EPT-MoE: Toward Efficient Parallel Transformers with Mixture-of-Experts for 3D Hand Gesture Recognition

Ahed Alboody, Rim Slama

Proceedings of the 10th World Congress on Electrical Engineering and Computer Systems and Sciences (EECSS'24)
Barcelona, Spain - August 19-21, 2024
Paper No. MVML XXX (The number assigned by the OpenConf System)
DOI: TBA

EPT-MoE: Toward Efficient Parallel Transformers with Mixture-of-Experts for 3D Hand Gesture Recognition

Ahed Alboody[1] , and Rim Slama[2]

[1]CESI LINEACT Laboratory, Équipe d'Accueil – UR 7527, CESI, France
13 Avenue Simone VEIL, 06200 Nice, France
aalboody@cesi.fr
[2]CESI LINEACT Laboratory, Équipe d'Accueil – UR 7527, CESI, France
15 c Av. Albert Einstein, 69100 Villeurbanne, France
rsalmi@cesi.fr

Abstract - The Mixture-of-Experts (MoE) is a widely known deep neural architecture where an ensemble of specialized sub-models (a group of experts) optimizes the overall performance with a constant computational cost. Especially with the rise of Mixture-of-Experts with Mixtral-8x7B Transformers, MoE architectures have gained popularity in Large Language Modeling (LLM) and Computer Vision. In this paper, we propose the Efficient Parallel Transformers of Mixture-of-Experts (EPT-MoE) coupled with Spatial Feed Forward Neural Networks (SFFN) to enhance the ability of parallel Transformer models with Mixture-of-Experts layers for graph learning of 3D skeleton-data hand gesture recognition. Nowadays, 3D hand gesture recognition is an attractive field of research in human-computer interaction, VR/AR and pattern recognition. For this purpose, our proposed EPT-MoE model decouples the spatial and temporal graph learning of 3D hand gestures by integrating mixture-of-experts layers into parallel Transformer models. The main idea is to combine the powerful layers of mixture-of-experts that process the initial spatial features of intra-frame interactions to extract powerful features from different hand joints, and then, to recognize 3D hand gestures within the parallel Transformer encoders with layers of Mixture-of-Experts. Finally, we conduct extensive experiments on benchmarks of the SHREC'17 Track dataset in order to evaluate the performance of EPT-MoE model variations. EPT-MoE greatly improves the overall performance, the training stability and reduces the computational cost. The experimental results show the efficiency of several variants of the proposed model (EPT-MoE), which achieves or outperforms the state-of-the-art.

Keywords**:** Mixture-of-Experts (MoE), Parallel Transformers, Hand Gesture Recognition, 3D Skeleton Data, Human-Machine Interaction

1. Introduction

Nowadays, the innovative use of a technique called the "Sparse Mixture-of-Experts" model in Mixtral-8x7B Transformer introduced by Mistral AI [1] have gained popularity in Large Language Modeling (LLM). Sparsely-gated Mixture-of-Experts networks (MoEs) have demonstrated excellent scalability in Large Language Models as Switch Transformer [2], GShared [3] and GLaM [4], in Graph learning [5] and in Computer Vision [6]. A Sparse Mixture-of-Experts (SMoE) language model named Mixtral 8x7B was introduced by Mistral AI in [1]. Mixtral transformer has the same architecture as Mistral 7B [7], with the difference that each Feed-Forward Neural Network (FFN) layer is composed of eight FFN blocks (which we call Experts) and two of experts are used for each Token of the sequence. Mixtral was based on a transformer architecture [8] by using the same modifications as described in [7], with the notable exception that the FFN blocks are replaced by Mixture-of-Expert layers in Mixtral. Mixture-of-Experts is an ensemble of FFN that combines multiple specialized models called "Experts" to make predictions on different subsets of data in a specific domain. The Mistral AI's Mixtral 8x7B model essentially involves training multiple smaller "expert" models, each specializing in specific tasks or domains. When confronted with a specific problem (Prompt-Question) of domain, the MoE model selects a group of experts best suited to handle the particular challenge. This collaborative model of multiple experts allows Mixtral to achieve remarkable accuracy and efficiency, even with a smaller parameter size, and outperforms OpenAI's GPT-3.5.

Motivated by the huge success of Mixtral transformer developed by Mistral AI [1], we propose the *Efficient Parallel Transformers of Mixture-of-Experts (EPT-MoE) deep learning architecture* to study the ability of parallel Transformer models with layers of Mixture-of-Experts (MoE) for graph learning of 3D Hand Gestures Recognition task. A Parallel

Transformers of Mixture-of-Experts (EPT-MoE) refers to a configuration where two or more Transformers with MoE layers are connected in parallel to each other (using ADD operation or Residual skip-connection). Using multiple configurations of parallel Transformers with MoE layers increase the overall performance capacity, improve reliability, optimize the overall performance with a constant computational cost, and provide redundancy in Transformers of Mixture-of-Experts. In our proposed model called EPT-MoE, each transformer model with MoE layers as a group of expert can specialize in recognizing specific types or aspects, a particular part of hand gestures, allowing them to capture subtle patterns and details. By combining in parallel multiple specialized Transformer models with MoE layers provide more robustness, efficient and accuracy in capturing various gestures present in different types/aspects.

The main application of EPT-MoE in this paper is skeleton-based 3D hand gesture recognition [9], [10], [11], [12] which has been an active research topic of computer vision due to its wide range of applications such as human-computer/-robot/-machine interaction, VR/AR and Robotics. It offers many opportunities in different contexts such as industry 5.0 and health-care. Low-cost depth sensors such as Microsoft Kinect and Intel RealSense captured dynamic hand skeleton data (with RGB images and RGB-D data) with high precision. These data are coupled with quick advances in action recognition [13], [14] and hand pose estimation research [15] that allows easily obtaining accurate time-series coordinates of hand joints. Skeleton-based methods, that presented by a sequence of hand joints with 2D or 3D coordinates, are more robust to varying lighting conditions and occlusions given the accurate joint coordinates. Recently, efficient graph learning representation of hand skeleton-data in STr-GCN work [11] coupled with multi-head-attention mechanism in transformer encoder model has demonstrated its capacity to extract powerful spatial-temporal information. In the skeleton-based action recognition [14], [16], [17], [18], using graph learning by Spatial-Temporal Graph Convolutional Neural Networks (GCNs) leads to the most successful ST-GCN work [14]. These networks exploits joint connections using graph convolutions in both spatial and temporal domains. Graph Convolutional Skeleton Transformer (GCST) based on GCNs and Transformer was introduced in [19] for human action recognition. Moreover, the concept of Mixture-of-Experts (MoE) to GNNs was introduced in [5] by proposing Graph Mixture-of-Experts (GMoE) model that empowers individual nodes in the graph to dynamically and adaptively select more powerful spatial-temporal information aggregation experts.

In summary, our contributions in this work are as follows: first, we introduce the ***E**fficient **P**arallel **T**ransformers of **M**ixture-**o**f-**E**xperts (EPT-MoE) architecture* that consists of multiple parallel Transformers with multiple MoE layers. Coupling *EPT-MoE with Spatial Feed Forward Neural Networks (SFFN)* is so useful to extract spatial and temporal correlations between 3D-skeleton sequences, pairs of nodes, taking advantage of the hand graph structure. Then, we present different configurations (several variants) of EPT-MoE for 3D-skeleton sequences processing and theirs influences on the recognition performance, training time, model size, GPU model performance usage and inference. We present the most important achieved and competitive results on main skeleton-based hand gesture benchmarks based on SHREC'17 Track dataset [15] for validation and testing. Parallel Transformers coupled with Mixture-of-Experts has shown the effectiveness of our EPT-MoE architecture for 3D hand gesture recognition. The rest of this paper is structured as follows. In Section 2, related works on Mixture-of-Experts model, Transformer architecture, and Graph Convolutional Neural Networks (GCNs) for 3D hand gesture and action recognition approaches are reviewed. In Section 3, we describe in details our proposed EPT-MoE architecture. In Section 4, we report our experimental results including performance evaluations for 17 several variants of EPT-MoE architecture. We conduct a discussion and a conclusion in Section 5.

2. Related Works

In this work, we focus on the traditional architecture of Transformers based upon Self-Attention/Multi-Head-Attention [9], [11], [16], [18], [20], [21] and the concept of Mixture-of-Experts transformer as Mistral AI [1], Switch Transformer [2], GShared [3] and GLaM [4]. Then, we introduce the concept of Parallel Transformer with Mixture-of-Experts layers for skeleton-based 3D hand gesture recognition. Skeleton-based 3D hand gesture recognition is an active research topic due to its wide range of applications such as human-computer interaction, robotics and VR/AR. With the rise of deep learning, it has been studied in recent years in many advanced skeleton-based methods [9], [10], [11], [13], [14], [15], [16], [17], [18], [20], [21], [22]. In this work, we focus on recent hand gesture and action recognition related works that are based on Graph Convolutional Networks (GCNs) [14], [17].

2.1. Transformers and Mixture-of-Experts (MoE) Models

The concept of Mixture-of-Experts (MoE) model has been studied in the Deep and Machine Learning (DL/ML) community [2], [6], [23], [24]. Nowadays, spurred by advancements in Large Language Models (LLMs) [1], [2], [3], [4], sparse MoE has re-gained prominence with the arrival of Chat/ Mixtral/ Mistral 7B transformers of Mistral AI [1], [7] as Large Language Models. The new trend is to scale these LLMs models to have much greater performance with more or less stable costs. Turns out that the performance of an LLM positively correlates with the model size and scalability. The remarkable success of sparse MoE in the realm of LLMs has spurred its adoption in diverse domains, including vision, multi-modal, and multi-task learning [1], [7]. The MoE main idea is explained by replacing the original FFN layer with a group of MoE layer in the standard Transformer architecture [8]. FFN layers plays a crucial role, typical following the multi-head-attention and normalization layers. Generally, MoE layer consists of a set of an ensemble of specialized models called "Expert Networks" to make predictions on different domains. Each expert network can be a simple Feed-Forward Neural Network (FFN) layer (i.e. Experts) as proposed in Mixtral transformers [1], [7] with a router to select the Expert Network for a specific domain. In [3], GSHard model [3] is an illustration of scaling of Transformer Encoder with MoE Layers. The MoE layer replaces the every other Transformer FFN layer. The encoder of a standard Transformer model is a stack of self-attention and FFN layers interleaved with residual connections and layer normalization. By replacing every other FFN layer with a MoE layer, they get the GSHard model structure of the MoE Transformer Encoder. As similar to GSHard, authors in [4], they proposed GLaM (Generalist Language Model) which uses a sparsely activated MoE architecture to scale the model capacity. In the same context, Vision MoE (V-MoE) [6] is a sparse version of the Vision Transformer scalable with the largest dense networks applied to image recognition on ImageNet.

