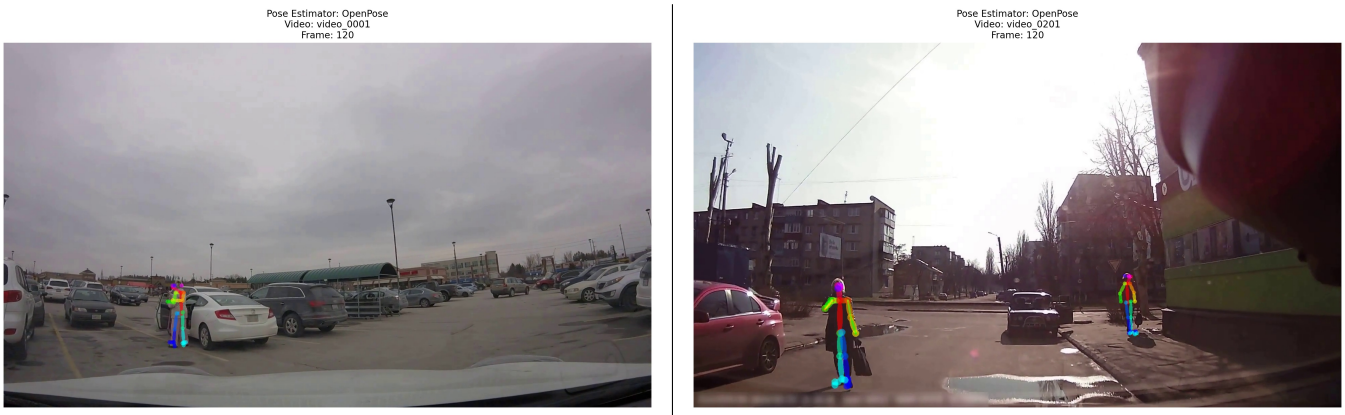
### A. *Estimation of the skeletons*

In accordance with the explanation provided in section III-A, both OpenPose and AlphaPose will be used in this paper as Pose Estimators, to be able in the final stage of the pipeline to compare the classification metrics between crossing and not-crossing when using the skeletons obtained by both approaches.

The developers of the two solutions have already provided pre-trained models as they require long training times. To simplify the networks installation process, the estimation of pedestrian skeletons in JAAD videos using both pose estimators has been performed on Google Colab.

*1) Using OpenPose:* OpenPose main functionality is the 2D real-time multi-person keypoint detection. It offers the option to estimate 15, 18, or 25 keypoints for the human body, and the execution time is invariant to the number of people appearing in the image. The input can be directly a video, so it is easy to use. It generates two outputs, a visually rendered video with the estimated joints such as the shown in Fig. 6, and a set of *JSON* files, one for each frame. Each *JSON* file contains several values about that frame, with the most important being the *people* array of objects. Each object within the array contains sub-objects related to the pose estimation. Among those sub-objects, the most relevant is the *pose-keypoints-2d* that contains a list of the estimated body part coordinates $(x, y)$ and the confidence $(c)$ for that joint, formatted as $[x_0, y_0, c_0, x_1, y_1, c_1, ...]$.

OpenPose takes only 16.6 seconds per video on average to estimate the pose, so it is capable of getting quite fast results in scenes with several people, and the estimated poses are accurate in many different situations and varied rotations of the pedestrians, as you can check in Fig. 6.




(a) (b)

Fig. 6: Example of using OpenPose in two frames of two videos of the JAAD dataset: (a) Video 1; (b) Video 201

Despite achieving visually good results overall, OpenPose occasionally detects people of extremely large sizes, even surpassing humanly possible proportions, and in locations where the presence of pedestrians is almost impossible, as the reader will agree after looking at the images shown in Fig. 7.

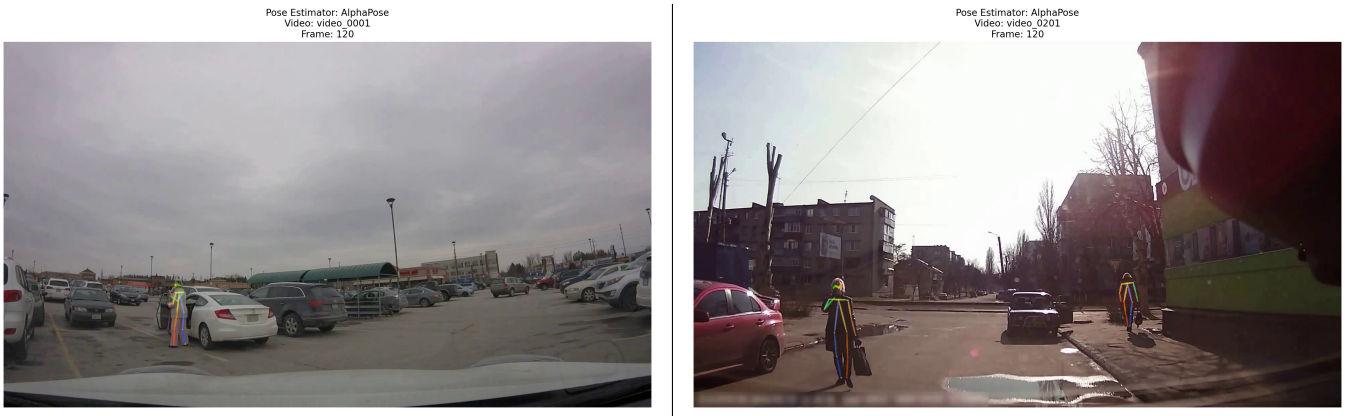
*2) Using AlphaPose:* Since AlphaPose is a top-down type pose estimator, it requires a network that is responsible for detecting people as its first stage. In this case AlphaPose uses YOLOv3. Afterwards, the result of the YOLOv3 detection, for each person detected by it, is passed to the FastPose network, designed by the creators of AlphaPose and which has a ResNet50 as its backbone. All the models provided by them are already trained on the images of *COCO key-point train 2017* dataset [34], which contains a total of 64115 images.

Similar to the approach in the previous subsection, AlphaPose method can also take the JAAD videos directly as input thus being also easy to use. And again, it produces as output both a video where the estimated keypoints are visually rendered, and *JSON* files with the coordinates of the keypoints corresponding to the joints of the detected people. In Fig. 8 we can observe the result of this pose estimation with AlphaPose, rendered on the same videos from the JAAD dataset that were examined in the preceding subsection with OpenPose. Visually, the results seem more accurate, but the average execution time per video has increased to 45.4 seconds, compared to 16.6 seconds with OpenPose.

*3) Comparison:* In Table VI we can see a comparison of the mean execution time per video to estimate the pose of the pedestrians averaged over all videos. The average inference time per video with AlphaPose is 45.4 seconds, compared to 16.6


(a) (b)

Fig. 7: Example of using OpenPose in two frames of two videos of the JAAD dataset: (a) Video 101; (b) Video 202




(a) (b)

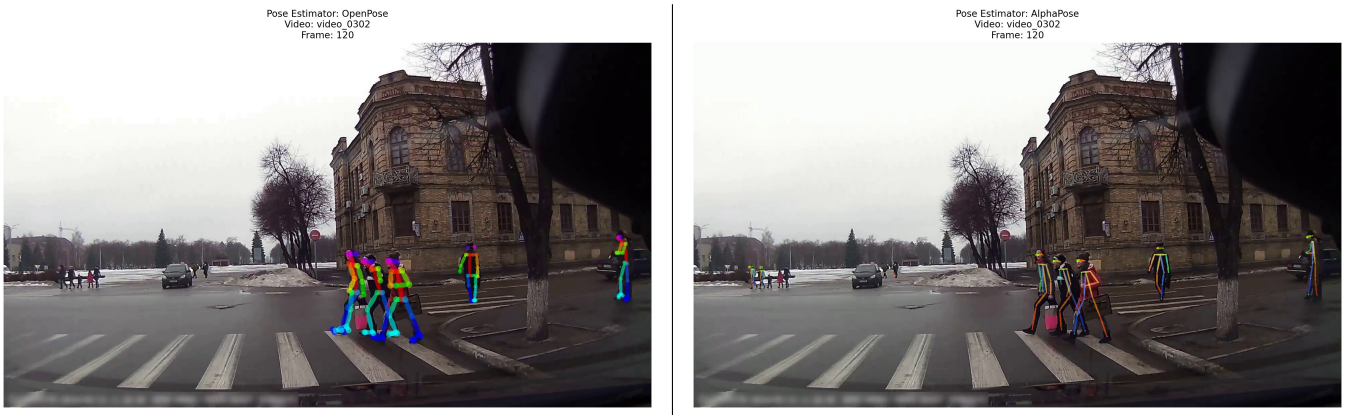
Fig. 8: Example of using AlphaPose in two frames of two videos of the JAAD dataset: (a) Video 1; (b) Video 201

seconds with OpenPose. This is due to the already explained underlying theory of both approaches. As discussed in section III-A, AlphaPose needs a first stage of people detection, followed by a second stage of pose estimation that is performed individually for each detected person. Consequently, the inference time of AlphaPose is dependent on the number of detected people, and most videos contain multiple pedestrians. On the other hand, OpenPose is independent of this variable and highly optimized, resulting in a lower inference time compared to AlphaPose.

| Pose estimator | Mean execution time |
|---|---|
| OpenPose | 16.6 seconds |
| AlphaPose | 45.4 seconds |

TABLE VI: Mean execution time per video to estimate the pose of the pedestrians averaged over all JAAD videos, both executed in standard Google Colab.

At a visual level, there are many situations in which both pose estimators give very similar results, but AlphaPose is always capable of giving visually more precise skeletons regarding the pose that the pedestrian is performing, as can be seen in Fig. 9. Moreover, in this figure it can be appreciated a fact that can be found in the vast majority of processed videos of the dataset: AlphaPose is capable of estimating the skeleton of pedestrians that are very far away as can be seen on the left part of image (b) in Fig. 9, but OpenPose is not capable of doing this as can be observed on the left part of (a) in Fig. 9. This capability is not crucial for this work since for an autonomous car that aims to predict the crossing intention, just the nearby pedestrians who may pose a danger need to be detected. However, it is interesting to mention that this capability arises from the fact that AlphaPose employs a top-down approach, that, as already mentioned, incorporates a first stage of people detection using YOLOv3, which yields highly accurate results in object detection, detecting even people who are far away.




(a) (b)

Fig. 9: Pose Estimation result for video 302 of JAAD dataset: (a) Using OpenPose; (b) Using AlphaPose

On the other hand, in Fig. 10 and Fig. 11 another strong point of AlphaPose can be observed. It avoids the OpenPose mistake of misidentifying gigantic pedestrians in places where there are trees, lampposts or roofs. Once again, this is attributed to the utilization of YOLOv3 in the first stage of AlphaPose's internal pipeline to detect the people in the frame.


(a) (b)

Fig. 10: Pose Estimation result for video 101 of JAAD dataset: (a) Using OpenPose; (b) Using AlphaPose


(a) (b)

Fig. 11: Pose Estimation result for video 202 of JAAD dataset: (a) Using OpenPose; (b) Using AlphaPose

Unfortunately, the JAAD dataset does not have a ground truth of the pedestrian skeletons, so we cannot verify quantitatively which of the two Pose Estimation methods has better estimated the pose. However, this is only the first step of the pipeline, and since the skeletons estimated by both techniques are used in the final stage of the classification, is in that final step where both techniques are numerically compared, using the classification metrics. Despite the results of AlphaPose appearing visually more accurate than those of OpenPose, in section VI-B it is evaluated which of the two networks estimate the skeletons that produce higher classification metrics. Comparing those results, that section determines whether or not AlphaPose's higher runtime is worth it.

