ATTENTION-BASED NETWORKS FOR HUMAN TRAJECTORY PREDICTION

AADIT PATEL [AADITP@SEAS], PRANAV SHAH [PRANAVPS@SEAS], SHARON RICHU SHAJI [SSHAJI@SEAS],

ABSTRACT. In this project, we implement the classical attention-based Transformer Network (TF) to predict the future trajectories of individuals in a scene. The project is inspired by the paper ”Transformer Networks for Trajectory Forecasting” [1]. We use the TrajNet dataset which is a superset of diverse datasets. Training on this dataset ensures that our model generalizes and performs well in unseen situations. We build our model using different optimizer and scheduler techniques and analyze the one that gives us the best performance. We then perform extensive testing using the best model and present some quantitative and qualitative results. The results show that our best TF model is able to predict future pedestrian trajectories with an average error of $\sim$ 45 cm.

1. INTRODUCTION

Human trajectory prediction is a key module for any autonomously navigating system and its applications cover a wide range from mobile robot navigation, including autonomous driving, smart video surveillance to object tracking. The ability to predict the future movement of humans using their past status can be used to avoid collisions and plan safer paths for an autonomous vehicle. Trajectory prediction is also useful in situations where the pedestrian is occluded by an object and is not visible to the camera for certain frames. In this project, we leverage the attention based learning capabilities of a transformer

Transformer networks are widely used for Natural Language Processing to model word sequences. In this project, we use transformers for the unique purpose of observing the movement of humans and forecasting their future steps.

1.1. Contributions:

Our contributions to the project are as follows:

- We implemented the Transformer architecture shown in Fig 1 for human trajectory forecasting.
- We examined different optimizers and learning rate schedulers and compared their outputs to finalize the ones that gave us the best performance. Once this was decided, we performed extensive testing and tuned the parameters of the model to arrive at the best model.
- Finally, we visualized the results and drew some important conclusions and possible improvements.

2. RELATED WORK AND BACKGROUND

The prediction of possible future paths is a central building block for an automated risk assessment. Trajectory forecasting is similar to sequence modeling and early work on sequence modeling has adopted Gaussian regression models [7], time series analysis [8] and auto regressive models [9]. Later, models like LSTM [5] and RNN models have performed provenly better on sequence/time-series models. These techniques either focus on analyzing the data sequentially and prediction for the future time steps or they use the interactions between different subjects and factor in the background information along with the sequential predictions. The later have proven to be better than the former primarily because they incorporate the background information and the various interactions between subjects. Transformers [2] have been proven to perform better than LSTM models and we believe that it will be more suitable to model the sequence of human trajectories and consequently forecast the trajectories. The most recent survey on methods used for trajectory prediction of humans [6] gives us a brief overview of the progress in this field and we intend to use Transformers to build upon it.

Transformer networks as mentioned are used for data that are inherently sequential. However, instead of sequentially processing the data like how RNN's and LSTM's work, it uses the attention mechanism which takes in all the sequentially observed data at once and enables the model to focus on what is important. A Transformer internally consists of a combination of Encoder and Decoder layers in which resides the attention layers.

3. Approach

In this work, we address the problem of future positions by processing their current and prior positions.

3.1. **Problem Formulation:**

Assume there are a total of N pedestrians involved in a scene, and t is the current time stamp (frame). The trajectory of a pedestrian $P_i(i \in [1, N])$ from timestamp t_s to t_e is denoted as $T_i^{t_s:t_e} = [(x_i^{t_s}, y_i^{t_s}), ..., (x_i^{t_e}, y_i^{t_e})]$. The spatial location at time t of a pedestrian P_i is denoted as $p_i^t = (x_i^t, y_i^t)$. If we let 0 be the current time stamp, the current and prior positions observed in cartesian coordinates are denoted as $\tau_{obs} = T_i^{t_s:0}$, and the predicted positions are $\tau_{pre} = T_i^{1:t_e}$. The proposed network aims to generate predicted trajectories $\hat{\tau}_{pre}$ that match the ground truth future trajectories τ_{gt}.