In this paper, we propose the Efficient Parallel Transformer with MoE layers for 3D hand gestures recognition. For this purpose, we interest to explore and study the integration of parallel Transformer with MoE layers of Mixtral transformers into Parallel Transformer Encoders with MoE, Graph Convolutional Neural Networks and Spatial FFN. In the next section, we present EPT-MoE architecture that extracts spatial and temporal correlations between 3D-skeleton sequences, pairs of nodes, taking advantage of the hand graph structure. We inspired our proposed EPT-MoE from Mixtral of experts [1], [7] and the main application from STr-GCN work [11].

3. Efficient Parallel Transformers with Mixture-of-Experts (EPT-MoE)

In this section, we present the Efficient Parallel Transformers with Mixture-of-Experts (EPT-MoE) architecture that take advantages of multiple parallel Transformer Encoder coupled with MoE models for capturing spatial temporal features in sequences of 3D hand skeletons. Fig. 1 describes the concept of Parallel Transformer with Mixture-of-Experts layers compared with Standard Transformer [8], GShard [3], and Mixtral [1]. The EPT-MoE architecture consists of multiple parallel Transformer encoders stacked in parallel configuration, each Transformer encoder's output is being added to all encoder's output together or to the residual stream.

In Fig1.a, the encoder of a standard Transformer model is a stack of self-attention (Multi-Head Self Attentions) and Feed Forward Neural layer (FFN) interleaved with residual connections and layer normalization. In Fig1.b, the MoE layer in Mixtral transformers [1], [7], GSHard model [3] replaces the every other Transformer feed-forward layer (FFN). They get the model structure of the MoE Transformer Encoder. In Fig1.c, the proposed model (EPT-MoE) consists of P parallel Transformer encoders with MoE layers. EPT-MoE refers to a configuration where two or more Transformers (P parallel Transformers) with MoE layers are connected in parallel to each other (using ADD operation or Residual skip-connection). First, we apply MoE layer on an input sequence to extract initial features by activation two or more experts (TopK= 2, 3, etc.) of the total number of experts (M) in MoE layer. Then, these features are passed via linear layers to be processed by P parallel Transformers with the Multi-head Attention and MoE layers. Each MoE layer (i.e. MoE layer 1, the bottom block) is interleaved with Transformer block (the upper block). For each input of p parallel Transformer encoder (I), we have an output of the Encoder p, which connect to ADD operation. This ADD operation takes the sum of the all outputs of the P parallel Transformer encoders. The ADD output is assigned to K of the M experts by a router of the MoE layers. The MoE layer's output is the weighted sum of the outputs of the K selected experts. In EPT-MoE, an expert is an adapted non-linear

FFN block with GELU (Gaussian Error Linear Unit) activation function [25], [26] as similar to SwiGLU activation function in Mixtral [1] and a vanilla transformer architecture. The final linear layer is considered as FFN classifier, which interprets the final output of the MoE layer. When scaling to multiple devices, the P parallel Transformer encoders with MoE layers is sharded across devices. We note the number of Parallel Transformer encoder (P), the number of layers in the P Transformer encoder (LP) and the total number of MoE layer experts in P Transformer encoder (P).

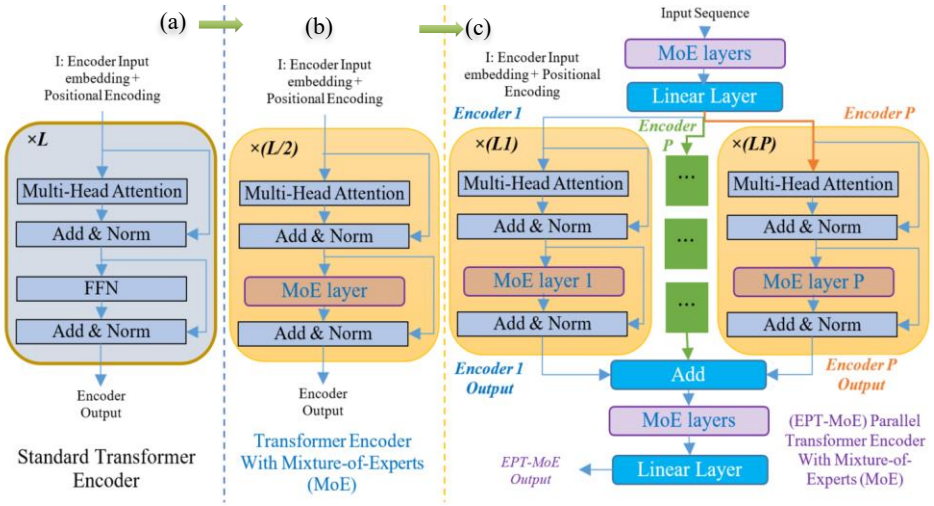

Fig.1. Illustration of the concept of the Efficient Parallel Transformers with Mixture-of-Experts (EPT-MoE) Architecture. (a) The Encoder of a standard Transformer model, (b) The MoE layer in transformers [1], [7], (c) Our proposed model (EPT-MoE) consists of P parallel Transformer encoders with MoE layers. EPT-MoE refers to a configuration where two or more Transformers (P parallel Transformers) with MoE layers are connected in parallel to each other.

Fig. 2 describes in details the different parts of our proposed EPT-MoE architecture for 3D skeleton-based hand gesture recognition. EPT-MoE architecture is extended with a one simple layer of Graph Convolutional Network (GCN) to extract spatial information from graphs and one block of Spatial Feed Forward Neural Networks (SFFN), which consists of three layers respectively: Linear layer, MoE layer, Linear layer (for more details See Appendix 3.). First, we apply a simple layer of GCN on a sequence X_{in} (from t_1, t_2, …, t_i, …t_n) $\in R^{C_{in} \times T \times N}$ of 3D hand skeletons to extract initial spatial features, where each vector $X_{in}^t = \{x_1^t, x_2^t, \dots, x_n^t\}$ represents the 3D coordinates of hand joints at a time stamp t, with T is the sequence length of skeleton sequences for N hand joints. N is the number of nodes. Then, GCN output X_S is connected to the Spatial Feed Forward Neural Networks (SFFN) (see Fig.2). We apply a SFFN on the GCN output X_S to select spatial features using two linear layers and one MoE layer by dispatching of inputs to the selected experts of MoE layer. An expert is an adapted non-linear FFN block with GELU activation function. SFFN output $X_P$is connect to the inputs of the P parallel Transformer encoders to extract spatial-temporal features with the Multi-head Attention of the Transformer block, and MoE layers. Each MoE layer is interleaved with Transformer block. For each input of Transformer block in the X_P sequence (SFFN output), we have an output Y_p which is connected to the Router R_p of the MoE layer. The Router module R_p plays the Gating Network module which dynamically selects K (two, or more) the most relevant experts out of M experts (4, 8, etc.), which is represented by experts from "Expert_1 (FFN1)" to "Expert_M (FFNM)" in the MoE layer. The weighted average of the outputs from these K experts (one, two, or more) will then be connected to the next layer. For the next input in the X_P sequence, K (two, three or more) different experts will be selected. Each input in the X_P sequence gives an output Y_p which is connected to the Router R_p of the MoE layer to be assigned to K of the M experts by the Router R_p. The MoE layer's output U_p is the weighted sum of the outputs of the K selected experts. In EPT-MoE, for each Transformer encoder, we add the MoE layer's output U_p of the P

Parallel Transformer encoders $O_P = \sum_{p=1}^{p=P} U_p$. This output is passed to a new MoE layer for a new selection of the final decision. The final linear layer (used as FFN classifier) interprets the output Z of the MoE layer.