### *B. Combining estimated skeletons with dataset annotations*

Remember from section IV that we have three *csv* files that contain the information of the behavior of the pedestrians that appear in the videos of the JAAD dataset divided in train, test, and validation subsets. It is needed to combine those annotations with the skeletons estimated in the previous subsection by each of the two pose estimators. Each pose estimator produce a different set of skeletons for each of the train, test and validation subsets, creating a total of six new *csv* files that combine both the joints and the annotations of the dataset. These are the *csv* files that are used in the classification task in section VI.

Importantly, JAAD can contain annotations about the behavior, and the crossing ground truth, for more than one pedestrian in the same frame of the same video. Therefore, to be able to relate the information provided by the JAAD dataset to the skeletons that were estimated from the videos in the previous subsection, it is required to relate each skeleton to its corresponding pedestrian in the dataset or its annotations will correspond to those of another person.

In order to address this issue, we can take advantage of the annotation's information itself. The original dataset *XML* files also contain the bounding box of each pedestrian. In this way, we calculate the center of those bounding boxes of each pedestrian and then we also compute the center of the skeletons detected by the pose estimator. Subsequently, we calculate the Euclidean distance from each skeleton center to each bounding box center for the same frame of the same video. By determining which skeleton has the smallest distance to each bounding box, it can be determined which is the skeleton detected by the pose estimator that corresponds to the bounding box of that pedestrian found in the JAAD dataset annotations, assigning that skeleton to its corresponding row of the *csv* files that were generated in section IV.

In Fig. 12 is shown the result of this process for frame 0 of video 1 with the OpenPose (a) and AlphaPose (b) skeletons. In both cases, each skeleton is assigned to the correct pedestrian of the dataset, since the estimated skeleton that is closest to its bounding box is assigned, as previously explained.


(a) (b)

Fig. 12: Center of the bounding boxes and the estimated skeletons of the pedestrians appearing in frame 0 of JAAD video 1: (a) Using OpenPose skeletons; (b) Using AlphaPose skeletons

The explained strategy works correctly even when there are many pedestrians, very close together or very far apart, as can be seen in Fig. 13 again for OpenPose (a) and AlphaPose (b). In the case of AlphaPose, all the estimated skeletons are assigned to the correct pedestrian of the ground truth. In the case of OpenPose, one of the skeletons was not estimated by the network, but that is unrelated to this process since the other skeletons that were estimated are always assigned to the correct pedestrian

of the annotations.


(a) (b)

Fig. 13: Center of the bounding boxes and the estimated skeletons of the pedestrians appearing in frame 60 of JAAD video 302: (a) Using OpenPose skeletons; (b) Using AlphaPose skeletons

In Fig. 13 it may be noticed that, especially in the case of AlphaPose, in some frames more skeletons are detected than those shown in the legend of the figure, and therefore they are not assigned to any pedestrian in the dataset. These skeletons are automatically ignored by using the explained process, since the JAAD dataset does not contain annotations on them. Consequently, the skeletons corresponding to these pedestrians cannot be used to train the classifier detailed in section III-B since no corresponding ground truth information exists for them in the dataset.

This process of using information from the annotations is not considered cheating since it is only done for the purpose of relating the estimated skeletons with the ground truth of the dataset's crossing intention, in order to train the network and calculate the metrics resulting from the intention classification. To use the proposed architecture in the real world after the classifier is trained, this process would not be necessary because the ground truth will no longer be required, since the prediction of the crossing intention would be performed directly after estimating the skeletons of the pedestrians that appear in the camera of the car. However, in the case of a real car, the pedestrians have to be tracked in order to include their skeletons in the classifier. Both OpenPose and AlphaPose offer their own trackers, but for this work, the use of the tracker is being avoided since, as mentioned, it has not been necessary due to the process described in this subsection, as well as to prevent possible tracker problems from interfering in this study.

## VI. Experiments

In this section the experiments are presented, first of all, in the subsection VI-A, related to a quantitative analysis of the quality of the skeletons estimated by each pose estimator. Secondly, the estimated skeletons are used in subsection VI-B as input to the classifier defined in section III-B, to perform experiments related to the crossing intention prediction task itself, to decide the best architecture, layers, and hyperparameters. Thirdly, in subsection VI-C, is described the explainability analysis that is performed to determine which are the most critical skeleton joints for the network to produce the prediction. Finally, subsection VI-D leverages that analysis to generate more meaningful and varied training skeletons by proposing an explainability-driven data augmentation method. All networks trained in this section have been trained using an NVIDIA GeForce RTX 3090 GPU.

### A. Quantitative analysis of pose estimation

Despite the lack of ground truth of the skeletons, after combining in section V-B the results of the pose estimation with the pedestrian annotations, it is possible to determine the number of pedestrians that have been detected by each pose estimation network. In Table VII it can be seen that OpenPose did not detect 6.43% of the pedestrians that contain crossing/not-crossing annotations in JAAD dataset, among all the frames of all the videos, while AlphaPose did not detect only 1.25% of them. Note that in section IV it was indicated that the dataset contains an extremely lower number of pedestrians, since in Table VII each person in each frame is considered a separate pedestrian even though they are the same pedestrians moving. The reason for this is to verify the actual number of times the network fails to detect people, since the pose estimation task is performed

| | OpenPose | AlphaPose |
|---|---|---|
| Total number of detected pedestrians | 116,358 (93.57%) | 122,801 (98.75%) |
| Total number of undetected pedestrians | 7,996 (6.43%) | 1,553 (1.25%) |
| Total number of pedestrians annotations | 124,354 (100.00%) | 124,354 (100.00%) |

TABLE VII: Comparison of the total number of pedestrians with crossing/not-crossing annotations, and the number of detected pedestrians (estimated skeletons) by OpenPose and AlphaPose among all frames of all videos of JAAD dataset.

for each frame separately.

Moreover, it must also be considered that there are pedestrians that were detected, but in which not all the joints of the body were estimated, either because there are partial occlusions or because the pose estimator was not able to identify them due to the distance of the camera to the pedestrian, the visibility conditions, the weather, or the lack of ability to generalize to videos from this dataset.

For this reason, in Table VIII is shown a summary that indicates, for each range of percentages, in how many skeletons that percentage of joints were not detected, for each pose estimator. AlphaPose shows really remarkable results by detecting and estimating all the joints of the body for 98.51% of the pedestrians in the dataset. However, OpenPose is only able to achieve this in 21.96% of the cases.

In addition, in the case of OpenPose, 46361 (37.29%) of the estimated skeletons have between 0% (non-inclusive) and 10% of undetected pedestrian body joints. On the other hand, for AlphaPose this number is only 294, which remarkably represents only 0.24% of the total number of pedestrians with annotations in the dataset.

| Missing joints (%) | OpenPose | AlphaPose |
|---|---|---|
| 0 | 27,314 (21.96%) | 122,507 (98.51%) |
| (0, 10] | 46,361 (37.29%) | 294 (0.24%) |
| (10, 20] | 25,017 (20.10%) | 0 (0.00%) |
| (20, 30] | 5,405 (4.35%) | 0 (0.00%) |
| (30, 40] | 5,852 (4.71%) | 0 (0.00%) |
| (40, 50] | 3,115 (2.50%) | 0 (0.00%) |
| (50, 60] | 1,862 (1.50%) | 0 (0.00%) |
| (60, 70] | 577 (0.46%) | 0 (0.00%) |
| (70, 80] | 352 (0.28%) | 0 (0.00%) |
| (80, 90] | 481 (0.39%) | 0 (0.00%) |
| (90, 100) | 21 (0.02%) | 0 (0.00%) |
| 100 | 7,996 (6.43%) | 1,553 (1.25%) |
| Total | 124,354 (100.00%) | 124,354 (100.00%) |

TABLE VIII: Comparison of missing joints in the skeletons estimated by OpenPose and AlphaPose: For each range of percentages, number of skeletons estimated by each pose estimator in which as many joints percentage as indicated by the interval, were not detected.

Additionally, if we look at Table IX that contains larger intervals, we will see that, using OpenPose, 85750 skeletons have between 0% (non-inclusive) and 50% undetected joints. This represents a percentage of 68.96% of the pedestrians in the dataset, which is very high. In contrast, in that interval with AlphaPose there are still only 294 skeletons that represents the 0.24% of the pedestrians. Furthermore, there are no cases of skeletons with more than 50% undetected joints with AlphaPose (if we do not consider the undetected skeletons, that have 100% of missing joints). However, while using OpenPose, 3293 skeletons have between 50% and 100% undetected nodes. Moreover, it can be seen in Table IX that for OpenPose the case

with the biggest percentage (68.96%) is when there are between $(0, 50]$ percent of missing body parts, while in AlphaPose the dominant percentage of skeletons (98.51%) are detected with all their joints without missing any.

| Missing joints (%) | OpenPose | AlphaPose |
|---|---|---|
| 0 | 27,314 (21.96%) | 122,507 (98.51%) |
| (0, 50] | 85,750 (68.96%) | 294 (0.24%) |
| (50, 100) | 3,293 (2.65%) | 0 (0.00%) |
| 100 | 7,996 (6.43%) | 1,553 (1.25%) |
| Total | 124,354 (100.00%) | 124,354 (100.00%) |

TABLE IX: Comparison of missing joints in the skeletons estimated by OpenPose and AlphaPose: For each range of percentages, number of skeletons estimated by each pose estimator in which as many joints percentage as indicated by the interval, were not detected.

Table X contains the same information than Table IX, but in this case without percentage intervals, but directly the number of missing joints, and also without including the undetected skeletons (that would have all joints missing). In this way, it can be observed another notable aspect of AlphaPose: the number of missing joints is at most one on each pedestrian, in contrast to OpenPose. Furthermore, it is very remarkable that in AlphaPose this fact only occurs in 0.24% of the cases. In the other 99.76% of detected pedestrians, the skeletons are recovered with all their joints. Conversely, when using OpenPose, the amount of estimated skeletons that have more than one missing joints is very large. Even the amount of estimated poses with more than five missing joints is very large (15.18%).

| Number of missing joints | OpenPose | AlphaPose |
|---|---|---|
| 0 | 27,314 (23.47%) | 122,507 (99.76%) |
| 1 | 29,631 (25.47%) | 294 (0.24%) |
| 2 | 16,730 (14.38%) | 0 (0%) |
| 3 | 15,062 (12.94%) | 0 (0%) |
| 4 | 5,618 (4.83%) | 0 (0%) |
| 5 | 4,337 (3.73%) | 0 (0%) |
| More than 5 | 17,666 (15.18%) | 0 (0%) |
| Detected pedestrians | 116,358 (100.00%) | 122,801 (100.00%) |

TABLE X: Comparison of missing joints in the skeletons estimated by OpenPose and AlphaPose: Number of skeletons estimated by each pose estimator in which as many joints as indicated, were not estimated. Undetected skeletons that would have all joints missing are not included.

