Human Trajectory Prediction Using Transformer and LSTM Models

Bingcheng Chen, Arvin Rokni
(Dated: October 2023)

Abstract—Both LSTM and Transformer architectures have shown significant capability in trajectory prediction, and these models differ in terms of parameters and data preprocessing techniques. In this paper, we present the implementation of three distinct trajectory prediction models from the ground up, the first is a conventional LSTM model, the second model leverages a transformer architecture with customed modification, and the third model combines an encoder-decoder LSTM with multi-head self-attention mechanism. Among these models, the Transformer model emerges as the top performer in this paper, and it is followed closely by the LSTM model, the LSTM with multi-head Attention model performed the least effectively. To access the code used in this paper, it is available at https://github.com/chenbin234/SSY340_Deep_machine_learning/tree/main/Project

I. INTRODUCTION

Predicting how people move is vital for autonomous systems in various fields like self-driving cars, video surveillance, and tracking objects. In an increasingly automated world, human trajectory prediction is not just a technological achievement but a matter of safety and efficiency. Human movement can be highly dynamic and influenced by a wide range of factors, and for this reason it is a challenging task to predict the trajectory of and individual. In this paper, we explore the idea: using Transformer and Long short-term memory (LSTM) networks to predict how people will move. Both Transformer and LSTM models offer solutions to these challenges through their ability to capture long-term dependencies, model complex interactions, and adapt to various trajectory patterns, making them valuable tools in this field.

Inspired by the research paper called "Transformer Networks for Trajectory Forecasting," our goal is to use a type of LSMT and Transformer network to predict where people will go in a dynamic scene. By using LSTMs and Transformers for this, we aim to improve how we predict movement and make autonomous vehicles safer by avoiding accidents and planning better routes.

The dataset that is used for this paper is the TrajNet data set. TrajNet consists of 3161 human trajectories, observing for each trajectory 8 consecutive ground-truth values in world plane coordinates and forecasting the following 12. The dataset is a collection of four different families, each representing diverse scenarios. In total, TrajNet contains 11,448 trajectories, challenging models to perform effectively across a wide range of practical scenarios. [1]

II. BACKGROUND THEORY

Prior to the emergence of the transformer architecture, LSTM models were widely regarded as the top performers, owing to their suitability for time-series sequence problems. However, with the advent of transformer, researchers has shifted their attention to exploiting the power of transformer for trajectory prediction.

First, LSTM [2] represents a specific variant of recurrent neural network (RNN) architecture, which is designed to be able to capture and remember long-term information. As is shown in Figure 1, a LSTM cell has four gates: Input(i), Forget(f), Modulation(g), and Output(0). The Input Gate lets new information in, the Forget Gate helps remember or forget old information, the Modulation Gate proposes possible updates, and the Output Gate decides what to show as output. These 4 gates help LSTMs manage information and learn patterns from data sequences.

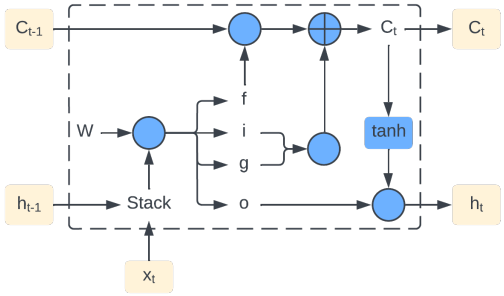

FIG. 1: LSTM Cell

The mathematical formulas for the operations performed within a LSTM cell are as follows:

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (1)$$
$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (2)$$
$$g_t = \tanh(W_g \cdot [h_{t-1}, x_t] + b_g) \quad (3)$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (4)$$
$$c_t = f_t \cdot c_{t-1} + i_t \cdot g_t \quad (5)$$
$$h_t = o_t \cdot \tanh(c_t) \quad (6)$$

Moving on to Transformer architecture, it is characterized by its self-attention mechanism [3], which allows it to

capture relationships between tokens in a sequence. The self-attention in transformer architecture is typically expressed as follows.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (7)$$

III. METHOD

In this report, we explored three main related architecture of models: the first has focus on classical LSTM model. the second was encoder-decoder transformer, and the third was encoder-decoder LSTM with multihead attention.