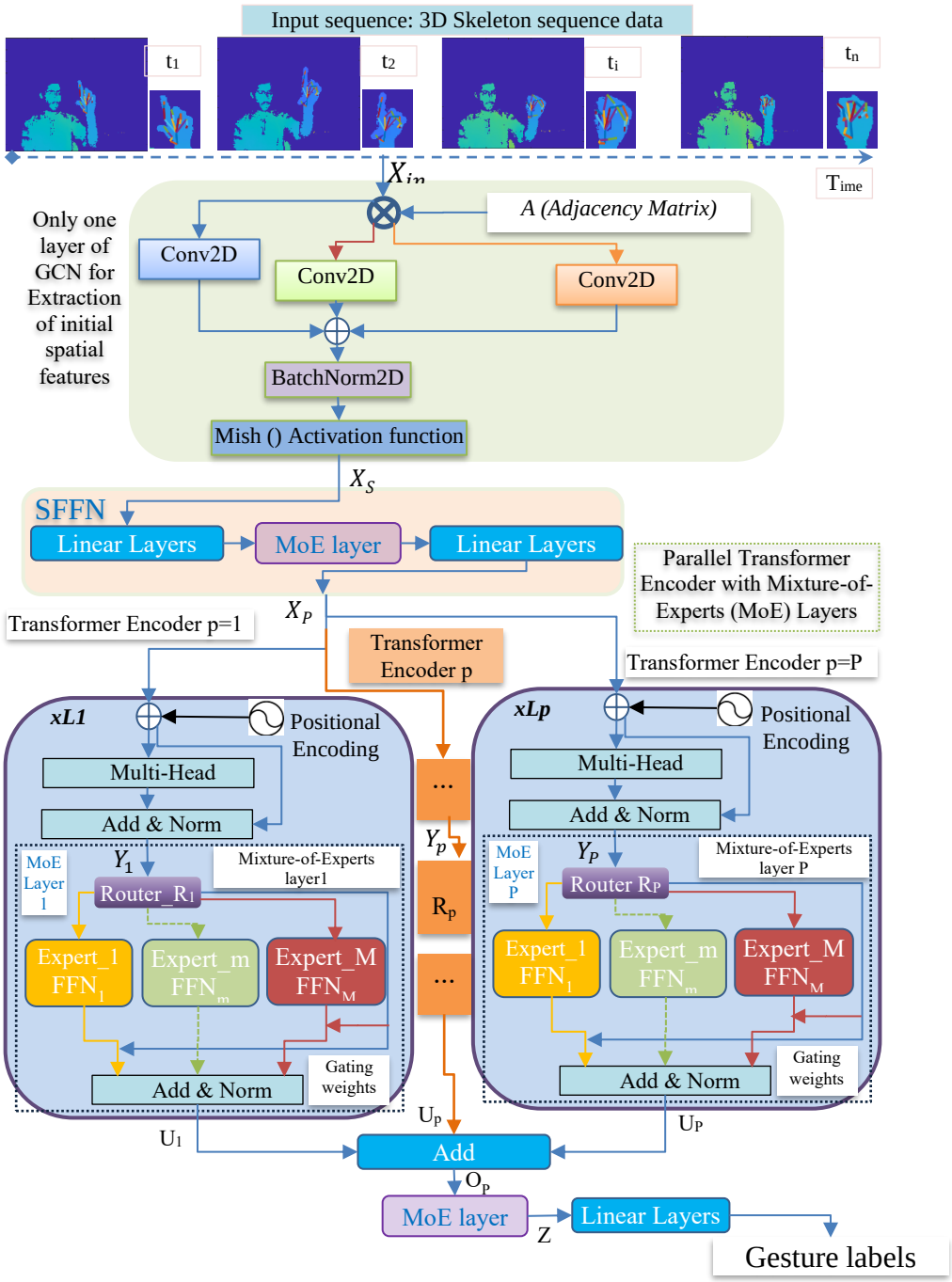

Fig.2. Architecture of the Efficient Parallel Transformers with Mixture-of-Experts (EPT-MoE) for 3D Hand Gesture Recognition.

To scale Transformer models for 3D hand gesture recognition, we introduce parallel transformer encoders with MoE layers by replicating the number of Transformer and replacing each Feed Forward Neural Network (FFN) in Transformer encoder with a sparse mixture of independent FFNs (which we call experts). A learnable router layer selects which experts are chosen (and how they are weighted) for every individual 3D skeleton-data of gesture's sequences. That is, different sequences from the same 3D skeleton-data of gesture's sequences may be routed to different experts into different parallel transformer encoders. Each individual 3D skeleton-data of gesture's sequences is only routed to at most K (typically 1, 2, or more than 2) experts, among a total of M experts (in our experiments, M is typically 8 or 16). This allows scaling the model's size while keeping its computation per gesture's sequences roughly constant.

The principal contribution is introducing the Efficient Parallel Transformer Encoders with Mixture-of-Experts layers into a new Mixture-of-Experts Transformer architecture, which extracts spatial and temporal correlations between 3D-skeleton sequences of the hand graph.

The main differences between our EPT-MoE architecture (see Fig.2) and STr-GCN [11] are summarized by:

(1) The main idea is introducing the concept parallel transformer encoders with Mixture-of-Experts (MoE) layers;

(2) Using multiple parallel transformer encoders, varying the number of transformer layers (from 6 to 4, 1 layers) and the number of MoE layers in transformer encoders;

(3) Replacing the number of convolutions in GCN by introducing a new spatial linear and MoE layers;

(4) Integrating a MoE layer to the final FFN classifier.

In the context of our work, basic graph convolution neural network [11], [14], [16], [22] and standard Transformer [8], Mixture-of-Experts (MoE) model [1], [2], [4], [7], [27] are summarized into our proposed EPT-MoE architecture.

3.1. Transformer Block of EPT-MoE

Transformer block of the p Parallel Transformer encoder in the proposed model EPT-MoE (see Fig.1 and Fig.2) is inspired from the standard Transformer block [8] where each block has two sublayers: a Multi-Head Self-Attention layer (MHSA) and a fully connected Feed-Forward Neural network layer (FFN). We use only the Multi-Head Self-Attention layer (MHSA) with two-dimensional convolution operations as in STr-GCN [11] to process the input 3D skeleton sequences and their spatial relations by a simple projection into three independent matrices of the joints spatial feature vectors: $Queries: Q_i \in R^{N\times D_q}$; $Keys: K_i \in R^{N\times D_k}$; $and\ Values: V_i \in R^{N\times D_v}$. Respectively: D_q, D_k, and D_v denote the dimensions of queries, keys and values.

In EPT-MoE Transformer, we set the D_q, D_k, and D_v the dimensions of queries, keys and values to the feature size 32 as similar as in [10], [11]. Each operation of queries, keys and values is a two-dimension Convolution (Conv2D with kernel size set to 1 with stride set to 1, the input model dimension set to128, and D_q, D_k, and D_v set to 32). The self-attention is computed between each joint at time stamp t, and its corresponding joint in all frames by the scaled dot-product self-attention denoted and used by Transformer as follows [8], [11]:

$$Attention_i = Attn_i(X_P) = Softmax\left(\frac{Q_i K_i^T}{\sqrt{D_k}}\right)V_i \qquad (1)$$

The operation in Eq. (1) is used to calculate locally spatial-temporal features between the joints. The MHSA mechanism projects the D-dimensional representation (d_{model} : the dimension of model, is the number of features in a sequence X_P) into multiple subspaces with H different sets of learned projections (H: the number of heads). For each of the projected queries, keys and values, single attention head is computed according to Eq. (1). All attention heads (MHSA) are concatenated and projected back to D-dimensional representation, which can be formalized as:

$$MHSA(Q,K,V,X_P) = Concat(Attn_1;:;Attn_i;:;Attn_{H-1};Attn_H)W^o \qquad (2)$$

$Where\ W^o$ is a linear projection from D-dimensional representation (dimension model) and to a 256-dimension feature space (hidden dimension) [8], [10], [11].

The main role of MHSA should allow each head to extract different spatial-temporal features from the graph sequence. In EPT-MoE Transformer (see Fig.1 and Fig.2), we employ the number of heads H=8 parallel attention heads. For each of these, we use D_q, D_k, and $D_v = 32$. We fix the model dimension to $d_{model} = 128$. Meanwhile, we vary the number of encoder layers (L) in the EPT-MoE transformer encoder. Due to the reduced dimension of each head, the total computational cost is similar to that of single-head attention with full dimensionality. The final output of the Transformer block can be formulated as:

$$Y_p = LayerNorm(MHSA(X_P) + X_P) \quad (3)$$

Where X_P is the SFFN output (seen as a sequence), which connected to the input of the first self-attention layer of MHSA. We note Y_p the output of the normalization layer of the Transformer block in Parallel Transformer encoder p.

By the next, we present Mixture-of-Experts (MoE) layers in the p Parallel Transformer encoder. The output Y_p of the normalization layer of the Transformer block is connected directly to the input MoE layer. In our work, we interest to explore and study the integration of MoE layers of Mixtral transformer [1], [7] into EPT-MoE Transformer Encoder.

3.2. MoE Layer of EPT-MoE

The structure of Mixture-of-Experts (MoE) layer of the p Parallel Transformer encoder in the proposed model EPT-MoE (see Fig.1 and Fig.2) consists of a given set of M expert networks $E = \{E_1, \dots, E_m \dots, E_M\}$, and a Router R_p of the MoE layer. This is similar to that used in Mixtral Transformer [1] (or called Gating network G similar to Switch Transformer [2] and GShard [3]) whose output is a sparse M-dimensional vector. The Router module R_p play the Gating Network module which dynamically selects K (two, three or more) the most relevant experts out of M experts (4, 8, 16, 32, etc.), which is represented by the "Expert_1 (FFN1)" and "Expert_M (FFNM)". For a given Y_p the output of Transformer block of the p parallel Transformer encoder (in EPT-MoE transformer, see Fig. 2), the output of the MoE layers is determined by the weighted sum of the outputs of the expert networks, where the weights are given by the gating network's output. i.e. given M expert networks $E = \{E_1, \dots, E_m \dots, E_M\}$, the output of the expert layer is given by Eq. (4):

$$\sum_{m=1}^{M} G(Y_p)_m . E_m(Y_p) \quad (4)$$

Where: $G(Y_p)_m$ *denotes the M-dimensional output of the gating network for the m-th expert, and* $E_m(Y_p)$ *is the output of the m-th expert network.* To implement the gating network $G(Y_p)$, there are multiple alternative ways discussed in many works [1], [2], [3], [28], but a simple and performant one is implemented by taking the *softmax* over the *Top-K logits* of a linear layer as sparsely-gated mixture-of-experts layer in [1], [27]. We use the same implementation proposed in Mixtral Transformer:

$$G(Y_p) = Softmax\left(TopK(Y_p, W_g)\right) \quad (5)$$

Where the notation of the top values $TopK(Y_p, W_g) = (TopK(l))_i \coloneqq l_i$ if l_i is among the top-K coordinates of logits $l_i \in R^M$ and $(TopK(l))_i \coloneqq -\infty$ otherwise. The value of K – the number of experts used per one input of the sequence – is a hyper-parameter that modulates the amount of compute used to process each input of the sequence.

In a Transformer model [8], the MoE layer is applied independently on token and replaces the feed-forward (FFN) sub-block of the transformer block. In Mixtral [1], [7], they used the SwiGLU architecture [25] as the expert function and set K = 2 (only two experts) with the total number of experts equals to eight. Each expert is itself a two-layer feed-forward (FFN) gated by SwiGLU activation function. This means each token is routed to only two SwiGLU sub-blocks with different sets of weights. In EPT-MoE Transformer, MoE layer is applied independently per each input of the sequence and replaces the feed-forward (FFN) sub-block. We apply GELU (Gaussian Error Linear Unit) activation function [25], [26] as the expert

function to gate each expert which is a two-layer feed-forward (FFN) with the model dimension fix to $d_{model} = 128$ and the hidden dimension set to 512. Meanwhile, we did not set K for 2 and it can be one, two or more than experts in order to generate more possible configurations of MoE model to scale EPT-MoE model. This means each input of a sequence is routed to one, or more FFN sub-blocks gated by GELU [25], [26] with different sets of weights.

$$E_m(Y_p) = GELU_m(Y_p) \qquad (6)$$

Taking this all together, we compute the output U_p for an input of sequence Y_p as:

$$U_p = \sum_{m=1}^{M} G(Y_p)_m . E_m(Y_p) = \sum_{m=1}^{M} Softmax\left(TopK(Y_p, W_g)\right)_m . GELU_m(Y_p) \qquad (7)$$

This formulation is similar to Mixtral of Experts [1] and GShard architecture [3]. In Mixtral, they replace all FFN sub-blocks by MoE layers while GShard replaces every other block, and that GShard uses a more elaborate gating strategy for the second expert assigned to each token. In EPT-MoE, for each Transformer encoder, we add the MoE layer's output U_p of the p Parallel Transformer encoders $O_P = \sum_{p=1}^{p=P} U_p$. This output O_P is passed to a new MoE layer for a new selection of the final decision.

$$Z = \sum_{m=1}^{M} G(O_p)_m . E_m(O_p) = \sum_{m=1}^{M} Softmax\left(TopK(O_p, W_g)\right)_m . GELU_m(O_p) \qquad (8)$$

The final linear layer (used as FFN classifier) is two-layer pointwise feedforward network (FFN) classifier interprets the output Z of the MoE layer (see Fig .2). We note O the total possible number of MoE layers in EPT-MoE model equals to the sum of 2 MoE layers and P MoE layers in the P Parallel Transformer encoders.