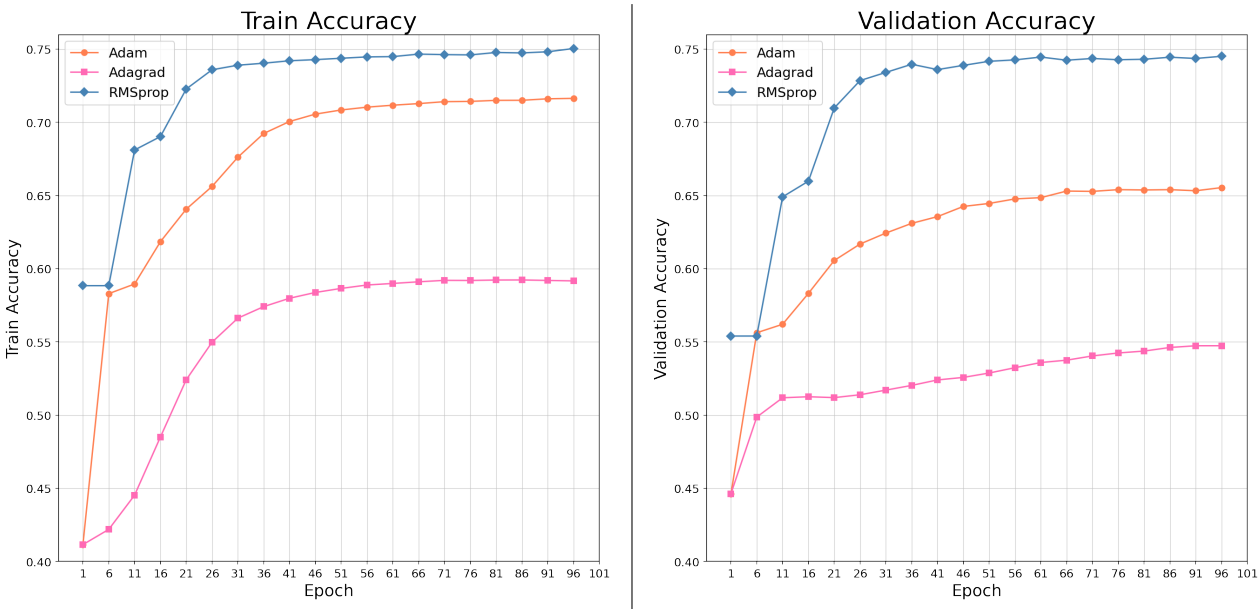
The aforementioned observations indicate not only that AlphaPose is capable of detecting nearly all the pedestrians in the dataset (98.75%), but also that it captures more robust skeletons, with 98.51% of the total pedestrians in the dataset estimated without any missing joints. That is the 99.76% of the skeletons that it estimates. OpenPose, on the other hand, detects 93.57% of the annotated pedestrians, with only 21.96% of them estimated without missing joints (23.47% of the people it detects).

However, even though AlphaPose estimates almost all skeletons, and with all or all but one of its joints, that does not mean that the estimated poses are more accurate than those of OpenPose. As visually shown in section V-A, it can be deduced that the results of AlphaPose are indeed more accurate and fit more to the real pose of people, but as mentioned in that section, the JAAD dataset does not contain ground truth of the skeletons, so it can not be verified which network produces better poses. Nevertheless, in section VI-B the comparison of the metrics of the classification between crossing and not-crossing using the skeletons of both pose estimators is the way to determine which network best estimates pose, at least for this task.

### *B. Crossing intention classification*

In this subsection of experiments, the architecture proposed in section III-B was employed. It is important to note that the proposed architecture comprises an initial set of Recurrent Graph Convolutional Layers. Four variations of these layers were used to determine the optimal choice for addressing this problem. These four layers tested are GConvGRU [29], GConvLSTM [29], TGCN [30], and GCLSTM [32]. Initially, the Diffusion Convolutional Gated Recurrent Unit (DCRNN) [31] was also considered, but after several experiments it was discarded from the Grid Search due to its extremely large training time and lower validation metrics than the other previously mentioned layers. Specifically, the training time per epoch using DCRNN was 20 times longer than using any of the other four layers. DCRNN captures the spatial dependency using bidirectional random walks on the graph, and since each random walk involves sampling and traversing multiple nodes, the computational cost of this method is high. All those layers are available in the Pytorch Geometric Temporal [35] module, and those implementations were used [28].

Prior to assessing the performance of each one of these layers when extracting skeleton graph features, a comparative analysis of different optimizers was conducted to identify the one that yielded the best validation metrics across all four layers. The three optimizers used in this first experiments are Adam, Adagrad, and RMSprop. The learning process in terms of accuracy for both training and validation subsets when utilizing the GConvGRU as the Recurrent Graph Convolutional layer can be observed in Figure 14 for Adam, Adagrad, and RMSprop optimizers. The optimal learning rate has been found and used for each optimizer. The same behavior was observed when employing the other three graph layers. Consequently, it was determined that RMSprop obtained the highest validation metrics when using any of the four graph layers. Indeed, RMSprop is a commonly used optimizer in Recurrent Neural Networks, and those graph layers are Recurrent Graph Convolutional Layers. Hence, RMSprop will be employed as the optimizer for all subsequent experiments and training procedures.




(a) (b)

Fig. 14: Accuracy during training for the train and validation subsets using GConvGRU [29] as Recurrent Graph Convolutional Layer: (a) Train subset; (b) Validation subset

After choosing the best optimizer for the four graph layers, it was needed to determine which of these Recurrent Graph Convolutional Layers performed better in terms of validation metrics. The next step consisted on run many executions of our proposed architecture, just changing the Spatio-Temporal Graph Convolutional layer used, the amount of skeletons of the same pedestrian (in consecutive frames) that are used as input of that recurrent graph layer, the dropout value used, and the Pose Estimation network used to extract the skeletons that are fed to our proposed network. Specifically, the values that are used in this Grid Search are shown in Table XI. All the 960 possible combinations of hyperparameters have been used to train the classifier described in section III-B to check the validation metrics.

Upon the execution of the Grid Search with the 960 combinations of parameters described in Table XI, Table XII shows a summary of the hyperparameter configuration of the classifiers that obtain the highest validation accuracy if we use the skeletons of each pose estimator as input. We can see in that table that the best validation accuracy if we use OpenPose as

| Hyperparameter | Values used in Grid Search |
|---|---|
| Input skeletons (Pose estimator used) | OpenPose, AlphaPose |
| Temporal info. (Number of input frames) | $\{n \mid n \in [3, 120],\ n \equiv 0 \,(\mathrm{mod}\, 3)\}$ |
| Graph layer | GConvGRU, GConvLSTM, TGCN, GCLSTM |
| Dropout | 0.3, 0.5, 0.7 |

TABLE XI: Hyperparameters of the classifier that are used in the Grid Search to find the optimal configuration.

pose estimator is 0.7668, while the best validation accuracy of the classifier if we use the skeletons estimated by AlphaPose is 0.8071. This is consistent with the quantitative analysis in section VI-A that showed that OpenPose suffers from the problem that a large number of skeletons have a high percentage of undetected joints. Moreover, in Table XII we can see how the validation f1-score using AlphaPose is also higher, 0.8609, while with OpenPose it is 0.8248.

| Pose estimator | Optimal classifier hyperparameters | | | Classifier training and validation metrics | | | |
|---|---|---|---|---|---|---|---|
| | | | | Train subset | | Validation subset | |
| | Graph Layer | Temporal Info | Dropout | Accuracy | f1-score | Accuracy | f1-score |
| OpenPose | GCLSTM | 81 | 0.3 | 0.7714 | 0.8376 | 0.7668 | 0.8248 |
| AlphaPose | TGCN | 87 | 0.5 | 0.7588 | 0.8403 | 0.8071 | 0.8609 |

TABLE XII: For each set of skeletons estimated by each pose estimator, the hyperparameters that have given the best **validation accuracy** using these skeletons as input to the classifier proposed in section III-B are shown. Both classifiers have been trained for 100 epochs.

Similarly, Table XIII shows the hyperparameters of the classifier, this time obtaining the highest validation f1-score, again using each of the two pose estimators to generate the input skeletons. The classifier configuration that gets the highest validation f1-score when using the skeletons estimated by AlphaPose as input gets 0.8609 validation f1-score. In fact, it is the same hyperparameter setting that gets the highest validation accuracy using AlphaPose skeletons and that we saw in Table XII. In contrast, the model that gets the highest validation f1-score when using the OpenPose skeletons only achieves 0.8373 f1-score. Given that AlphaPose-estimated skeletons achieve higher validation metrics, the increased execution time of AlphaPose in comparison to OpenPose that was indicated in section V has been proven worthwhile.

Therefore, all of the above indicates that the best classifier model is the one with the parameters indicated in the last row of Table XIII (the same as the last row of Table XII). That is, the classifier that uses the skeletons estimated by AlphaPose as input, and that processes them using the TGCN layer as Recurrent Graph Convolutional Layer. Furthermore, it uses 87 frames as temporal information, that is, 87 consecutive frames as input for each pedestrian, using a sliding window with a frame step of one. It uses a dropout value of 0.5 on all the dropout layers that were listed in the proposed architecture in section III-B. From now on these hyperparameters will always be used in all subsequent experiments in all the following sections or subsections, as well as in the final results of section VII.

| Pose estimator | Optimal classifier hyperparameters | | | Classifier training and validation metrics | | | |
|---|---|---|---|---|---|---|---|
| | | | | Train subset | | Validation subset | |
| | Graph Layer | Temporal Info | Dropout | Accuracy | f1-score | Accuracy | f1-score |
| OpenPose | GConvGRU | 87 | 0.3 | 0.7235 | 0.8275 | 0.7579 | 0.8373 |
| AlphaPose | TGCN | 87 | 0.5 | 0.7588 | 0.8403 | 0.8071 | 0.8609 |

TABLE XIII: For each set of skeletons estimated by each pose estimator, the hyperparameters that have given the best **validation f1-score** using these skeletons as input to the classifier proposed in section III-B are shown. Both classifiers have been trained for 100 epochs.

Nevertheless, continuing with the previously mentioned Grid Search, in Fig. 15 a plot is shown with the result of the search for the models that use the OpenPose skeletons as input. Specifically, the validation accuracy is shown based on the number of frames used by each classifier in its input. The dropout is not shown, because the accuracy of the model with the highest

validation accuracy is plotted for each value of temporal information (frames) regardless of its dropout value.

In Fig. 15 it can be seen that no Recurrent Graph Convolutional Layer stands out above the rest in all the models or values of temporal information. Depending on the number of input frames, the graph layer that allows obtaining the highest validation accuracy is different. The same occurs when using the skeletons estimated by AlphaPose, in the plot of Fig. 16.