We sequentially predict the points in each future frame. It was observed that the displacement prediction is easier in sequential estimation [3] [4], therefore we predict the location displacement corresponding to current frame for each pedestrian. When feeding to the transformer, the input vector is embedded into a higher D-dimensional space by a linear projection with a matrix of weights $e_{obs}^{(i,t)} = D_{obs}W_i$, where D_{obs} is the input displacement vector. Similarly, the output of the transformer is back-projected to the cartesian coordinates.

3.2. **Positional Encoding:**

The transformer network discards the sequential nature of time-series data used in the LSTM-based model [5], but models temporal dependencies with self-attention mechanism. Thus, the input embedding data $E_{obs}^{i,t}$ consists of spatial trajectory embedding $e_{obs}^{(i,t)}$ and temporal position embedding P^t. By following the same setting as [1], the position embedding P^t is defined by sine and cosine functions.

$$\begin{aligned} E_{obs}^{(i,t)} &= e_{obs}^{(i,t)} + P^t \\ P^t &= p(t,d)_{d=1}^{D} \\ p(t,d) &= \begin{cases} \sin\left(\frac{t}{10000^{d/D}}\right) & \text{for } d \text{ even} \\ \cos\left(\frac{t}{10000^{d/D}}\right) & \text{for } d \text{ odd} \end{cases} \end{aligned}$$

3.3. **Encoder-Decoder Transformer:**

Figure 1 shows the overall framework of the proposed approach for pedestrian trajectory prediction. In the figure, N_x denotes the number of encoder and decoder layers. We've chosen N_x to be 6 and hence the encoder and decoder are made up of 6 layers each and each layers consists of 3 major building blocks, namely: **i.** an attention module, **ii.** a feed-forward fully connected module, and **iii.** residual connections.

The attention module is responsible for capturing the sequence non-linearities. In an attention module, an input query (Q) is compared against other sequence entries, called keys (K), by using a scaled dot product. This dot product is then scaled down by $\sqrt{d_k}$, where d_k is the embedding dimensionality. The output is then used to weight the same sequence entries, not called values (V). Attention is given by the equation:

$$\text{Attention}(Q,K,V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

Multi-head attention is the core component of Transformer. Unlike simple attention, the multi-head mechanism breaks the input into many small chunks, computes the scaled dot product of each subspace in parallel, and finally concatenates all the attention output.

$$\begin{aligned} \text{Multi-head Attention}(Q,K,V) &= concat(head_1, ..., head_h)W^O \\ head_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned}$$

where $W_i^Q, W_i^K, W_i^V, W_i^O$ are the weight matrices in queries, keys, values and output. All these weights can be trained.

The encoding stage is given an input E_{obs} and we get an out as two vectors of keys (K_{enc}) and values (V_{enc}). These vectors are then passed on to the decoder stage. The decoder iteratively predicts the future trajectory positions. At each new prediction step, a new decoder query Q_{dec} is compared against the encoder keys K_{enc} and values V_{enc} using the equation 1.