A. LSTM Model (Many-To-Many)

As illustrated in Figure 2, we first implemented the conventional LSTM architecture from scratch. For this purpose, we configured the model with a hidden size of 128 and incorporated three hidden layers. Additionally, the input sequence length was set to 8 and output sequence length set to 12. To further process the model's outputs, we stacked all the outputs from the final hidden layer and fed them through a two-layer Linear network. To evaluate the performance of our LSTM-based models, we employed MSE loss function:

$$\text{Loss} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

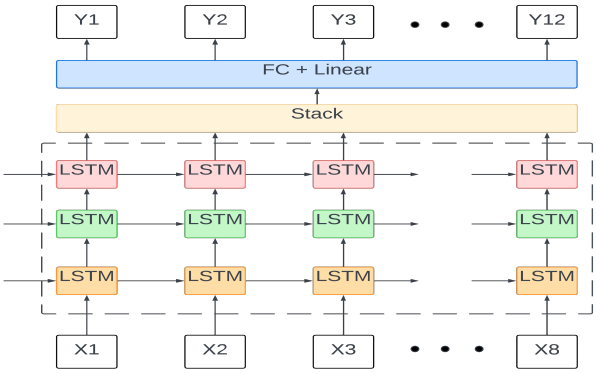

FIG. 2: LSTM (many-to-many)

B. Encoder-decoder Transformer

As is shown in Figure 3, the transformer has a encoder and decoder structure, the encoder processes input trajectories and the decoder generates the future trajectories. Compared with the classical transformer architecture, we add one more norm layer to both the encoder and decoder before the data is fed into the multi-head self attention layer, expecting an improved generalization ability to unseen data.

We also changed the activation function to GELU [4] activation, which, based on its probability under a Gaussian distribution, gates the input by its sign, in contrast to the RELU activation. It combines a scaled linear component $(0.5x)$ with a scaled hyperbolic tangent component $(1 + \tanh(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)))$. This combination can help to capture a wider range of input values and can mitigate the vanishing gradient problem.

$$\text{GELU}(x) = 0.5x\left(1 + \tanh\left(\sqrt{\frac{2}{\pi}}(x + 0.044715x^3)\right)\right) \quad (8)$$

Compared with Sharon's work [5], we also removing the dropout layers considering that in this specific case that the dataset is not big and the model has generalized well, this has been confirmed by the performance on the validation data.

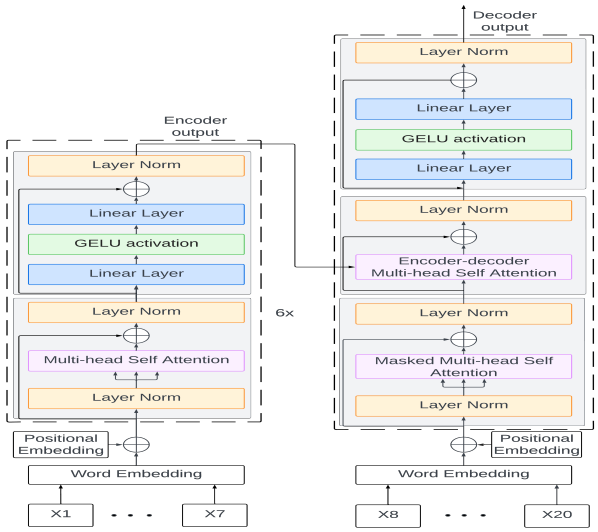

FIG. 3: Encoder-decoder Transformer

C. Encoder-decoder LSTM With Multihead Attention

In this section, inspired by the work of Yao Qin et al [6], we consider for trajectory prediction a third model which combine multi-head self attention with encoder-decoder LSTM architecture, as illustrated in Figure 4.

In the encoder part, an input multi-head attention (2 attention heads) is applied after the input sequences are processed by a word embedding layer (embedding size is 8) and norm layer. For each 'Encoder' in the figure it has 2 hidden layers, and we set hidden size to 128.

In the decoder part, another multi-head attention mechanism is used, the query comes from the output of the decoder in previous time step h_{t-1}^{D}, the keys and values comes from the corresponding output of the encoder $\tilde{h}_{t}^{D}$. we set the 2 head attention, the output of the cross multi-head attention is then fed into the decoder, which has the same structure as encoder, to generate the output of decoder.