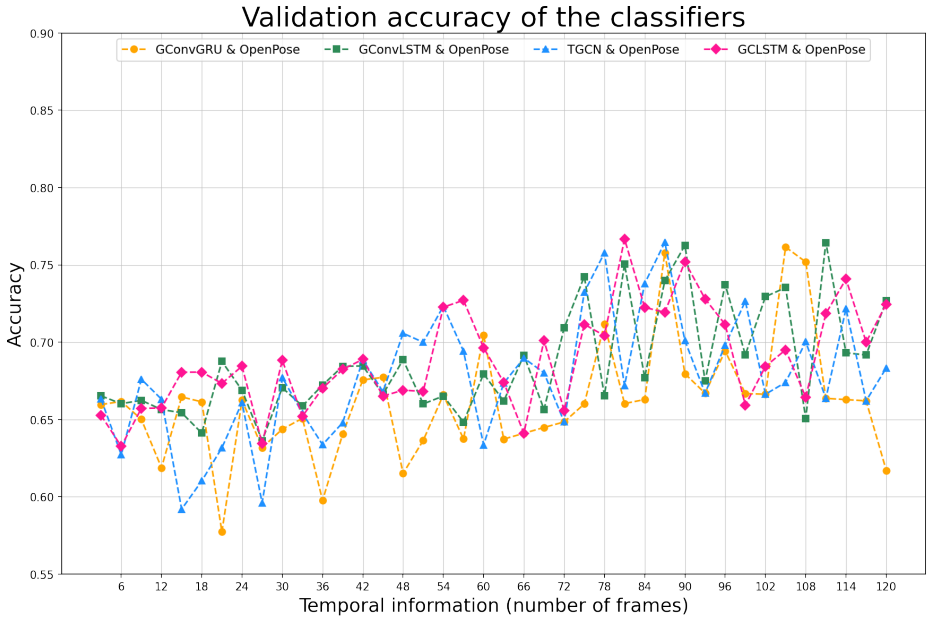



Fig. 15: Validation accuracy based on the number of frames used by each classifier in its input, using OpenPose skeletons as input.

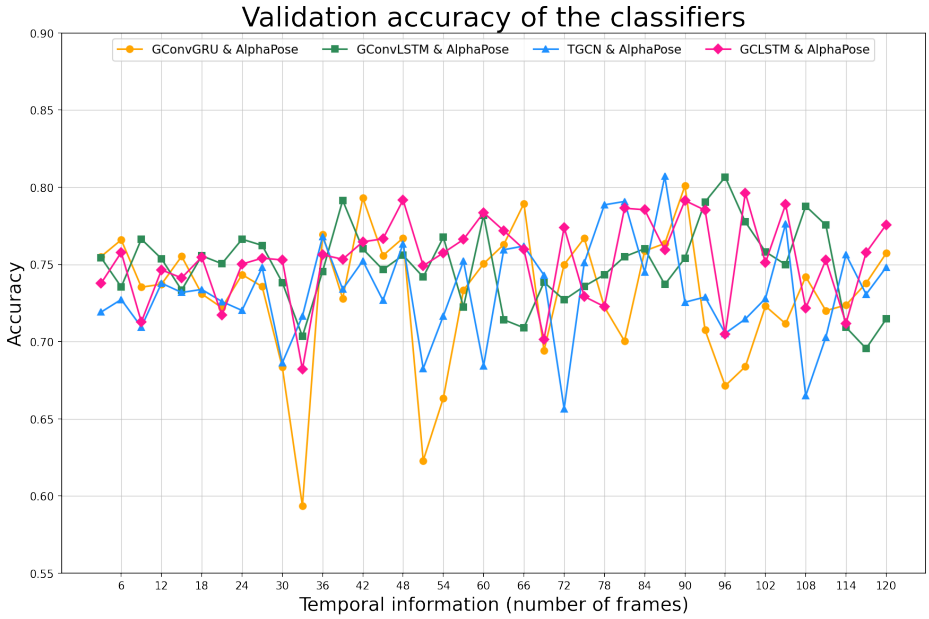



Fig. 16: Validation accuracy based on the number of frames used by each classifier in its input, using AlphaPose skeletons as input.

In Fig. 17 and Fig. 18 the best models for each temporal information value are shown when using OpenPose and AlphaPose skeletons, respectively, but this time showing the validation f1-score. In the case of this metric, an upward trend can be more easily observed as the number of frames increases, up to 87 for both pose estimator skeletons, which begins to decrease again.

In fact, the model that uses 87 frames and the input AlphaPose skeletons is the optimal model that was indicated before.

Furthermore, the best f1-score obtained with the OpenPose skeletons is much lower than the best f1-score obtained with the AlphaPose-estimated skeletons as we have already seen. This is coupled with the fact that the best f1-score model using AlphaPose-estimated joints has also the highest accuracy, also higher than the accuracy of the best model using OpenPose, and there is no clear trend that the accuracy could increase more in the case of using more than 120 frames. This justifies the fact that it is not necessary to try frame values greater than 120. Furthermore, using higher values would increase the training time, which grows linearly with respect to the number of input frames. Finally, another justification for not testing values greater than 120 is the decreasing trend of the value of the f1-score when utilizing more than 87 frames/skeletons as input.

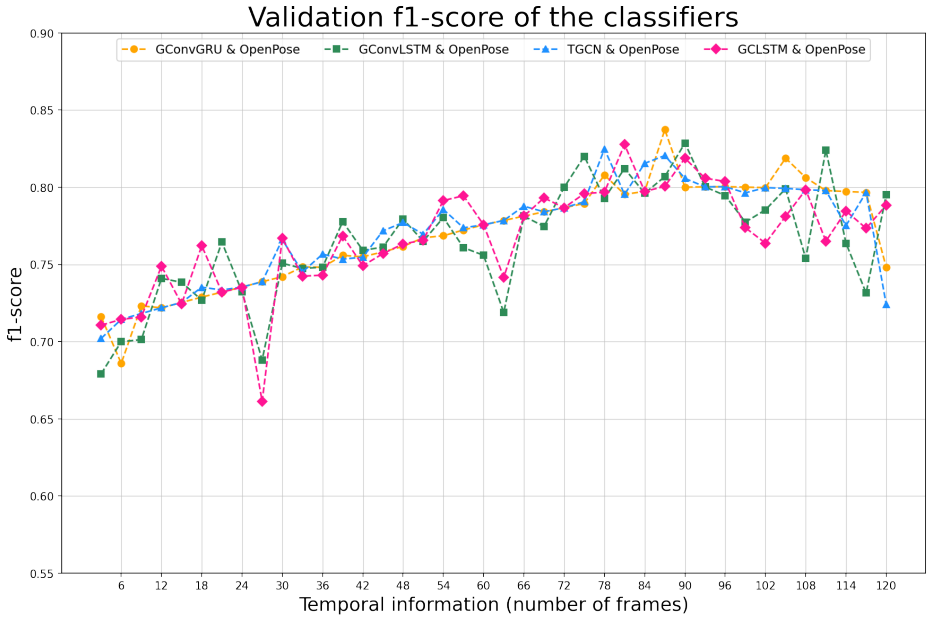



Fig. 17: Validation f1-score based on the number of frames used by each classifier in its input, using OpenPose skeletons as input.

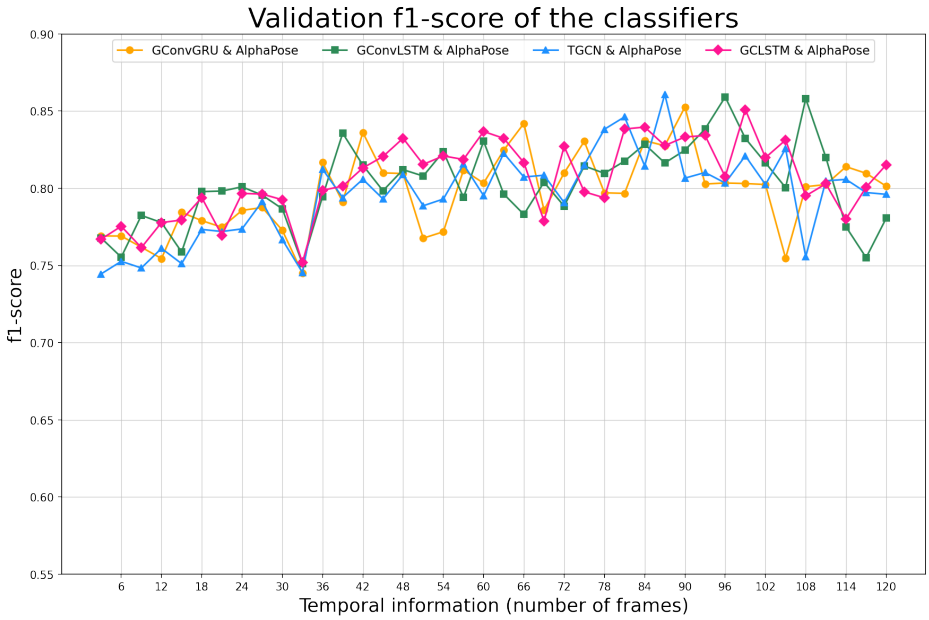



Fig. 18: Validation f1-score based on the number of frames used by each classifier in its input, using AlphaPose skeletons as input.

### C. Explainability

In this subsection are performed the experiments related to the explainability analysis aimed at understanding how the classifier, described in section III-B, generates the predictions of the crossing intention. The insights gained from this explainability analysis are then utilized to design an explainability-driven data augmentation technique that generates new skeletons by modifying the training subset. The objective is to generate more meaningful and varied skeletons for the training subset, and then use these augmented skeletons to retrain the classifier, with the aim of improving the validation results.

As explained in III-B, Graph Neural Networks (GNNs) are a type of neural network that are well-suited to processing structured data, such as graphs, which are used in this work to represent the skeleton joints of pedestrians. One way to use explainability methods in the context of GNNs for pedestrian crossing intention prediction is to analyze the node representations learned by the network and identify the most important nodes (i.e., joints) for predicting the pedestrian's intention.

Once the most important nodes have been identified, this information can be used to guide the creation of a more effective training set through data augmentation. For instance, new skeletons could be generated by manipulating the most important joints in the original skeletons, while keeping the structure and edge connectivity of the graph intact. This can help to create more diverse and representative training samples, and improve the performance of the classifier.