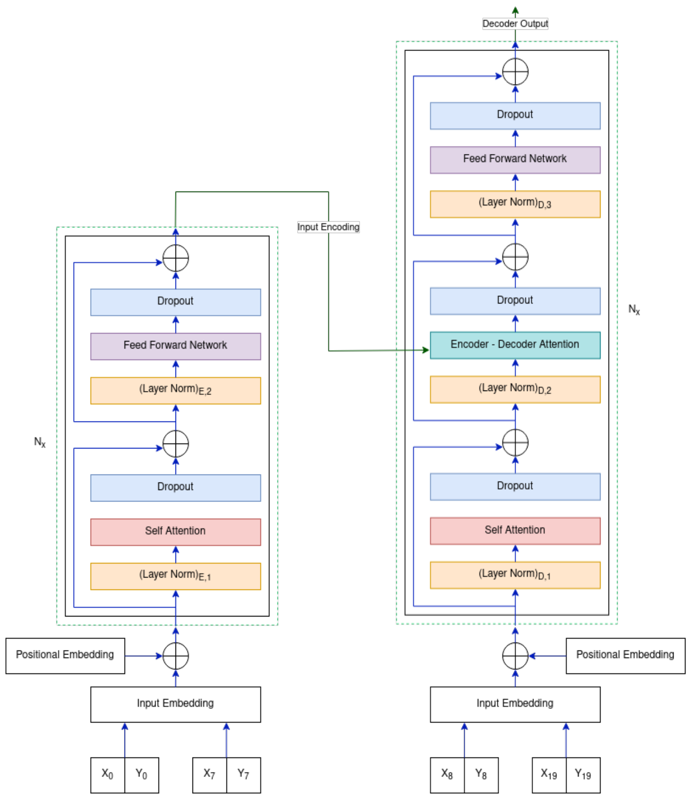


FIGURE 1. Transformer architecture

3.4. **Implementation details:**

Our TF implementation adopts from the original transformer network in [2], i.e., d_{model} = 512, 6 layers and 8 attention heads. The network is trained using an Adam optimizer with a cosine learning rate scheduler. We use an L2-loss between the predicted and target pedestrian trajectories. The dropout probability is 0.1.

4. EXPERIMENTAL RESULTS

Our Transformer model was tested using the TrajNet dataset. This dataset is a very large dataset and consists of four groups of trajectories: 1) BIWI Hotel[10], 2) MOT PETS [11], 3) Crowds UCY [12] and the 4) Standford Drone Dataset [13].

4.1. **Metrics used:**

The following two metrics were used to test the performance of our Transformer model:

(1) **Mean Average Displacement Error (MAD):** This metric is equivalent to the Average Displacement Error (ADE). It takes the mean of the L-2 distance error between the ground truth and the model predictions for each time step of the horizon.
(2) **Final Average Displacement (FAD):** This metric calculates the mean of the L-2 distance error between the ground truth and the model predictions at the final time step of the horizon.

4.2. **Quantitative Results:**

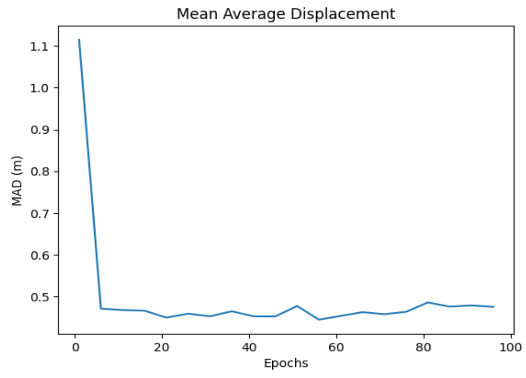
We tried out different learning rate schedulers and optimizers and chose the one that gave us the best performance on the TrajNet dataset. Table 1 shows a comparison between the results obtained using different combinations of schedulers and optimizers. The schedulers we tested were Cosine and Step schedulers and the optimizers were Adam optimizer and Stochastic Gradient Descent (SGD) with momentum=0.9 and Nesterov's acceleration.

Transformer Models			
Optimizer	Learning Rate Scheduler	MAD (m)	FAD (m)
SGD+Nestrov	-	1.2737	2.1309
Adam	-	0.47342	1.0574
SGD+Nestrov	Step	0.8604	1.6073
Adam	Step	0.4757	1.0626
SGD+Nestrov	Cosine	1.2665	2.1955
Adam	Cosine	0.4592	1.0108

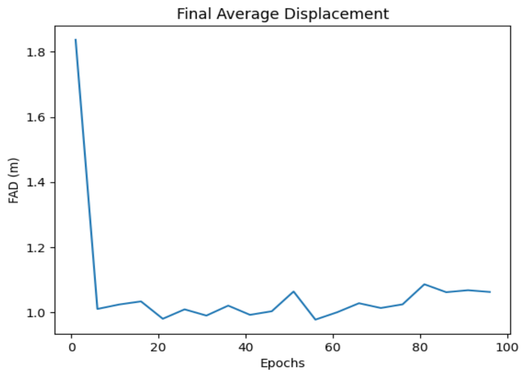
TABLE 1. Transformer Models Performance comparison