Compared traditional single MoE Transformer with EPT-MoE architecture, we summary some differences : (1) Integrating parallel Transformer encoders with MoE layer; (2) replacing SwilGLU [1] by GELU; (3) extending the number of experts used per one input of the sequence (K) to be more than 2 [1]. The main advantages of parallel transformer with MoE layers: first, if one increases p (number of parallel Transformer) while keeping the total number of layers (Lp=4) fixed, one can decrease the model's parameter count (See Appendix: Model 15 and Model 17 in Table 4.) while enhancing its computational cost effectively and its recognition performance. Secondly, if one increases M (number of experts) while keeping K fixed, one can increase the model's parameter count while keeping its computational cost effectively constant. Thirdly, it can be run efficiently on single GPUs with high performance specialized kernels. Finally, EPT-MoE architecture parameters are summarized in Table 1. (See Appendix for more details see Table 3. and Table 4.).

4. Experimental Results

In this section, we provide details about the datasets that we use for our experiments. Then, we provide details about the Efficient Parallel Transformers with Mixture-of-Experts (EPT-MoE), which is considered a family of sparse EPT-MoE parallel transformer encoders coupled with MoE layer and spatial FFN and graph convolutional neural network (GCN) to recognize 3D hand gestures. Therefore, we first elaborate our EPT-MoE model transformer variations, hyper-parameters, training settings, and training and evaluation protocols. We present different configurations (several variants) of EPT-MoE for 3D-skeleton sequences processing and theirs influences on the recognition performance, training time, model size, GPU model performance usage and inference. We discuss our EPT-MoE model variations with training details and analysis of the results obtained by EPT-MoE. We present the most important achieved and

competitive results on main skeleton-based hand gesture benchmarks based on SHREC'17 Track dataset [15] for validation and testing. Finally, a comparison of our results with the state-of-the-art approaches is performed based upon on SHREC'17 datasets. Parallel Transformers coupled with Mixture-of-Experts has shown the effectiveness of EPT-MoE architecture for 3D hand gesture recognition.

4.1. Datasets and evaluation metrics

In our experiments, we train and evaluate EPT-MoE Transformer using datasets from SHREC'17 TRACK [15]. It is one of the first benchmark for the recognition of hand gestures recorded in two configurations: 14 gestures performed using one finger and 28 gestures performed with the full hand. Each gesture is performed between 1 and 10 times by 28 participants totalling 2800 sequences [11]. We split these sequences into 70 for training datasets and 30 split for test/validation datasets. The SHREC'17 datasets is evaluated into one case for 14 gestures. For the evaluation metrics, we report the recognition test accuracy and the confusion matrix in our experiments.

4.2. EPT-MoE Model Variations, Hyper-parameters and Training Setting

Before going into the details of training efficiency, we first investigate the effect of various design choices on building EPT-MoE Transformer. In our work, we train several variants of EPT-MoE transformer (presented in Table 3. See Appendix) to study the behaviour of parallel transformer and MoE models on the same training data (herein SHREC'17 datasets). Table 1 shows the architectures and sizes of EPT-MoE models with hyper-parameter settings of different scale EPT-MoE models ranging model size from 34.679 (MB) to 95.064 (MB) that we trained in our experiments. We note P the total number of p Parallel Transformers, L is the number of encoder layers, H is the number of Heads, M is the number of experts in MoE layers, K the number of experts per Token in MoE layer, and O is the total number of MoE layers in all blocks of the EPT-MoE model.

To study the effect of scaling the total number of P parallel Transformer encoders and MoE layers. First, we fix the GCN architecture as shown above in Section 3. The size of GCN architecture is 3.2 KB. In addition, we fix the linear layers of the linear classifier with size 131.0 KB. Then, we start by setting H the number of heads for the multi-head self-attention (MHSA) to H=8, and the model dimension $d_{model} = 128$. We fix the input sequence length T=8. We vary the total number P of parallel Transformer encoders, then varying the number Lp of encoder layers to each p parallel Transformer, the total number M of experts in each MoE layer, and the K the number of experts used per one input of the sequence in each MoE layer (See Table 1 and Table 3). For our hyper-parameters and training setting, a batch size of B=32 was chosen for training. We choose the Adam optimizer with $\beta1 = 0.9$, $\beta2 = 0.98$ and epsilon $\varepsilon = 10^{-9}$. We varied the learning rate over the course of training with an annealing learning rate that drops from 1e-3 to 5e-6. The initial learning rate was set to 1e-3, reduced by a factor of 0.5 if the learning stagnates and the loss does not improve in the next 5 epochs. The training stops if the validation accuracy does not improve in the next 50 epochs. For SHREC'17 datasets, we uniformly sample eight frames from each video as the input of GCN. The dropout for all layers set to 0.1 and the loss function is Cross-entropy.

Table 1: Architectures and sizes for 4 variants of EPT-MoE model, Recognition accuracy (%) on test datasets for 4 model variations of the EPT-MoE architecture trained on SHREC'17 (14 Gestures) using GPU Model Performance (Tesla V100-SXM2-16GB).

Name of Project RUNS on wandb.ai	EPT-MoE Model N°	EPT-MoE Transformers with MoE Layers						Model params size (MB)	Recognition test accuracy (%)	Mean of GPU Power usage (Watt)	GPU Memory Allocated (%)	GPU Power Usage (%)
		Parallel Transformer			MoE Layers			$n_{parameters}$				
		p	L_p	H	M	K	O					
festive-festival-100	Model 1	1	L_1=4	8	16	2	2	58.977	91.34	47.96	10.57	15.96
beaming-festival-116	**Model 2**	**1**	**L_1=4**	**8**	**8**	**3**	**3**	**40.974**	**92.18**	**46.23**	**16.33**	**15.36**
spring-energy-141	**Model 3**	**4**	**L_1=1** **L_2=2** **L_3=3** **L_4=4**	**8**	**8**	**2**	**5**	**76.430**	**92.63**	**46.15**	**15.55**	**15.43**
toasty-field-8	**Model 4**	**3**	**L_1=1**	**8**	**8**	**2**	**3**	**42.105**	**92.10**	**47.20**	**11.68**	**15.73**

			L_2=2 L_3=3									

We evaluate several variants of EPT-MoE model by comparing with the obtained results from SOTA methods for 3D-skeleton hand gesture recognition (see Table 2.). We conduct this comparison on SHREC'17 datasets. Fig.3 shows the overall performance for recognition test accuracy and training for several EPT-MoE models. Undoubtedly, the ensemble of EPT-MoE models brought about a significant performance boost. However, the improvement in accuracy does not come at the price of requiring a greater amount of model parameters. EPT-MoE Model 3 needing 76.2MB parameters achieved an accuracy of 92.63% with a relative time about 1hours and 40 minutes. We conduct our experiments using PyTorch on GPU NVIDIA (Tesla V100-SXM2-16GB), Google Colab, Weight & Bias (https://wandb.ai: AI Platform) (See Appendix).

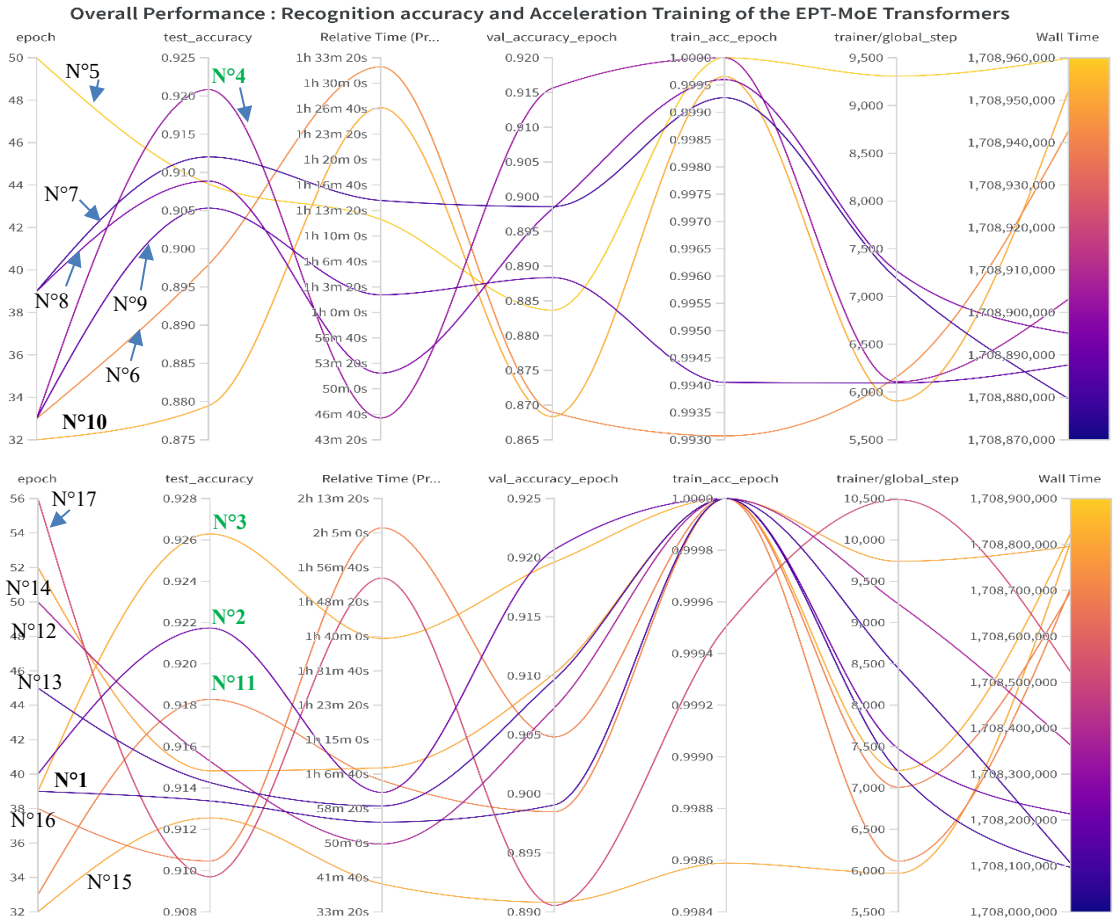

Fig. 3: Overall performance of 17 variants of EPT-MoE model (See Appendix: Reports generated using Wandb based on an extensive experiments and Training EPT-MoE Models) where N° refers to model number as given in Table 3.