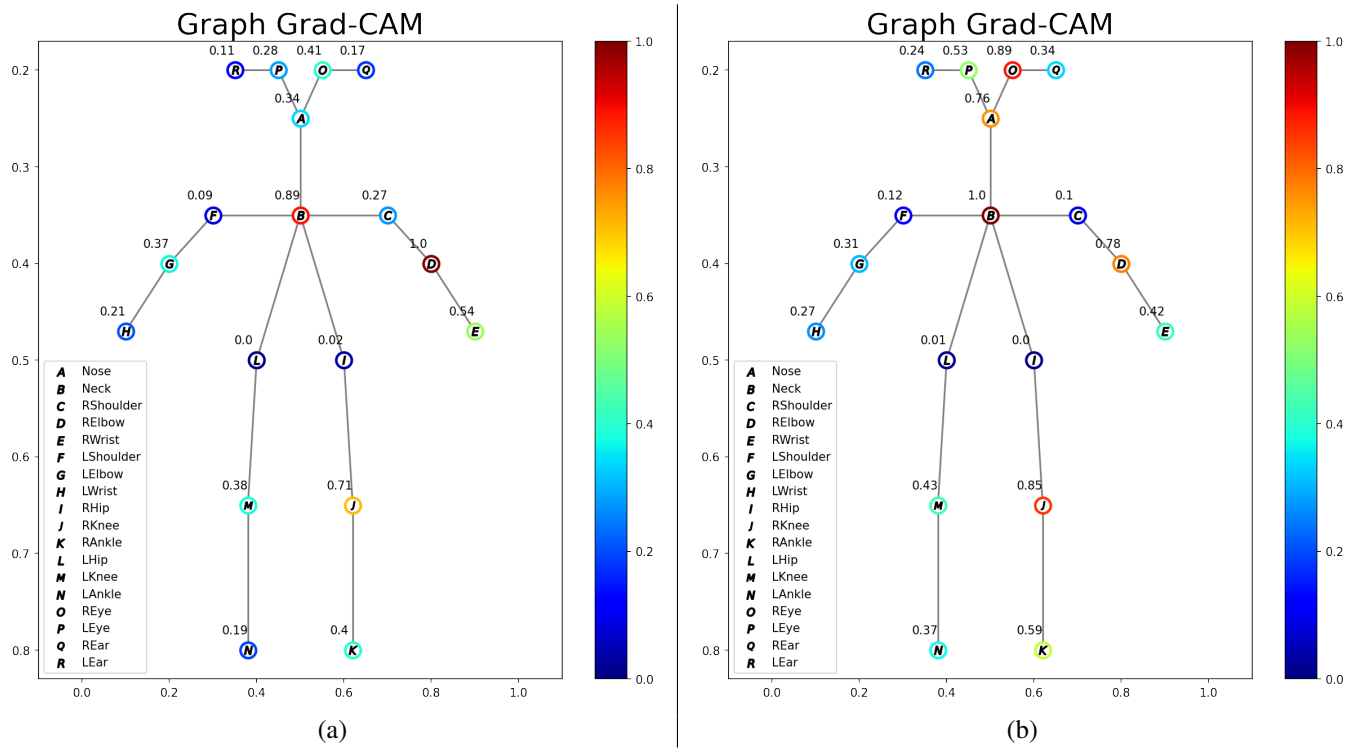
Adding noise to specific joints in the input data and observing the resulting change in validation accuracy is a first initial approach that could potentially provide some insights into which joints most influence the model's prediction of pedestrian crossing intention. However, adding noise to the input skeletons may not necessarily provide a clear indication of which joints are most important for the prediction, since the classifier may be able to compensate for the added noise by relying on other joints, being it difficult to interpret the results. Moreover, if the added noise is too strong or unrealistic, the analysis could provide incorrect information. Therefore, it may be more appropriate to use explainability techniques such as Grad-CAM (Gradient-weighted Class Activation Mapping) [36]. Grad-CAM is a technique that generates a heatmap that highlights the regions of an input image that are important for the model to produce a prediction. In the case of our proposed architecture for crossing intention prediction in which the classifier input are not images, but pedestrian skeletons instead, it is necessary to design a modification of Grad-CAM to generate a heatmap that highlights the critical joints in the input skeletons.

One of the advantages of Grad-CAM in contrast to other explainability techniques is that it does not require retraining the network. Instead, Grad-CAM computes the gradient of the output of the network with respect to the feature maps of a convolutional layer. These gradients are global-average-pooled over the width and height dimensions, to obtain neuron importance weights $\alpha_k^c$ of feature map $k$ for a target class $c$. Subsequently, a weighted combination of forward activation maps using those weights is performed, followed by a ReLU function. ReLU is applied to discard features that do not have a positive influence on the class of interest. This results in a heatmap of the same size as the feature maps.

In this section, it has been implemented a new version of Grad-CAM that utilizes the activations of Recurrent Graph Convolutional Layers (RGCL) instead of traditional convolutional layers, to achieve compatibility with the architecture proposed in section III-B. Specifically, the RGCL gradients of the last time step are used, and then global-average-pooled over the amount of nodes, to obtain the importance weights for each feature. Then, using those weights, a weighted combination of the RGCL forward activation maps of the last time step is performed, also followed by a ReLU function. This results in a heatmap of the same size as the number of output nodes of the RGCL, that in the proposed architecture is the same as the number of input nodes.

Fig. 19 shows the result of the Grad-Cam execution over all skeletons estimated by AlphaPose in JAAD dataset, for the not-crossing class (a) and crossing class (b). In both cases, the most important joints for the network to predict the class are the Neck, the Right Elbow and the Right Knee. Initially, it may seem counter-intuitive that the neck is one of the most important joints for the classification, but it is the joint that is located most in the center of the body on the horizontal axis, and therefore its movement indicates the trajectory of the pedestrian on said axis. This can provide the network with valuable information about where the person is going, which can be crucial to the decision. For instance, if the neck barely moves on the horizontal axis, the pedestrian is moving towards or away from the car, but in the same direction as the car. However, if the neck is moving horizontally, even with a certain angle, the pedestrian will be walking perpendicular (or almost) to the direction of the car, that is, the pedestrian will be crossing.

Nevertheless, to carry out the prediction, the Right Elbow and Right Knee joints are also very important, since they indicate the position and orientation of the arms and legs, respectively, very useful information to define the general pose of the pedestrian and determine in which direction intends to walk, since the arms and legs point in the direction in which people will walk. However, those same joints for the left side of the body, Left Elbow and Left Knee, despite being considerably more important than other joints, are much less relevant than their namesakes on the right side of the body. This can be caused by various reasons. For example, the model may be able to infer crossing intention primarily with the right arm and leg, since




(a) (b)

Fig. 19: Heatmap resulting of Grad-Cam execution **over all skeletons** in the dataset. Higher values mean more important joints: (a) Not-crossing class; (b) Crossing class

the other leg typically behaves similarly. It is noteworthy that the reason is not that there are more pedestrians crossing to one side than the other, since in Fig. 20 we can see the values of the Grad-Cam heatmap for two specific skeletons, both of crossing pedestrians, one to the left and the other to the right side, and in both the Right Elbow and Right Knee have a much greater importance than those joints on the left side. However, it is important to remember that explainability techniques such as Grad-CAM provide information about the regions of the input, in this case nodes, that contribute the most to the prediction of the model, but this cannot always be interpreted or understood by people, since the final decision of the classification is a complex mathematical process that is determined by many factors, and in which each articulation contributes to reach the final output of the network.

In order to gain a deeper understanding of the importance of the different body joints and how the model predicts the crossing intention, we could consider analyzing the dataset, to check if there is any kind of deviation to the right side in the pedestrian posture caused by the people appearing in the videos, or even if AlphaPose generates skeletons that visually were not skewed to the right but analytically they were, but this would be outside the scope of the project to be able to perform data augmentation based on this explainability analysis, and thus is left as future work.

### *D. Data augmentation*

The proposed data augmentation technique operates on the estimated skeletons of pedestrians and aims to augment the training data to improve the performance of the proposed classifier of section III-B that predicts the pedestrian crossing intention.

The top view representation of a pedestrian is not utilized in this project, but it is useful for the theoretical explanation of the proposed data augmentation technique. Therefore, if we observe the top view representation of a pedestrian crossing as shown in Fig. 21 (a), we can identify the positions of both shoulders of the pedestrian, next to an arrow with the direction in which that person is moving. If another pedestrian were in the same location but slightly rotated to the right, it would still be considered as performing the action of crossing or having the intention to cross, since it is a small rotation that hardly changes the trajectory.

However, although each estimated joint contains three values, the third component of each node is not the depth, but the confidence score of how confident the pose estimator is about that joint, as mentioned in section V. Therefore, 3D rotation is simulated by increasing or decreasing the distance on the horizontal axis with respect to the center of the body on that axis, which is the neck, as illustrated in Fig. 21 (b). It could be considered an approximation to simulate a rotation in the