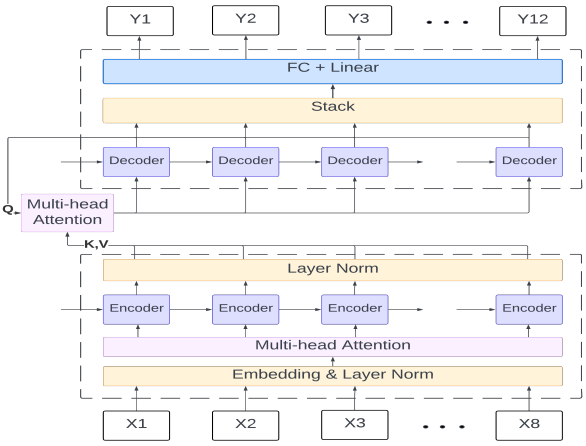

FIG. 4: Encoder-decoder LSTM With Multihead Attention

IV. RESULT

Same as the approach employed in other related papers, the evaluation of the model's performance is conducted through the use of two metrics: MAD and FAD. MAD, which stands for Mean Average Displacement, calculates the average discrepancy at each time step, providing insights into the model's overall performance throughout the entire time sequence prediction. On the other hand, FAD, or Final Average Displacement, specifically assesses the model's ability to capture the discrepancy at the final time step, evaluating on the models' effectiveness in predicting the final state.

Recalling the TrajNet dataset that we utilized, we are given 8 consecutive ground-truth values i.e., $t-7, t-6, \ldots, t$, the primary objective of the model is to predict the subsequent 12 values, spanning from $t+1$ to $t+12$. To compute the metrics MAD and FAD for each 12-value sequence prediction, the following formulas are applied:

$$\text{MAD} = \frac{1}{12}\sum_{t=1}^{12} \|y_t, \hat{y}_t\|$$

$$\text{FAD} = \sum_{t=12}^{12} \|y_t, \hat{y}_t\|$$

where, $\|y_t, \hat{y}_t\|$ is the euclidean distance between prediction and ground truth in time step t.

As depicted in Figure 5, after training for 100 epochs respectively for each model, to facilitate a clear representations of our predictions, we chose three distinct examples for visualization, showcasing the observations, ground truth data, and the models' predictions. The figures reveal that all the predictions closely align with the ground truth trajectory to a significant degree.

Within Figure 6, we made a comparison of the MAD and FAD metrics across the three distinct models, each evaluated across 100 epochs of training. In terms of MAD metrics, the Transformer model exhibited the best performance, achieving MAD reduction to $0.452m$, following closely, the LSTM model got a good performance as well, which is $0.49m$, the LSTM with attention model exhibited the least effective performance with a relatively higher MAD score of $0.713m$.

When considering the FAD metrics, The LSTM models outperformed the Transformer model a little bit, which were $0.973m$ and $0.998m$ respecticely, the LSTM with attention model continued to exhibit the least effective performance.

In Table I, we present a comparison of our models' performance against recent approaches. To determine the ranking, we based our assessment on the average of Mean Average Displacement (MAD) and Final Average Displacement (FAD) metrics. In conclusion, our transformer model performs best,just as well as Sharon's Transformer model, then followed by LSTM, with LSTM with Multi-head Attention performing the least effectively.

Rank	Method	Avg	FAD	MAD	Cit	Year
2	*Transformer*	*0.725*	*0.998*	*0.452*		*2023*
2	Transformer_sharon	0.725	1.0	0.45	[5]	2022
4	*LSTM(Many-to-Many)*	*0.732*	*0.973*	*0.490*		*2023*
5	TF	0.776	1.197	0.356	[7]	2020
6	TFq	0.858	1.300	0.416	[7]	2020
7	*BERT*	0.879	1.354	0.440	[7]	2020
8	*BERT NLP pretrained*	0.902	1.357	0.447	[7]	2020
9	*LSTM With Attention*	*0.975*	*1.236*	*0.713*		*2023*

TABLE I: Results of different models, *Blue italic indicates methods implemented in this work.*