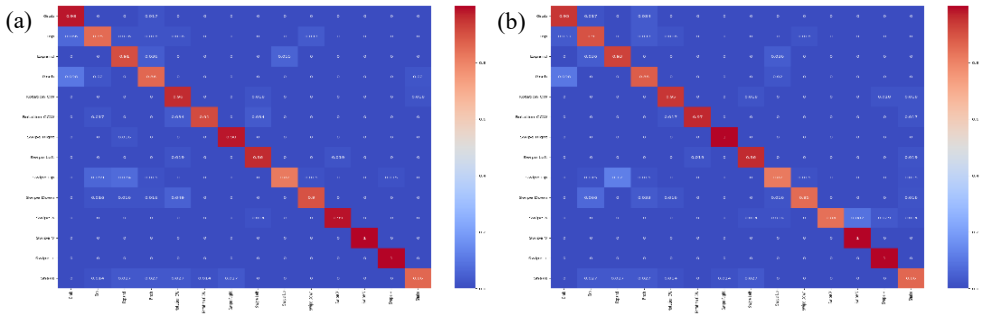

Fig. 4: Confusion matrix of SHREC'17 (14 Gestures) for (a) the best EPT-MoE Model 3 with recognition accuracy 92.63 %, and (b) STr-GCN.

Fig.4 shows the confusion matrix from pre-trained model for the highest performance of EPT-MoE Model 3 compared with our experiments of STr-GCN. We collect most of the results in Table 2 from STr-GCN[3], STA-GCN [16] and DG-STA [9]. EPT-MoE achieves the performance of STr-GCN and STA-GCN methods based on the graph representation of the 3D-skeleton hand.

Table 2: Recognition accuracy for the best model of the EPT-MoE architecture trained on SHREC'17 (14 Gestures) in comparison with three principal state-of-the-art methods.

EPT-MoE Model N°	EPT-MoE Transformers with MoE Layers						Model params size (MB)	Recognition test accuracy (%)	Mean of GPU Power usage (Watt)	GPU Memory Allocated (%)	GPU Power Usage (%)
	Parallel Transformer			MoE Layers			$n_{parameters}$				
	p	L_p	H	M	K	O					
Model 3	**4**	**L_1=1** **L_2=2** **L_3=3** **L_4=4**	**8**	**8**	**2**	**5**	**76.430**	**92.63**	**46.15**	**15.55**	**15.43**
STA-GCN [16]	-	-	-	-	-	-	-	92.70	-	-	-
STr-GCN [11]	1	6	8	0	0	0	-	92.76	-	-	-
DG-STA [9]	-	-	-	-	-	-	-	94.40	-	-	-

4. Conclusion and Perspectives

In this paper, we proposed a new deep learning architecture named EPT-MoE (Efficient Parallel Transformers with Mixture-of-Experts) for 3D skeleton-based hand gesture recognition. EPT-MoE utilizes a parallel configuration of Transformer encoders coupled with Mixture-of-Experts (MoE) layers. The MoE layer of EPT-MoE is inspired from Mixtral/Mistral-7B Transformers. EPT-MoE aims to enhance the ability of parallel Transformer models with MoE layers, Spatial FNN and Graph Convolutional Networks (GCN) for graph learning of hand gesture recognition using 3D hand skeleton data. We explored several variants of EPT-MoE transformer in order to understand theirs behaviours on the same training data. We performed extensive experiments and have demonstrated the robustness of the best variants of our EPT-MoE transformer model in dealing with SHREC'17 Track benchmarks. The experiments show the efficiency of three top-model variations of the proposed EPT-MoE, which achieves or outperforms the state-of-the-art.

In future works, we will interest to propose multi-modal EPT-MoE transformers to exploit multi-sources of RGB Videos with Depth-information, LIDAR and IR modalities. We intend to evolve EPT-MoE architecture for human motion prediction and action recognition tasks required in Human-Machine Interaction applications.

References

[1] A. Q. Jiang *et al.*, “Mixtral of Experts.” arXiv, Jan. 08, 2024. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2401.04088

[2] W. Fedus, B. Zoph, and N. Shazeer, “Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity.” arXiv, Jun. 16, 2022. Accessed: Nov. 28, 2023. [Online]. Available: http://arxiv.org/abs/2101.03961

[3] D. Lepikhin *et al.*, “GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding.” arXiv, Jun. 30, 2020. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2006.16668

[4] N. Du *et al.*, “GLaM: Efficient Scaling of Language Models with Mixture-of-Experts.” arXiv, Aug. 01, 2022. Accessed: Nov. 28, 2023. [Online]. Available: http://arxiv.org/abs/2112.06905

[5] H. Wang *et al.*, “Graph Mixture of Experts: Learning on Large-Scale Graphs with Explicit Diversity Modeling.” arXiv, Oct. 17, 2023. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2304.02806

[6] C. Riquelme *et al.*, “Scaling Vision with Sparse Mixture of Experts.” arXiv, Jun. 10, 2021. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2106.05974

[7] A. Q. Jiang *et al.*, “Mistral 7B.” arXiv, Oct. 10, 2023. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2310.06825

[8] A. Vaswani *et al.*, “Attention Is All You Need.” arXiv, Aug. 01, 2023. Accessed: Nov. 29, 2023. [Online]. Available: http://arxiv.org/abs/1706.03762

[9] Y. Chen, L. Zhao, X. Peng, J. Yuan, and D. N. Metaxas, “Construct Dynamic Graphs for Hand Gesture Recognition via Spatial-Temporal Attention.” arXiv, Jul. 20, 2019. Accessed: Feb. 02, 2024. [Online]. Available: http://arxiv.org/abs/1907.08871

[10] A. Caputo *et al.*, “SHREC 2021: Track on Skeleton-based Hand Gesture Recognition in the Wild.” arXiv, Jun. 21, 2021. Accessed: Feb. 04, 2024. [Online]. Available: http://arxiv.org/abs/2106.10980

[11] R. Slama, W. Rabah, and H. Wannous, “STr-GCN: Dual Spatial Graph Convolutional Network and Transformer Graph Encoder for 3D Hand Gesture Recognition,” in *2023 IEEE 17th International Conference on Automatic Face and Gesture Recognition (FG)*, Waikoloa Beach, HI, USA: IEEE, Jan. 2023, pp. 1–6. doi: 10.1109/FG57933.2023.10042643.

[12] J. Qi, L. Ma, Z. Cui, and Y. Yu, “Computer vision-based hand gesture recognition for human-robot interaction: a review,” *Complex Intell. Syst.*, Jul. 2023, doi: 10.1007/s40747-023-01173-6.

[13] F. Yang, S. Sakti, Y. Wu, and S. Nakamura, “Make Skeleton-based Action Recognition Model Smaller, Faster and Better.” arXiv, Mar. 18, 2020. Accessed: Feb. 02, 2024. [Online]. Available: http://arxiv.org/abs/1907.09658

[14] S. Yan, Y. Xiong, and D. Lin, “Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition.” arXiv, Jan. 25, 2018. Accessed: Feb. 04, 2024. [Online]. Available: http://arxiv.org/abs/1801.07455

[15] Q. D. Smedt, H. Wannous, J.-P. Vandeborre, J. Guerry, B. L. Saux, and D. Filliat, “3D Hand Gesture Recognition Using a Depth and Skeletal Dataset,” *Eurographics Workshop on 3D Object Retrieval*, p. 6 pages, 2017, doi: 10.2312/3DOR.20171049.

[16] W. Zhang, Z. Lin, J. Cheng, C. Ma, X. Deng, and H. Wang, “STA-GCN: two-stream graph convolutional network with spatial–temporal attention for hand gesture recognition,” *Vis Comput*, vol. 36, no. 10–12, pp. 2433–2444, Oct. 2020, doi: 10.1007/s00371-020-01955-w.

[17] Z. Cheng, S. Chen, and Y. Zhang, “Spatio-Temporal Graph Complementary Scattering Networks.” arXiv, Oct. 23, 2021. Accessed: Feb. 04, 2024. [Online]. Available: http://arxiv.org/abs/2110.12150

[18] S.-B. Zhou, R.-R. Chen, X.-Q. Jiang, and F. Pan, “2s-GATCN: Two-Stream Graph Attentional Convolutional Networks for Skeleton-Based Action Recognition,” *Electronics*, vol. 12, no. 7, p. 1711, Apr. 2023, doi: 10.3390/electronics12071711.