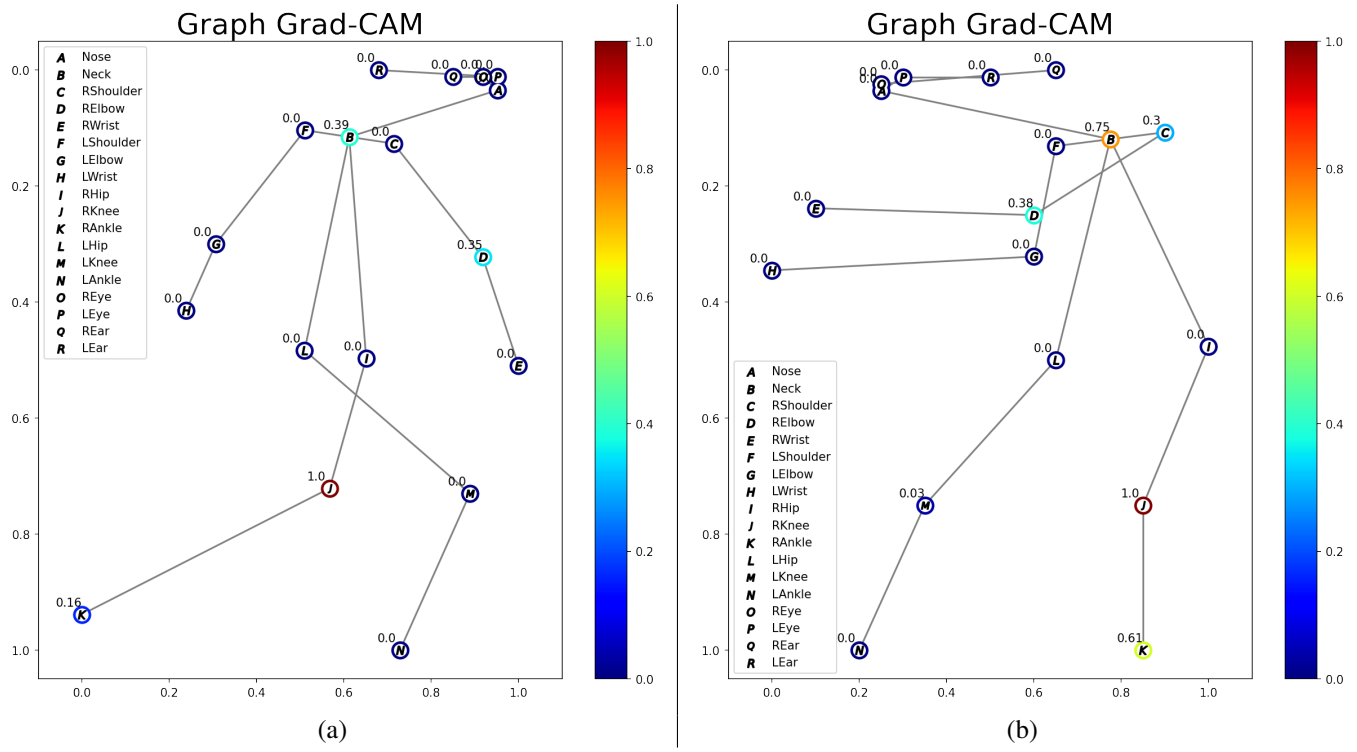


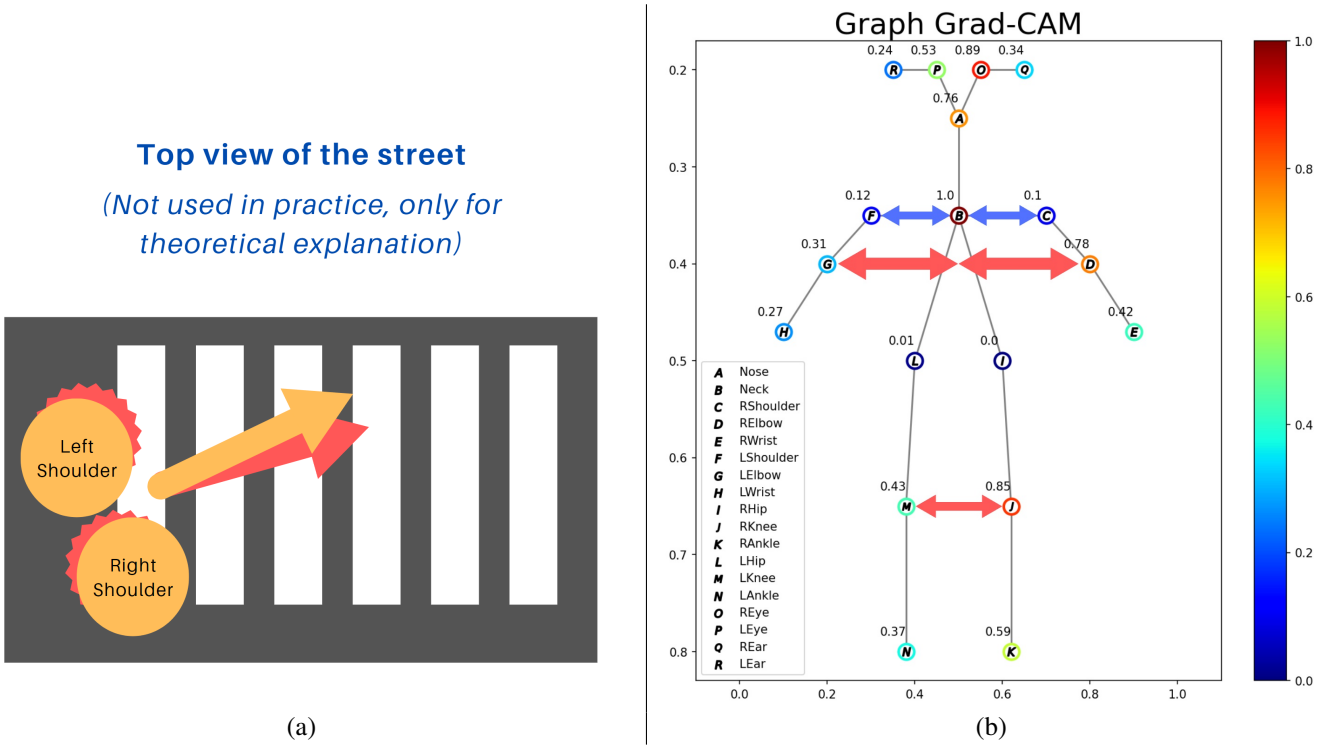

Fig. 20: Heatmap resulting of Grad-Cam execution **for two specific skeletons** of the dataset. Higher values mean more important joints: (a) Crossing from left to right (car perspective); (b) Crossing from right to left (car perspective)

3D vertical axis when depth information is not available, since, from the perspective of the car, when the pedestrian increases its rotation to be more perpendicular to the road direction, the pairs of joints (e.g. Left and Right Shoulders, Left and Right Elbows, etc.) appear in the estimated 2D projection to be closer to the horizontal center of the body, that is the Neck (and the Nose). While it may not accurately represent the true rotation in the three-dimensional space, the horizontal displacement can provide an enough approximation to generate augmented samples that include slight variations in the rotation of pedestrians.

Table XIV shows a comparison of the proposed explainability-driven data augmentation technique performing the previously explained horizontal displacement for the critical joints that came out of the explainability analysis, which are the Right Elbow and the Right Knee. This displacement is performed in pairs of the same joints, that is, a random displacement is carried out both to the Left Elbow and the Right Elbow to move them closer to or further from the Neck on the horizontal axis. The neck is also a critical joint, but in this case, since the proposed technique consists of simulating a rotation of the body with a horizontal displacement, it would not make sense to carry out the same procedure with the Neck, since when rotating the pedestrian, in the projection in two dimensions this joint would remain in the same position. However, the Neck is being used indirectly in this technique, since the other critical joints, and their equivalent on the opposite side of the body, are randomly moving closer or further away from the Neck on the horizontal axis in 2D.

In Table XIV it is also shown the result of applying this technique but for the two shoulders, which are not critical joints, but to compare the result with those that were determined to be critical in the previous subsection. A simpler data augmentation based on random multiplicative noise is also included as a comparative. In each experiment that uses data augmentation, the number of samples is twice as many compared to not using data augmentation, since the new augmented train subsets contain the modified skeletons while still preserving the original samples in the dataset. We can observe in the table that without using data augmentation, the best model obtained in subsection VI-B achieved a validation accuracy of 0.8071 and a validation f1-score of 0.8609. Using random multiplicative noise, both in all joints and only in the Neck, the accuracy only reaches 0.71. However, using our proposal based on horizontal displacements, if we use the shoulders, even despite the fact that they are not critical joints, the validation accuracy and f1-score are higher to using random multiplicative noise. Nonetheless, they are still inferior to not using any data augmentation at all.

However, if we carry out the proposed technique but with the critical joints determined in the explainability analysis of subsection VI-C, both metrics are superior compared to using the shoulders. Notably, using the Right Knee and its counterpart




(a) (b)

Fig. 21: Pedestrian joints: (a) Top view; (b) Front view

the Left Knee, the model achieves a higher validation accuracy compared to not using data augmentation, going from 0.8071 to 0.8226. The validation f1-score rises slightly from 0.8609 to 0.8659. Therefore, the explainability analysis has provided useful information to be combined with the data augmentation method, which increases the model's capability to generalize and improves the validation metrics. This is achieved by generating more varied skeletons in the train dataset.

| Data augmentation | Perturbed joints | Perturbation range | Classifier training and validation metrics | | | |
|---|---|---|---|---|---|---|
| | | | Train subset | | Validation subset | |
| | | | Accuracy | f1-score | Accuracy | f1-score |
| No augmentation | - | - | 0.7588 | 0.8403 | 0.8071 | 0.8609 |
| Random multiplicative noise | All joints | $[0.95, 1.05]$ | 0.7750 | 0.8453 | 0.6326 | 0.7077 |
| Random multiplicative noise | Neck | $[0.95, 1.05]$ | 0.6927 | 0.8184 | 0.6675 | 0.8006 |
| Random multiplicative noise | All joints | $[0.98, 1.02]$ | 0.7380 | 0.8366 | 0.7101 | 0.8163 |
| Random multiplicative noise | Neck | $[0.98, 1.02]$ | 0.7910 | 0.8606 | 0.7112 | 0.8031 |
| Horizontal displacement | LShoulder, RShoulder | $[0.98, 1.02]$ | 0.7292 | 0.8318 | 0.7032 | 0.8119 |
| Horizontal displacement | LElbow, RElbow | $[0.98, 1.02]$ | 0.7911 | 0.8550 | **0.8226** | **0.8659** |
| Horizontal displacement | LKnee, RKnee | $[0.98, 1.02]$ | **0.7973** | **0.8635** | 0.7223 | 0.8073 |
| Horizontal displacement | LElbow, RElbow, LKnee, RKnee | $[0.98, 1.02]$ | 0.7812 | 0.8505 | 0.7769 | 0.8343 |

TABLE XIV: Metrics of the classifier using different types of data augmentation.

## VII. Final results

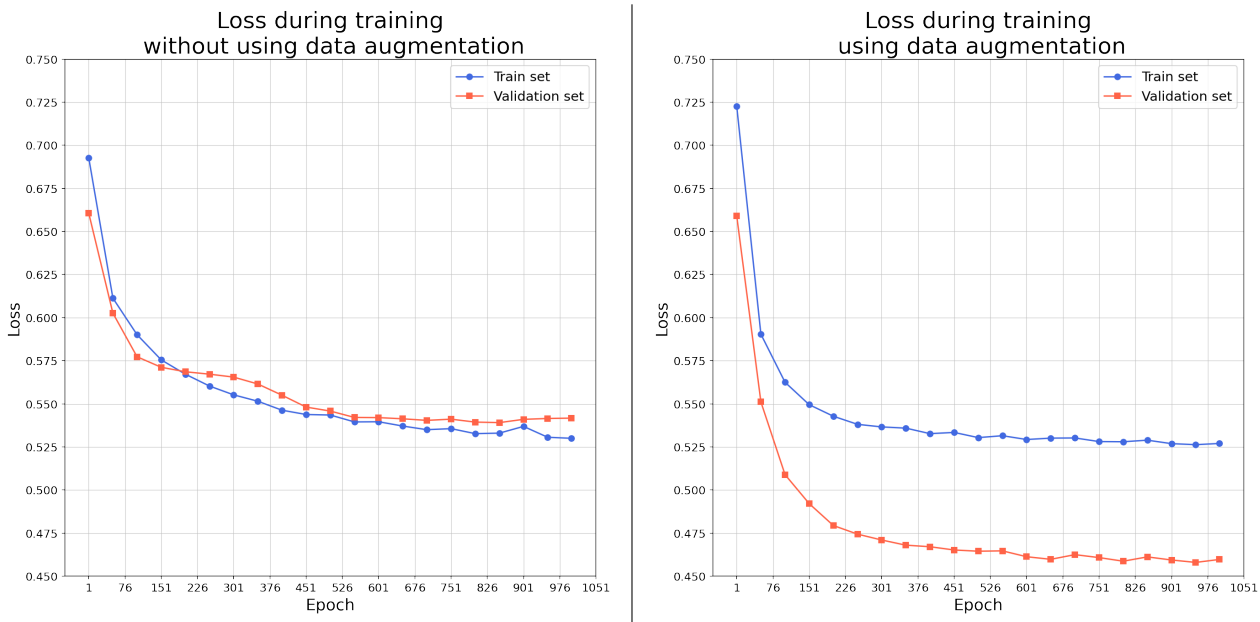
This section shows the final results of the binary classification of the intention to cross or not the street, for the best model obtained in the Grid Search of section VI-B. The optimal parameters are shown again in Table XV. That model has been retrained but using 1000 epochs instead of 100 as in the Grid Search, since for said search using 1000 epochs with the 960 hyperparameter combinations would have been unfeasible.

| Pose estimator (classifier input skeletons) | Optimal classifier hyperparameters | | |
|---|---|---|---|
| | Graph Layer | Temporal Info | Dropout |
| AlphaPose | TGCN | 87 | 0.5 |

TABLE XV: Hyperparameters that have given the best validation metrics.

Fig. 22 shows the loss plots during the training of the optimal classifier, without using data augmentation (Fig. 22 (a)) and using the best explainability-driven data augmentation (Fig. 22 (b)) proposed in section VI-D, which as we saw consisted of a horizontal displacement of the Left Elbow and Right Elbow joints to randomly get closer or move away to the Neck. It can be observed that when employing data augmentation, in spite of the fact that the loss stabilizes earlier compared to the case without its usage, the introduction of data augmentation results in a lower loss value.

Without using data augmentation, the training time of 1000 epochs was 92.5 minutes, and using data augmentation, 180.3 minutes. The training time is doubled in the case of using data augmentation because there are twice as many elements in the dataset, since in subsection VI-D each skeleton in the train subset was duplicated and then modified by a random horizontal displacement of the Left Elbow and Right Elbow joints. However, using the explainability-driven data augmentation proposed, by reaching 100 epochs it already exceeds the train and validation metrics compared to not using data augmentation, having a model that takes only 18 minutes to train. It has been tried to use more than 1000 epochs in the case of not using data augmentation, since its training time is lower, but after 700 epochs it converges and the loss does not decrease further.