From the table, we can clearly see that the combination of the Cosine scheduler and the Adam optimizer gave us the best results with an MAD error of ~ 45 cm and FAD error of ~ 1 m. This makes sense since the Adam optimizer, unlike SGD, maintains a separate learning rate for every network parameter and adapts this learning rate during training using estimates of the first and second moments of the gradients. Adam is also computationally much faster than SGD and is often the default optimizer chosen for deep learning. Although the Adam optimizer computes adaptive learning rates, it can benefit from the use of a global learning rate scheduler. The performance of the Adam optimizer was significantly better than SGD with momentum and Nesterov's acceleration. However, the difference in the performance of the Adam optimizer with different learning rate schedulers was not that significant. Since we got the best performance from the Adam + Cosine scheduler combination, we chose that to be our best model.

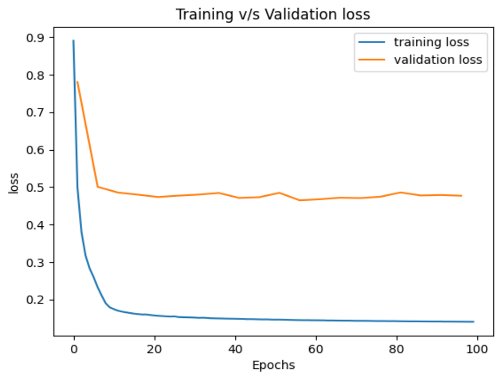
Once we found our best model, we tested the model on the test dataset from TrajNet. Figures 2 and 2c show the training loss, validation loss, MAD and FAD as a function of the number of epochs.



(A)



(B)

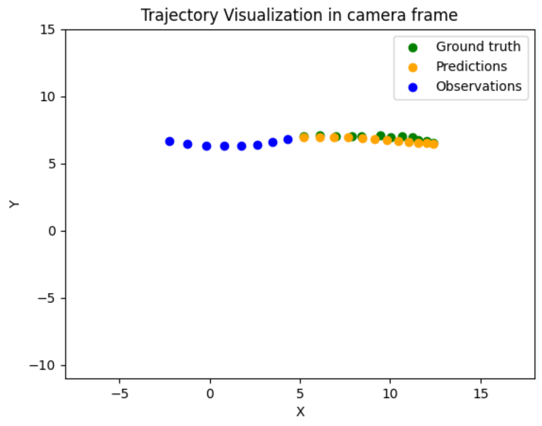


(C)

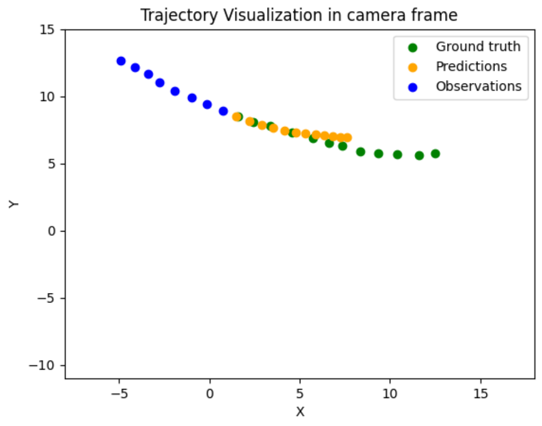
FIGURE 2. MAD, FAD distance metrics (meters) and training and validation loss curves as a function of the number of epochs.