[19] R. Bai *et al.*, “Hierarchical Graph Convolutional Skeleton Transformer for Action Recognition.” arXiv, Jan. 10, 2022. Accessed: Feb. 04, 2024. [Online]. Available: http://arxiv.org/abs/2109.02860

[20] R. Bai *et al.*, “Hierarchical Graph Convolutional Skeleton Transformer for Action Recognition.” arXiv, Jan. 10, 2022. Accessed: Feb. 06, 2024. [Online]. Available: http://arxiv.org/abs/2109.02860
[21] C. Plizzari, M. Cannici, and M. Matteucci, “Skeleton-based Action Recognition via Spatial and Temporal Transformer Networks,” *Computer Vision and Image Understanding*, vol. 208–209, p. 103219, Jul. 2021, doi: 10.1016/j.cviu.2021.103219.
[22] Y. Li, Z. He, X. Ye, Z. He, and K. Han, “Spatial temporal graph convolutional networks for skeleton-based dynamic hand gesture recognition,” *J Image Video Proc.*, vol. 2019, no. 1, p. 78, Dec. 2019, doi: 10.1186/s13640-019-0476-x.
[23] Z. Chen, Y. Deng, Y. Wu, Q. Gu, and Y. Li, “Towards Understanding Mixture of Experts in Deep Learning.” arXiv, Aug. 04, 2022. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/2208.02813
[24] T. Gale, D. Narayanan, C. Young, and M. Zaharia, “MegaBlocks: Efficient Sparse Training with Mixture-of-Experts.” arXiv, Nov. 28, 2022. Accessed: Jan. 26, 2024. [Online]. Available: http://arxiv.org/abs/2211.15841
[25] N. Shazeer, “GLU Variants Improve Transformer.” arXiv, Feb. 12, 2020. Accessed: Feb. 09, 2024. [Online]. Available: http://arxiv.org/abs/2002.05202
[26] M. Lee, “GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance.” arXiv, Aug. 01, 2023. Accessed: Feb. 09, 2024. [Online]. Available: http://arxiv.org/abs/2305.12073
[27] N. Shazeer *et al.*, “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer.” arXiv, Jan. 23, 2017. Accessed: Jan. 25, 2024. [Online]. Available: http://arxiv.org/abs/1701.06538
[28] Y. Zhou *et al.*, “Mixture-of-Experts with Expert Choice Routing.” arXiv, Oct. 13, 2022. Accessed: Jan. 26, 2024. [Online]. Available: http://arxiv.org/abs/2202.09368
[29] M. Artetxe *et al.*, “Efficient Large Scale Language Modeling with Mixtures of Experts.” arXiv, Oct. 26, 2022. Accessed: Mar. 05, 2024. [Online]. Available: http://arxiv.org/abs/2112.10684

Supplementary material-Appendix-EPT-MoE Transformer_MVML24_Paper-ID #105

2 Related Works : Spatial-Temporal GCNs for skeleton-based hand gestures, action recognition

Spatial-Temporal Graph convolutional network (GCN) have emerged as a powerful approach for learning graph representations. Recently variants of GCN have been proposed [9], [10], [11], [13], [14], [15], [16], [17], [18], [20], [21] achieving state-of-the-art performance in graph learning tasks of hand gesture and action recognition. GCNs have been widely adopted in skeleton-based action recognition due to its effective representation of the human body- and hand-based graph. The first built model was a spatial-temporal graph convolutional network (ST-GCN) [14] for action recognition, in which each layer contains a graph convolutional block and a temporal convolutional block. The HG-GCN [22] was as a simple model of spatial-temporal graph convolutional network for hand gesture recognition which was built based upon ST-GCN [14]. STr-GCN work [11] is the most recent graph based approach which is based on Spatial Graph Convolutional Networks (SGCNs) [14], [17], coupled with multi-head-attention mechanism in traditional Transformer Graph Encoder (TGE) model. STr-GCN has demonstrated its capacity to extract powerful spatial-temporal information for 3D hand gesture recognition. STA-GCN [16] is the most notable related work that used spatial and temporal self-attention modules to learn trainable adjacency matrices. DG-STA [9] proposed to leverage the spatial and temporal attention mechanism to construct dynamic spatial-temporal graphs by automatically learning the node features and edges from the hand skeleton graph. The proposed Graph Convolutional Skeleton Transformer (GCST) [20] for human action recognition consists of three-stage architecture hierarchically with residual connections to extract high-dimensional features from the input skeleton sequences.

3 EPT-MoE model architecture parameters

Table 3 shows the hyper-parameter of EPT-MoE model architecture that we trained in our experiments. We note P the total number of p Parallel Transformers, L is the number of encoder layers, H is the number of Heads, M is the number of experts in MoE layers, K the number of experts per Token in MoE layer, and O is the total number of MoE layers in all blocks of the EPT-MoE model.

Table 3: EPT-MoE model architecture parameters.

for p Parallel Transformer encoder of the EPT-MoE model architecture with $1 \leq p \leq P$			
GCN	Parallel Transformer Block	MoE Layers	SFFN and Linear Layers
3*Conv2D BatchNorm2D Mish() activation function	The number of encoder layers (L_p) in the *p transformer encoder* L_p=1, 2, 3, 4, 6, 8 The number of heads H = 6, 8 Model dimension $d_{model} = 128$ D_q, D_k, and $D_v = 32$ Dimension for linear feature space (Transformer hidden dimension) = 256 Sequence length T = 8	O: the total number of MoE layer I the whole model The total number of experts : M=2, 4, 8, 16 The number of experts per one input of the sequence : K=1, 2, 3, 4 *Activation function GELU for Gating network (G) Softmax (TopK(.))* Expert network: a feed-forward (FFN) with the model dimension fix to $d_{model} = 128$ and MoE hidden dimension of FFN set to 512.	linear layers with a fixed number of a two-layer feed-forward with the model dimension $d_{model} = 128$ and MoE hidden dimension of FFN set to 512

3.1 Spatial Graph Convolutional Network (GCN) of EPT-MoE

Suppose the raw skeleton sequences of hand joints are denoted as $X_{in} \in R^{C_{in} \times T \times N}$, where each vector $X_{in}^t = \{x_1^t, x_2^t, \dots, x_n^t\}$ of the sequence X represents the 3D coordinates of hand joints at a time stamp t, with *T is the sequence length of skeleton sequences for N hand joints*. *N is the number of nodes (hand joints).* The basic spatial graph convolution operation at the *l*th layer is formulated as follows:

$$X^{l+1} = \sum_k^{K_v} W_k (X^l A_k) \qquad (1)$$

With the partition strategy applied in STr-GCN [11] and ST-GCN [14]. Where K_v is set to 3 (the kernel size of the spatial dimension given by 3D coordinates of hand joints). W_k is a trainable weight matrix and its shared between all graph to capture spatial features. Each edge of the adjacency matrix $\overline{A}_k \in \{0,1\}^{N \times N}$ has two values o or one, and the adjacency matrix represents the inter hand-joints connections. The final adjacency matrix A_k is the normalization of the adjacency matrices (for more information, STr-GCN [11] and ST-GCN [14]).

$$A_k = D_k^{\frac{-1}{2}} (\overline{A}_k + I) D_k^{\frac{-1}{2}} \qquad (2)$$

Here, introducing the self-loop constant matrix I considers the influence of the node itself and using the same partitioning strategy in STr-GCN [11] to the 3D hand skeleton graph. In the proposed EPT-MoE architecture, GCN consists of three parallel convolution operation for each dimension of K_v (set to 3 is the kernel size of the spatial dimension given by 3D coordinates of hand joints) applied on adjacency matric (A) of size *N*N=22*22 (the number of nodes (hand joints)*) and *the* X_{in} *feature size* $C_{in} \times T \times N = 3 * 8 * 22$ [11]. GCN consists of one block of three layers of two-dimension Convolutions (Conv2D with kernel size set to 1 with stride set to 1, and number of filters set to 128) followed by applying batch normalization *(BatchNorm2D)* and the *Mish() activation function* through the sum of three-convolution output (see Fig.2). We note the GCN output by X_S which is connected directly to the input of the spatial SFFN of the EPT-MoE Transformer.

3.2 Spatial Feed Forward Neural Networks (SFFN)

SFFN consists of three principal layers: First Linear Layer, MoE layer and the Second Linear Layer. In the first Linear Layer[1], we apply a linear transformation to the incoming data from the GCN output X_S. Linear layer consists of two linear layers with input dimension set to the model dimension to $d_{model} = 128$ and the hidden dimension set to 512. Then, MoE layer selects the spatial features the most representative by *K experts of the total number M of experts* (refers to Section 3.4). In the second Linear Layer, we apply the same linear transformation using in the first one to the incoming data from the MoE layer output. We note X_P the SFFN output. X_P is the input for all P Parallel Transformer encoders with MoE layers. In the next sections 3.3 and 3.4, we present the Transformer block and MoE layer of the proposed model EPT-MoE.

3.3 Notes about Standard Transformer and EPT-MoE model architecture:

Generally, in the second layer of Transformer [8], [11], the fully connected layer (FC) consists of one or two-layer pointwise feedforward network (FFN). Transformer employs a residual connection around each module, followed by Layer Normalization. The final output of the entire process of the standard Transformer can be formulated as a the output of the FFN block:

$$Output_{standard\_transformer} = FFN(Y_p) = FFN(LayerNorm(MHSA(X_P) + X_P) \qquad (3)$$

$$\text{Where : } Y_p = LayerNorm(MHSA(X_P) + X_P) \qquad (4)$$

Where X_P is the SFFN output (seen as a sequence), which connected to the input of the first self-attention layer of MHSA. We note Y_p the output of the normalization layer of the Transformer block in Parallel Transformer encoder p.

In EPT-MoE, we replace this FFN layer by block of MoE layer in all in p Parallel Transformer encoder.

[1] Linear Layer PyTorch: https://pytorch.org/docs/stable/generated/torch.nn.Linear.html

3.4 Notes about MoE layer of EPT-MoE model architecture improvements:

We use the GeLU non-linearity in each MoE layer. To ensure similar compute FLOPs to GeLU model, EPT-MoE uses MoE layer with *args.hidden_dim* = d_{ffn} =$4d_{model}$ to account for the additional projection. Each expert network is a feed-forward (FFN) with the model dimension fixed to args. dim $= d_{model} = 128$ and MoE hidden dimension of FFN set to d_{ffn} =512.

A class presents each FFN layer of the MoE layer in EPT-MoE Model architecture as follows:

```
# FeedForward as Mixture-of-Expert (FFN1, FFN2, FFNm, ... FFNM) of EPT-MoE
class FeedForward(nn.Module):
    def __init__(self, args: ModelArgs):
        super().__init__()
        self.w1 = nn.Linear(args.dim, args.hidden_dim, bias=False)
        self.w2 = nn.Linear(args.hidden_dim, args.dim, bias=False)
        self.w3 = nn.Linear(args.dim, args.hidden_dim, bias=False)
    def forward(self, x) -> torch.Tensor:
        return self.w2(nn.functional.gelu(self.w1(x)) * self.w3(x))
```

Remarks:
In EPT-MoE, we suppose having a sequence (from t_1, …, t_i,…t_n) of 3D hand skeletons similarly to STr-GCN [11] and ST-GCN [14]. In addition, we use an adjacency matrix to construct a graph sequence. First, in the spatial domain, we use one layer of GCN to extract spatial hand features at each frame taking advantage of the natural graph structure of the hand skeleton. Then, in a temporal domain and respecting the graph structure of the spatial features, we introduce a multiple layers of Mixture-of-Experts with parallel transformer encoders to extract inter-frame relevant features. Finally, we introduce a global pooling operation to aggregate the graph into a representation to interpret it by the final linear layer.