(a) (b)

Fig. 22: Loss plots during training for 1000 epochs: (a) Without using data augmentation; (b) Using LElbow and RElbow horizontal displacement data augmentation

The next two subsections present the analysis of the optimal model trained for 1000 epochs. In subsection VII-A a quantitative analysis is performed comparing the metrics with the state of the art, and in subsection VII-B we can find a qualitative analysis showing pedestrian images in the JAAD dataset together with the crossing/not-crossing prediction made by the network.

### A. Quantitative results

Table XVI shows the metrics of the binary classification of the pedestrian crossing intention in the test subset for the state-of-the-art methods, as well as our two proposals. With our proposed classifier model we considerably outperform the state of the art in the f1-score, precision and recall metrics, both without using the explainability-driven data augmentation proposal, and using it. Notably, we greatly surpass the f1-score of the state-of-the-art BiPed method, which obtains 0.60, while we obtain 0.83 without using data augmentation, and 0.84 using it. Our method also supposes a very notable improvement in precision and recall, which rise from 0.65 and 0.71 respectively, in the state of the art, to 0.79 and 0.90 using our method that uses data augmentation, and 0.78 and 0.89 without using it.

Finally, it is not a significant drawback to fall behind the other state-of-the-art methods in terms of accuracy and ROC AUC, since in the case of ROC AUC, our method achieves a value of 0.78 and the state of the art 0.79. Regarding accuracy, these methods achieve high values but at the expense of very lower values in f1-score, precision, and recall. Since the number of samples of each class in all subsets is very unbalanced, obtaining a high accuracy with such low values in the other metrics implies that the classification is not as successful as our approach. This is due to the fact that, although our proposal may have a lower accuracy, it presents higher values in f1-score, precision, and recall, indicating a superior classification performance.

| Method | Accuracy | ROC AUC | F1-score | Precision | Recall |
|---|---|---|---|---|---|
| ATGC | 0.64 | 0.60 | 0.53 | 0.50 | 0.56 |
| MM-LSTM | 0.80 | 0.60 | 0.40 | 0.39 | 0.41 |
| I3D | 0.82 | 0.75 | 0.55 | 0.49 | 0.63 |
| SF-GRU | 0.83 | 0.77 | 0.58 | 0.51 | 0.67 |
| PCPA | 0.83 | 0.77 | 0.57 | 0.50 | 0.66 |
| BiPed | 0.83 | **0.79** | 0.60 | 0.52 | 0.71 |
| PedFormer | **0.93** | 0.76 | 0.54 | 0.65 | 0.46 |
| *PedRGCN without D. Augm. (ours)* | 0.76 | 0.77 | 0.83 | 0.78 | 0.89 |
| *PedRGCN with explainability-driven D. Augm. (ours)* | 0.76 | 0.78 | **0.84** | **0.79** | **0.90** |

TABLE XVI: Comparison to the state-of-the-art methods for the test subset classification metrics of the pedestrian crossing intention prediction. Values of the other methods obtained from the corresponding papers. D. Augm.: Data augmentation.

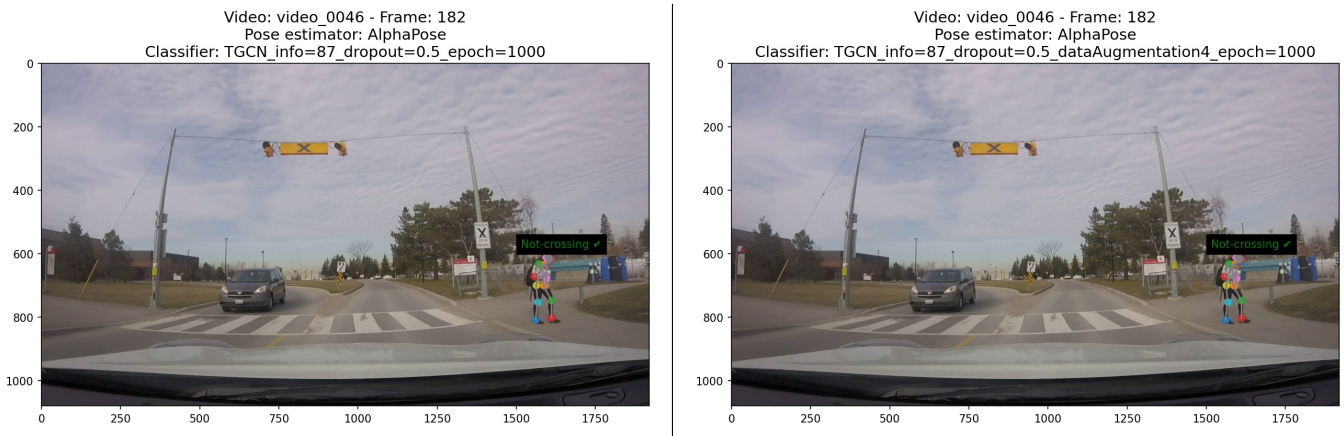
To ensure that the improvement produced by using data augmentation is not caused by randomness, six trainings and tests have been performed using the proposed classifier both with and without data augmentation. Since the data augmentation generation has some randomness in the amount of horizontal displacement that is applied for each pedestrian, for each of the six training processes that use data augmentation, a different augmented train subset has been generated, to ensure the validity of the experiment. We have computed the classification metrics for each execution and utilized them in a Wilcoxon signed-rank test to determine the statistical significance of using data augmentation. This test is suitable when the number of samples is small and the data does not follow a normal distribution. The null hypothesis assumes that there are no significant differences between the two approaches. The resulting p-value has been 0.03125 for all the metrics, which is lower than the significance level of 0.05. Consequently, we can reject the null hypothesis that there are no significant differences between the models. Therefore, it can be concluded that the proposed explainability-driven data augmentation indeed improves the metrics.

### B. Qualitative results

Qualitatively, the network prediction results are very accurate both without using the proposed data augmentation and using it. The two approaches are correct in most cases both when crossing (Fig. 23) and when not-crossing (Fig. 24). Furthermore, in Fig. 23 the prediction is correct, crossing, despite the fact that they are not performing said action yet. This indicates that, indeed, the network manages to anticipate the intention of both pedestrians to cross. On the other hand, in Fig. 25, Fig. 26 and Fig. 27 we can observe some cases in which the use of data augmentation during training has allowed, in these test videos, a greater generalization, by correctly predicting cases where the network trained without data augmentation is wrong. Notably, in Fig. 25, the data augmentation allows to correctly classify the intention of a little girl, who has a smaller skeleton. Finally, in Fig. 28 we can observe a case in which both approaches fail according to the ground truth, which is not-crossing. However, the pedestrian is actually in the process of crossing, which is adequately indicated by the network. We have identified several other examples of questionable or inaccurate annotations that can introduce conflicting information to the network, thereby hindering its training. Nevertheless, due to the widespread usage of this dataset for this task, we proceeded with its utilization.


(a) (b)

Fig. 23: Classifier prediction for frame 168 of video 55: (a) Trained without data augmentation; (b) Trained with data augmentation

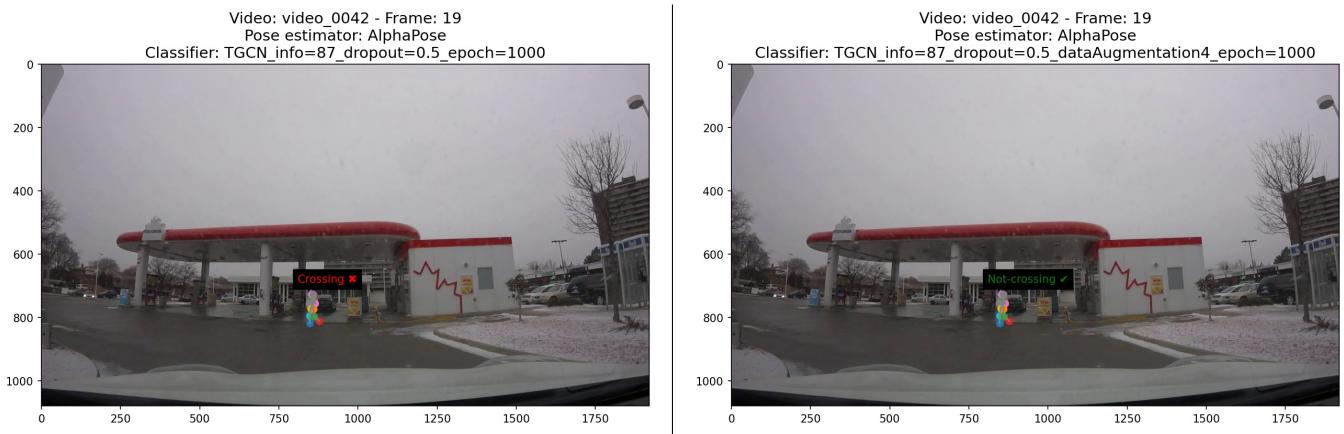



(a) (b)

Fig. 24: Classifier prediction for frame 182 of video 46: (a) Trained without data augmentation; (b) Trained with data augmentation

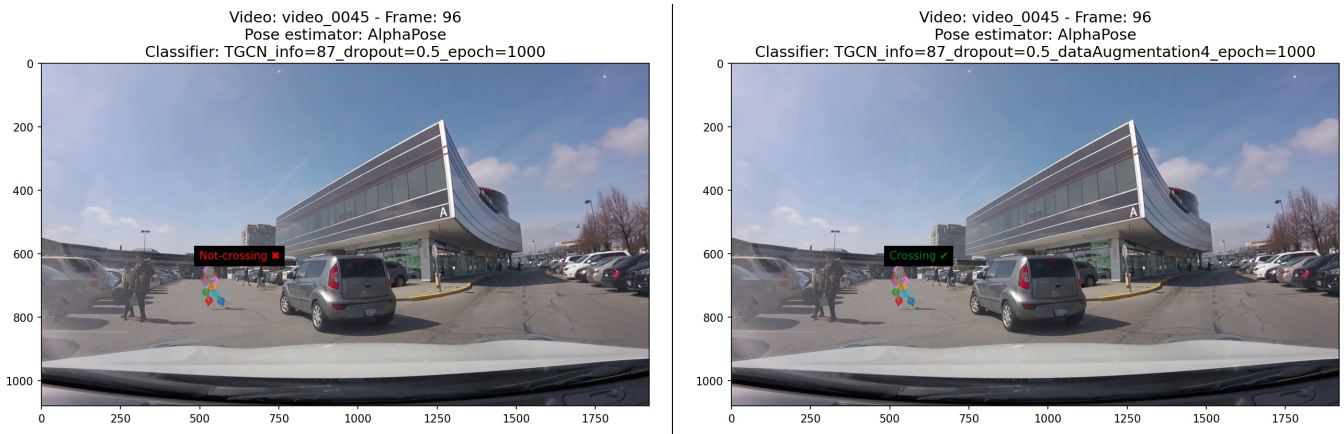

(a) (b)

Fig. 25: Classifier prediction for frame 145 of video 5: (a) Trained without data augmentation; (b) Trained with data augmentation




(a) (b)

Fig. 26: Classifier prediction for frame 19 of video 42: (a) Trained without data augmentation; (b) Trained with data augmentation




(a) (b)

Fig. 27: Classifier prediction for frame 96 of video 45: (a) Trained without data augmentation; (b) Trained with data augmentation


(a) (b)

Fig. 28: Classifier prediction for frame 130 of video 28: (a) Trained without data augmentation; (b) Trained with data augmentation

# VIII. Conclusions

This work presents three main contributions to the field of autonomous driving, along with several comprehensive analyses of the different methods employed. Firstly, we propose an architecture for predicting the crossing intention of pedestrians, defined as a binary classification between the crossing and not-crossing labels. The proposed architecture significantly outperforms the state-of-the-art methods in terms of f1-score, precision, and recall metrics. In addition, through qualitative analysis, we demonstrate the network's ability to predict pedestrian intention with a similar performance to that of a human driver.