4.3. **Qualitative Results:**

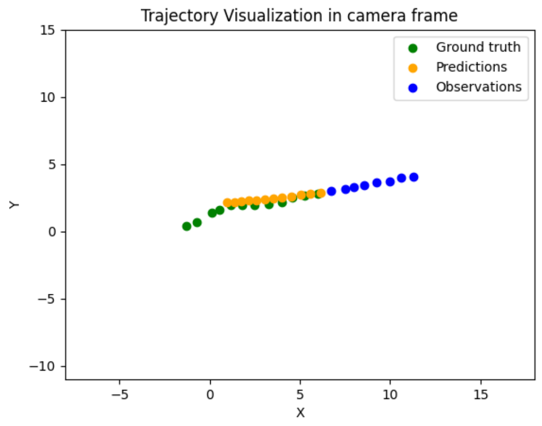
To further motivate the quantitative results presented above, we plot the predictions from the Transformer model along with the ground truth for a few randomly selected pedestrians 3. We also plot the observation points that the model uses to make the predictions. The results were quite impressive as we can see that the predictions closely follow the ground truth.



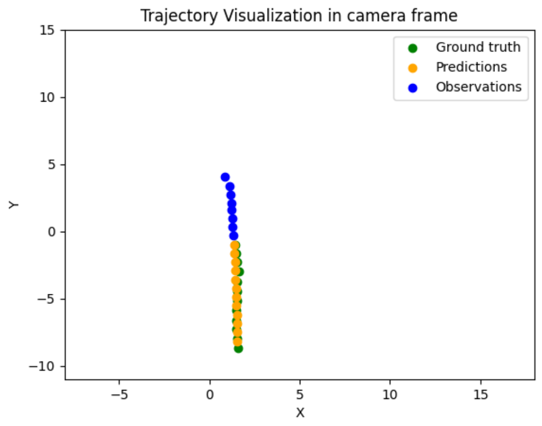
(A)



(B)



(C)



(D)

FIGURE 3. Visualizations of the predictions along with the ground truth and observations for four random pedestrians. The units of measurements for the above figures are in meters (m)

5. DISCUSSION

In this project, we implemented attention-based Transformers for the prediction of human trajectories. We used the TrajNet dataset for training and testing as it is one of the largest and most diverse datasets for trajectory forecasting. We tested various learning rate schedulers and optimizers and found that the combination of Adam optimizer and Cosine scheduler gave the best results for the dataset with a mean error of ~ 45 cm. Qualitative visualizations of the pedestrian trajectory predictions were comparable to the ground truth. Further, we wish to test the model for longer prediction horizons and more complex human trajectories. We would also like to test the capability of the model to handle missing input observations which is inevitable when working with real sensor data.

References

[1] Giuliari, F., Hasan, I., Cristani, M. and Galasso, F., 2021, January. Transformer networks for trajectory forecasting. In 2020 25th international conference on pattern recognition (ICPR) (pp. 10335-10342). IEEE.

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems, 30.

[3] Xu, Y., Piao, Z. and Gao, S., 2018. Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 5275-5284).

[4] Yi, Shuai, Hongsheng Li and Xiaogang Wang. "Understanding pedestrian behaviors from stationary crowd groups." 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2015): 3488-3496.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural computation, vol. 9, no. 8, pp. 1735-1780, 1997

[6] S. Becker, R. Hug, W. Huber, and M. Arena, "An evaluation of trajectory prediction approaches and notes on the trainer benchmark," arXiv preprint arXiv:1805.07663, 2018.

[7] C. K. I. Williams, "Prediction with gaussian processes: From linear regression to linear prediction and beyond," in Learning in graphical models. Springer, 1998, pp. 599–621.

[8] M. B. Priestley, Spectral analysis and time series. Academic Press 1981.

[9] H. Akaike, "Fitting autoregressive models for prediction," Annals of the institute of Statistical Mathematics, vol. 21, no. 1, pp. 243-247, 1969.

[10] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in ICCV, 2009.

[11] J. Ferryman and A. Shahrokni, "Pets2009: Dataset and challenge," in 2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, Dec 2009, pp. 1–6.

[12] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in Computer Graphics Forum, 2007.

[13] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in European conference on computer vision. Springer, 2016, pp. 549–565.