4 Experimental Results of EPT-MoE

4.1 EPT-MoE Model Variations, Ablation Study and Results Analysis

We conduct extensive experiments for scaling studies and evaluation performance overall several variants of EPT-MoE model family to show the advantages of combining parallel transformer encoders with multiple layers of Mixture-of-Experts into one model transformer for 3D hand gesture recognition. EPT-MoE are coded using PyTorch and executed on GPU Model NVIDIA (Tesla V100-SXM2-16GB) based on Google Colab and Weight & Bias (wandb).

In Table. 4, we present 17 variant models of EPT-MoE with their architectures, parameters size, GPU Model performance and recognition accuracy. We explore how several variants of EPT-MoE transformer model and datasets can affect the EPT-MoE performance in order to understand the behavior of MoE models on the same training data (herein SHREC'17 datasets). All results in this study are carried out on the SHREC'17 dataset (see Table 2 and Table 4).

In contrast, EPT-MoE Models 6, 10 and 16 are able to achieve a recognition accuracy of 89.97%, 87.95%, and 91.05% with model size, respectively: 89.622MB, 90.083MB, 95.064MB. We conduct our experiments using PyTorch on GPU NVIDIA (Tesla V100-SXM2-16GB), Google Colab, Weight & Bias (https://wandb.ai: AI Platform).

Table 4: Architectures and sizes for 17 variants of EPT-MoE model, Recognition accuracy (%) on test datasets for 17 model variations of the EPT-MoE architecture trained on SHREC'17 (14 Gestures) using GPU Model Performance (Tesla V100-SXM2-16GB).

Name of Project RUNS on wandb.ai	EPT-MoE Model N°	EPT-MoE Transformers with MoE Layers						Model params size (MB)	Recognition test accuracy (%)	Mean of GPU Power usage (Watt)	GPU Memory Allocated (%)	GPU Power Usage (%)
		Parallel Transformer			MoE Layers			$n_{parameters}$				
		p	L_p	H	M	K	O					
festive-festival-100	Model 1	1	L_1=4	8	16	2	2	58.977	91.34	47.96	10.57	15.96
beaming-festival-116	**Model 2**	**1**	**L_1=4**	**8**	**8**	**3**	**3**	**40.974**	**92.18**	**46.23**	**16.33**	**15.36**
spring-energy-141	**Model 3**	**4**	**L_1=1** **L_2=2** **L_3=3** **L_4=4**	**8**	**8**	**2**	**5**	**76.430**	**92.63**	**46.15**	**15.55**	**15.43**
toasty-field-8	**Model 4**	**3**	**L_1=1** **L_2=2** **L_3=3**	**8**	**8**	**2**	**3**	**42.105**	**92.10**	**47.20**	**11.68**	**15.73**
wise-wave-11	Model 5	2	L_1=2 L_2=4	8	8	2	2	42.038	90.84	47.64	11.03	15.88
twilight-vortex-9	Model 6	3	L_1=3 L_2=4 L_3=5	8	8	2	4	89.622	89.79	48.04	17.33	15.98
hopeful-star-5	Model 7	2 + Res	L_1=2 L_2=4	8	8	3	3	54.629	91.21	46.25	17.88	15.42
fast-energy-7	Model 8	2	L_1=2 L_2=3	6	8	2	4	47.143	90.89	47.34	10.74	15.81
restful-music-6	Model 9	3	L_1=1 L_2=2 L_3=4	8	8	2	4	55.228	90.54	45.99	13.41	15.3
breezy-bee-10	Model 10	3	L_1=3 L_2=4 L_3=5	8	8	3	4	90.083	87.95	48.65	17.38	16.21
glowing-horse-135	Model 11	2 + Res	L_1=6 L_2=6	8	4	2	3	47.848	91.83	51.18	15.84	17.12
chromatic-fuse-125	Model 12	1	L_1=4	8	8	3	2	34.679	91.53	49.49	10.28	15.91
alight-firecracker-103	Model 13	1	L_1=4	8	8	3	3	40.447	91.42	44.22	11.05	15.23
silver-capybara-138	Model 14	4	L_1=1 L_2=1 L_3=1 L_4=1	8	8	3	6	41.175	91.49	45.80	10.92	15.62
dainty-star-139	Model 15	4	L_1=1 L_2=1 L_3=1 L_4=1	8	8	2	5	35.534	91.26	44.46	9.89	14.83
vivid-peony-132	Model 16	2	L_1=6 L_2=6	8	8	2	3	95.064	91.05	45.88	17.39	14.58
alight-dragon-129	Model 17	1	L_1=4	8	8	Mix 2 & 3	3	40.974	90.97	49.03	15.81	16.37

EPT-MoE Parallelizes Better Than Standard Transformer and MoE Models

Scaling Study of EPT-MoE models:

Generally, in MoE models sparsely activated in that only a small part of the total parameters can be activated for each prediction. Therefore, MoE models can scale by growing the size or number of experts in the MoE layer [1], [4], [29]. To scale up language models without making models deeper by adding more layers, another possible way is to propose parallel transformer encoders with smaller number of layers and coupled with MoE layers. This process can increase or decrease the total number of parameters of EPT-MoE model. It depends on the parameters presented in Table.4 and Table.3 : P the total number of p Parallel Transformers, L is the number of encoder layers, M is the number of experts in MoE layers, K the number of experts per Token in MoE layer, O is the total number of MoE layers in all blocks of the EPT-MoE model, and H is the number of Heads. In Fig.3 and Fig.4, For each prediction on a given input sample, we observe EPT-MoE models perform similarly at smaller scales with a high performance of recognition accuracy that make training more stable but MoE models may be outperform recognition accuracy at larger scales. We also show experiments with scaling the number of p Parallel Transformers and the number of experts (see Model 3 in Table 4) where we observe that, for a fixed budget of computation per prediction, adding more p parallel Transformer encoders with scaling the number of encoder layers and the number of MoE layers leads to better predictive performance.

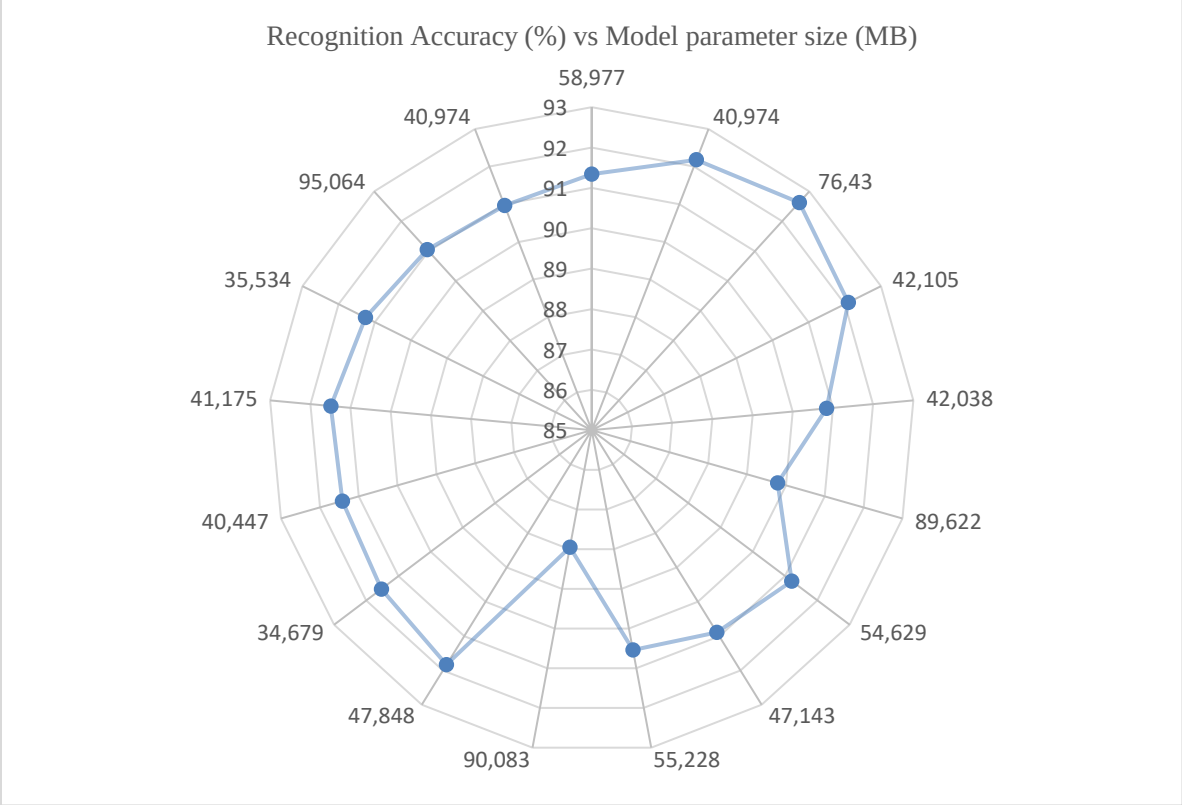

Fig.3. Comparison of Performance of Recognition Accuracy vs. Model parameters size for 17 variants of EPT-MoE model