The proposed method consists of a first step in which two pose estimators, OpenPose and AlphaPose, are employed to estimate the skeletons of pedestrians appearing in the JAAD dataset videos. These skeletons are then saved on disk for subsequent utilization as input for a classifier that performs the crossing/not-crossing prediction.

On the one hand, regarding the first step of the estimation of the pose, OpenPose did not detect 6.43% of the pedestrians that contain crossing/not-crossing annotations among all the frames of all the videos in the JAAD dataset, while AlphaPose did not detect only 1.25% of them. Furthermore, AlphaPose has been capable of estimating all joints without missing any in 99.76% of the detected pedestrians, while OpenPose only succeeds in 25.47% of the skeletons it detects. In the rest, there is always at least one undetected articulation. In fact, using OpenPose, in 15.18% of the skeletons, there are more than five missing joints. Consequently, the classifier's validation metrics using the skeletons estimated by AlphaPose as input were found to be higher compared to those achieved using OpenPose.

On the other hand, in the second step of the proposed method, in the classifier, a Recurrent Graph Convolutional Layer (RGCL) has been used to capture both the spatial dependencies of the joints of the pedestrian skeletons and the temporal dependencies along the different frames. To determine which RGCL gives better metrics, a Grid Search was performed comparing the GConvGRU, GConvLSTM, TGCN, and GCLSTM layers and the amount of input frames to the network. The optimal architecture utilizes the TGCN layer and 87 input skeletons for each pedestrian, corresponding to the preceding 87 frames.

An explainability analysis has also been performed using our own implementation of Grad-Cam on the RGCL, which has determined that the most critical joints of the human body when classifying the pedestrian crossing intention are the Neck, Right Knee, and Right Elbow.

Finally, based on the results of the explainability analysis, we have proposed a data augmentation method that generates new skeletons for the training subset by modifying the existing skeletons through horizontal displacements of the critical joints. Notably, it has been determined that applying random horizontal displacements to both the Right Knee and Left Knee, causing them to randomly move closer or farther away from the Neck, enhance the validation metrics and, subsequently, the test metrics. This improvement is observed when compared to not using data augmentation and when data augmentation methods not based on the information from the explainability analysis are used. As a result, the proposed explainability-driven data augmentation approach also surpasses the state-of-the-art performance in terms of f1-score, precision, and recall metrics.

In conclusion, the proposed advancements have led to significant improvements in the test metrics for predicting the crossing intention of pedestrians, particularly in the f1-score metric, which increases from 0.60 in the state of the art, to 0.84 with our method. This improvement can potentially enhance the safety of both drivers and pedestrians. In the future, the results could be further improved by proposing new data augmentation methods based on the performed explainability analysis. Additionally, the proposed pose-based model could be combined with visual models that provide a context of the scene from the car's point of view, as well as incorporating information of the objects visible to the driver, such as other pedestrians, crosswalks, or traffic lights. Furthermore, the implemented explainability and data augmentation techniques can be applied to other deep learning graph-based tasks, further expanding their utility.

# REFERENCES


[1] K. Saleh, M. Hossny, and S. Nahavandi, "Real-time intent prediction of pedestrians for autonomous ground vehicles via spatio-temporal densenet," 2019.
[2] Z. Fang and A. M. López, "Is the pedestrian going to cross? answering by 2d pose estimation," 2018.
[3] S. Zhang, M. Abdel-Aty, Y. Wu, and O. Zheng, "Pedestrian crossing intention prediction at red-light using pose estimation," April 2021. [Online]. Available: https://shilezhang.github.io/files/paper4.pdf
[4] S. A. Bouhsain, S. Saadatnejad, and A. Alahi, "Pedestrian intention prediction: A multi-task perspective," 2021.
[5] A. Rasouli, M. Rohani, and J. Luo, "Bifold and semantic reasoning for pedestrian behavior prediction," 2021. [Online]. Available: https://arxiv.org/abs/2012.03298
[6] H. Zhang, Y. Liu, C. Wang, R. Fu, Q. Sun, and Z. Li, "Research on a pedestrian crossing intention recognition model based on natura observation data," March 2020. [Online]. Available: https://www.researchgate.net/publication/340144653_Research_on_a_Pedestrian_Crossing_Intention_Recognition_Model_Based_on_Natural_Observation_Data
[7] D. Yang, H. Zhang, E. Yurtsever, K. Redmill, and Ümit Özgüner, "Predicting pedestrian crossing intention with feature fusion and spatio-temporal attention," 2021. [Online]. Available: https://arxiv.org/abs/2104.05485
[8] J. Gesnouin, S. Pechberti, B. Stanciulescu, and F. Moutarde, "Trouspi-net: Spatio-temporal attention on parallel atrous convolutions and u-grus for skeletal pedestrian crossing prediction," 2021. [Online]. Available: https://arxiv.org/abs/2109.00953
[9] I. Kotseruba, A. Rasouli, and J. K. Tsotsos, "Benchmark for Evaluating Pedestrian Action Prediction," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 1258–1268. [Online]. Available: https://openaccess.thecvf.com/content/WACV2021/papers/Kotseruba_Benchmark_for_Evaluating_Pedestrian_Action_Prediction_WACV_2021_paper.pdf
[10] A. Rasouli and I. Kotseruba, "Pedformer: Pedestrian behavior prediction via cross-modal attention modulation and gated multitask learning," 2022. [Online]. Available: https://arxiv.org/abs/2210.07886
[11] Y. Yao, E. Atkins, M. J. Roberson, R. Vasudevan, and X. Du, "Coupling intent and action for pedestrian crossing behavior prediction," 2021. [Online]. Available: https://arxiv.org/abs/2105.04133
[12] T. Chen, T. Jing, R. Tian, Y. Chen, J. Domeyer, H. Toyoda, R. Sherony, and Z. Ding, "Psi: A pedestrian behavior dataset for socially intelligent autonomous car," 2022. [Online]. Available: https://arxiv.org/abs/2112.02604
[13] J. Lorenzo, I. P. Alonso, R. Izquierdo, A. L. Ballardini, Álvaro Hernández Saz, D. F. Llorca, and M. Ángel Sotelo, "Capformer: Pedestrian crossing action prediction using transformer," 2021. [Online]. Available: https://doi.org/10.3390/s21175694
[14] S. Zamboni, Z. T. Kefato, S. Girdzijauskas, N. Christoffer, and L. D. Col, "Pedestrian trajectory prediction with convolutional neural networks," 2021. [Online]. Available: https://arxiv.org/abs/2010.05796
[15] A. Rasouli, I. Kotseruba, and J. K. Tsotsos, "Are they going to cross? a benchmark dataset and baseline for pedestrian crosswalk behavior," October 2017. [Online]. Available: https://openaccess.thecvf.com/content_ICCV_2017_workshops/papers/w3/Rasouli_Are_They_Going_ICCV_2017_paper.pdf
[16] E. Insafutdinov, L. Pishchulin, B. Andres, M. Andriluka, and B. Schiele, "Deepercut: A deeper, stronger, and faster multi-person pose estimation model," November 2016. [Online]. Available: https://arxiv.org/pdf/1605.03170.pdf
[17] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," 2017.
[18] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: Realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. [Online]. Available: https://arxiv.org/pdf/1812.08008v1.pdf
[19] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask r-cnn," March 2017. [Online]. Available: https://arxiv.org/pdf/1703.06870v1.pdf
[20] S. H. Tsang, "Review: Mask r-cnn (instance segmentation and human pose estimation)," April 2020. [Online]. Available: https://medium.com/analytics-vidhya/review-mask-r-cnn-instance-segmentation-human-pose-estimation-61080a93bf4
[21] Fang, Hao-Shu, Xie, Shuqin, Tai, Yu-Wing, Lu, and Cewu, "Rmpe: Regional multi-person pose estimation," 2017. [Online]. Available: https://arxiv.org/pdf/1612.00137.pdf
[22] H.-S. Fang, J. Li, H. Tang, C. Xu, H. Zhu, Y. Xiu, Y.-L. Li, and C. Lu, "Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. [Online]. Available: https://arxiv.org/pdf/2211.03375.pdf
[23] Machine Vision and Intelligence Group @ Shanghai Jiao Tong University (SJTU), "Alphapose: Multi-person pose estimation and tracking - github repository." [Online]. Available: https://github.com/MVIG-SJTU/AlphaPose
[24] J. Schapke, "An introduction to graph neural networks," February 2020. [Online]. Available: https://towardsdatascience.com/an-introduction-to-graph-neural-networks-e23dc7bdfba5
[25] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," 2017. [Online]. Available: https://arxiv.org/abs/1606.09375
[26] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *International Conference on Learning Representations (ICLR)*, 2017. [Online]. Available: https://arxiv.org/abs/1609.02907
[27] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," 2018. [Online]. Available: https://arxiv.org/abs/1801.07455
[28] B. Rozemberczki, P. Scherer, Y. He, G. Panagopoulos, A. Riedel, M. Astefanoaei, O. Kiss, F. Beres, G. Lopez, N. Collignon, and R. Sarkar, "PyTorch Geometric Temporal - Recurrent Graph Convolutional Layers." [Online]. Available: https://pytorch-geometric-temporal.readthedocs.io/en/latest/modules/root.html#recurrent-graph-convolutional-layers
[29] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," 2016. [Online]. Available: https://arxiv.org/pdf/1612.07659.pdf
[30] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, September 2020. [Online]. Available: https://arxiv.org/pdf/1811.05320.pdf
[31] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2018. [Online]. Available: https://arxiv.org/pdf/1707.01926.pdf
[32] J. Chen, X. Wang, and X. Xu, "Gc-lstm: Graph convolution embedded lstm for dynamic network link prediction," October 2021. [Online]. Available: https://arxiv.org/pdf/1812.04206.pdf
[33] The pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134
[34] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, and et al., "Microsoft coco: Common objects in context," February 2015. [Online]. Available: https://arxiv.org/pdf/1405.0312.pdf
[35] B. Rozemberczki, P. Scherer, Y. He, G. Panagopoulos, A. Riedel, M. Astefanoaei, O. Kiss, F. Beres, G. Lopez, N. Collignon, and R. Sarkar, "PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models," *Proceedings of the 30th ACM International Conference on Information and Knowledge Management*, p. 4564–4573, 2021.
[36] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 336–359, oct 2019. [Online]. Available: https://arxiv.org/abs/1610.02391