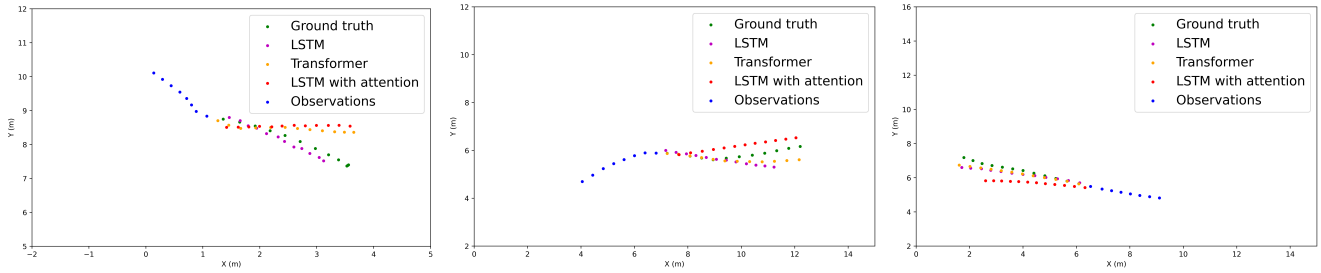

FIG. 5: Examples of Observation, Ground Truth and Predictions by three models

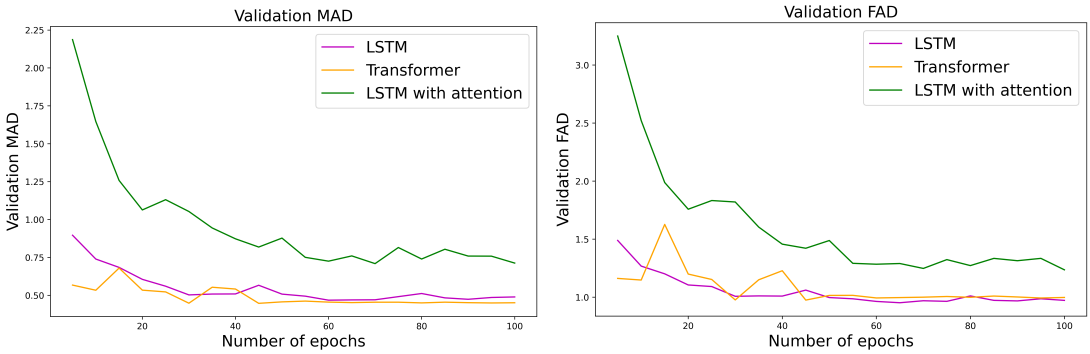

FIG. 6: MAD and FAD of three models

V. CONCLUSION

In this paper, we proposed 3 distinct models for human trajectory prediction: LSTM (Many-to-Many), Encoder-decoder Transformer and Encoder-decoder LSTM With Multi-head Attention. Among these models, the Transformer model emerged as the top performer, closely followed by the LSTM, while the LSTM with Multi-head Attention exhibited comparatively lower effectiveness.

It's important to note that the LSTM with attention model did not meet our expectations in terms of performance. However, we believe that there remains potential in enhancing its capabilities. Further experimentation in this architecture could improve its predictive ability.

Furthermore, our approach in this paper focused on modeling each individual's trajectory independently. However, we acknowledge that a more complex scenario, which considers interactions between individuals, should be explored in future research.

VI. REFERENCES

[1] W. H. M. A. Stefan Becker, Ronny Hug, Trajnet: A trajectory forecasting and behavioral prediction dataset, https://paperswithcode.com/dataset/trajnet-1 (2018).

[2] Manu, A simple overview of rnn, lstm, and attention mechanism, Medium (2021).

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, Attention is all you need (2017), arXiv:1706.03762 [cs.CL].

[4] D. Hendrycks and K. Gimpel, Gaussian error linear units (gelus) (2023), arXiv:1606.08415 [cs.LG].

[5] S. R. Shaji, Trajectory prediction transformers, `https://github.com/sharonrichushaji/trajectory-prediction-transformers` (2022).

[6] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. Cottrell, A dual-stage attention-based recurrent neural network for time series prediction (2017), arXiv:1704.02971 [cs.LG].

[7] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, Transformer networks for trajectory forecasting (2020), arXiv:2003.08111 [cs.CV].