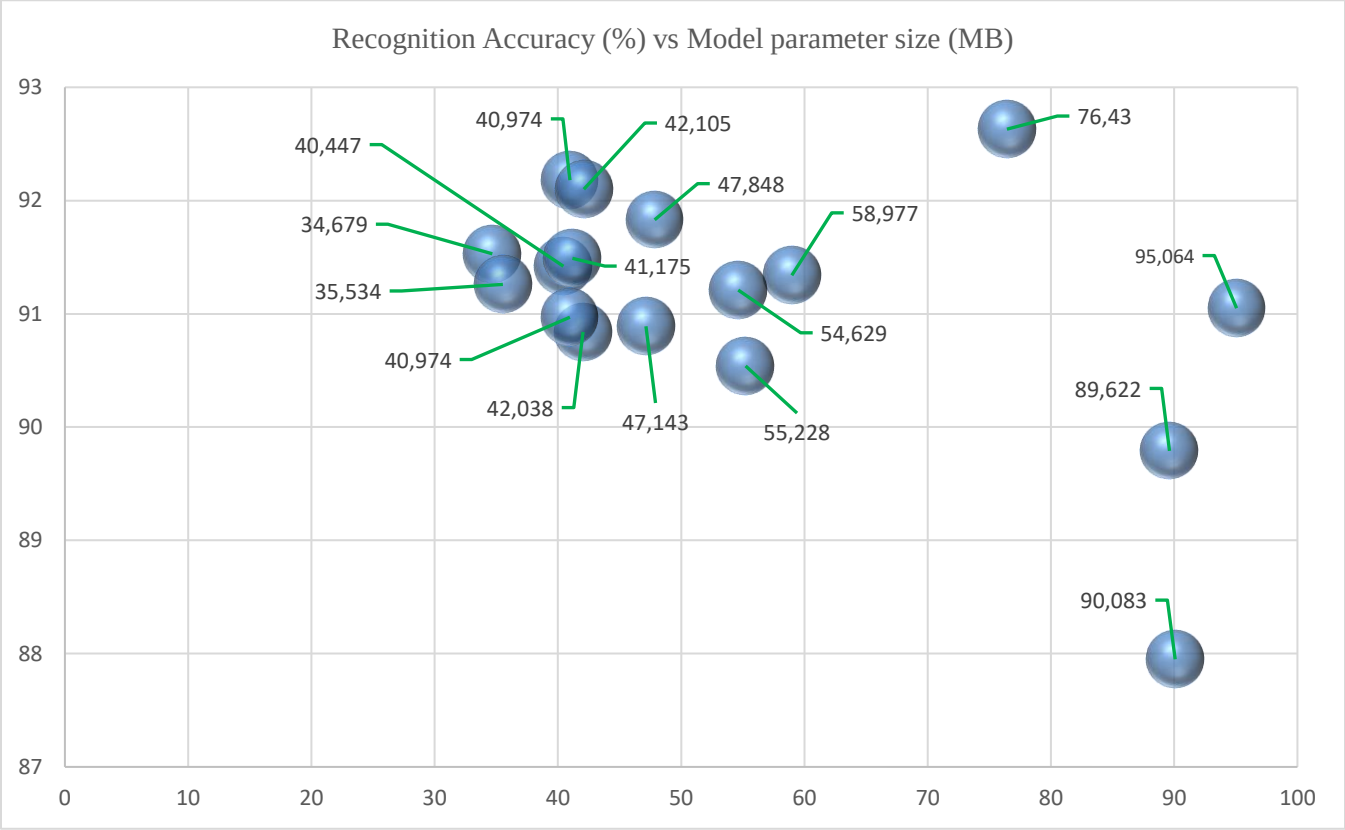

Fig.4. Comparison of Performance of Recognition Accuracy vs. Model parameters size for 17 variants of EPT-MoE model

Training Efficiency of EPT-MoE models:

We investigate the training loss stability and compute efficiency of the proposed EPT-MoE models. It is common for LLMs to encounter loss instability, which can lead to loss divergence and require careful manual interventions to recover training. Fig. 5 and Fig.6 show EPT-MoE models training progressed with excellent loss stability and accelerating training.

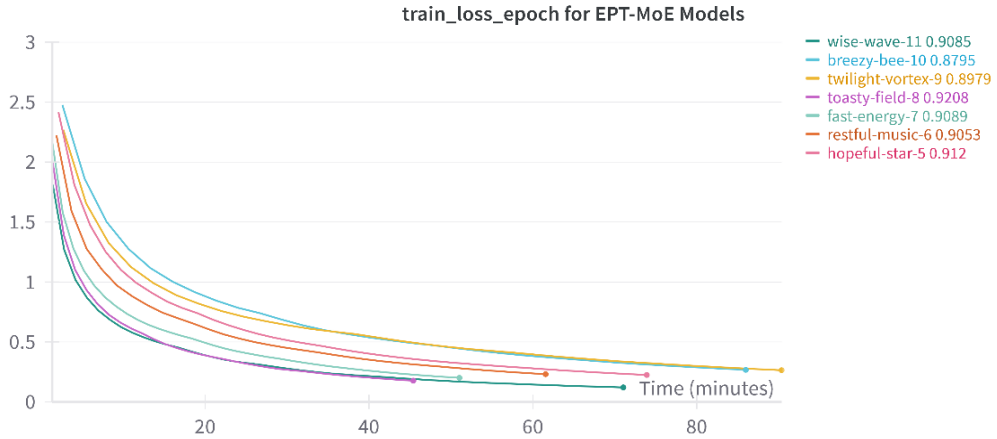

Fig.5. EPT-MoE Parallel models achieve better performance faster: Accelerating Training of 7 variants of EPT-MoE model (See Appendix: Reports generated using Wandb).

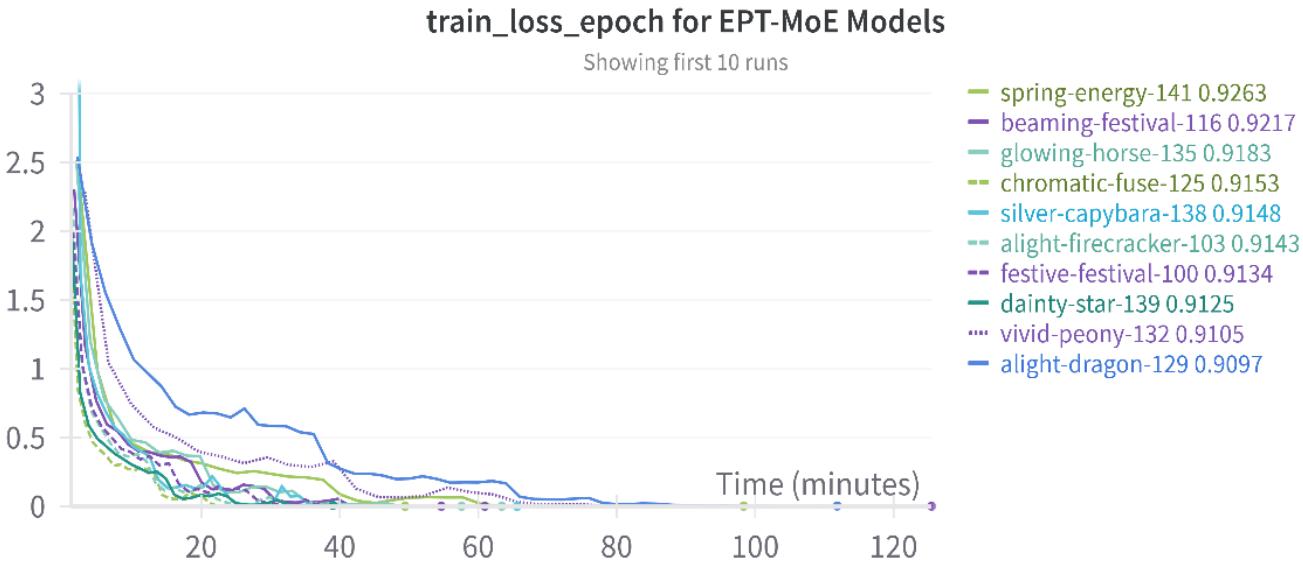

Fig.6. EPT-MoE Parallel models achieve better performance faster: Accelerating Training of 10 variants of EPT-MoE model (See Appendix: Reports generated using Wandb).

We observe parallel models (Model 3: *springer-energy-141*, Model 4: *toasty-field-8*) achieve better performance faster than parallel deeper models (Model 16: *vivid-peony-132*, Model 6: *twilight-vortex-2*, and Model 10: *breezy-bee-10*).

Fig.7 and Fig. 8 show EPT-MoE models with excellent stability of recognition accuracy performance for Training and inference time.

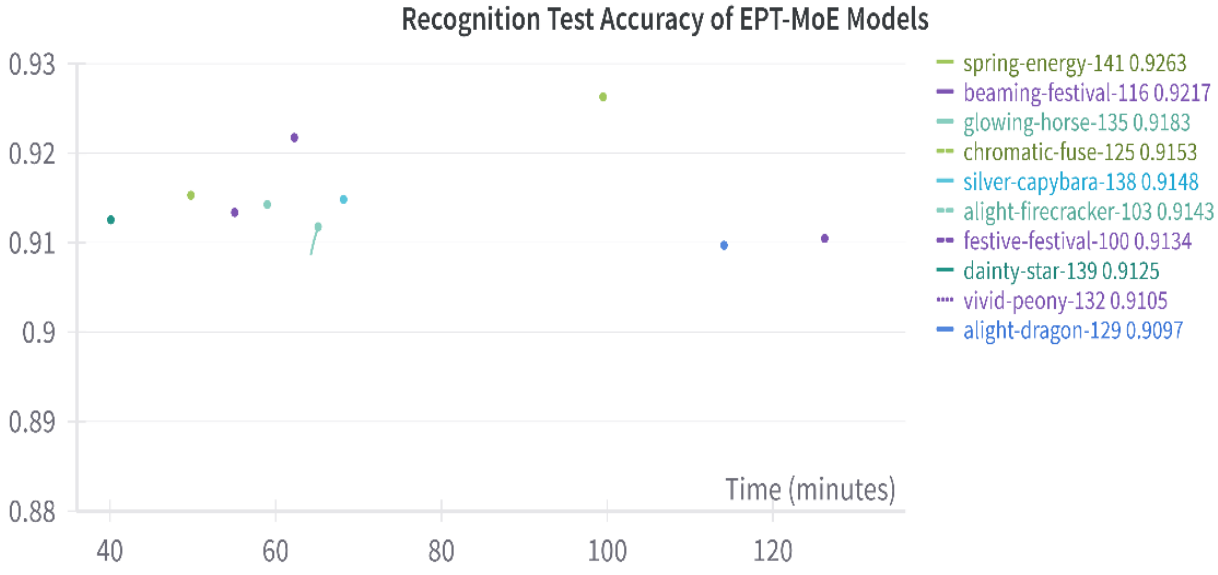

Fig.7. Comparison of Performance of Recognition Accuracy vs. Model Time Inference for 10 variants of EPT-MoE model

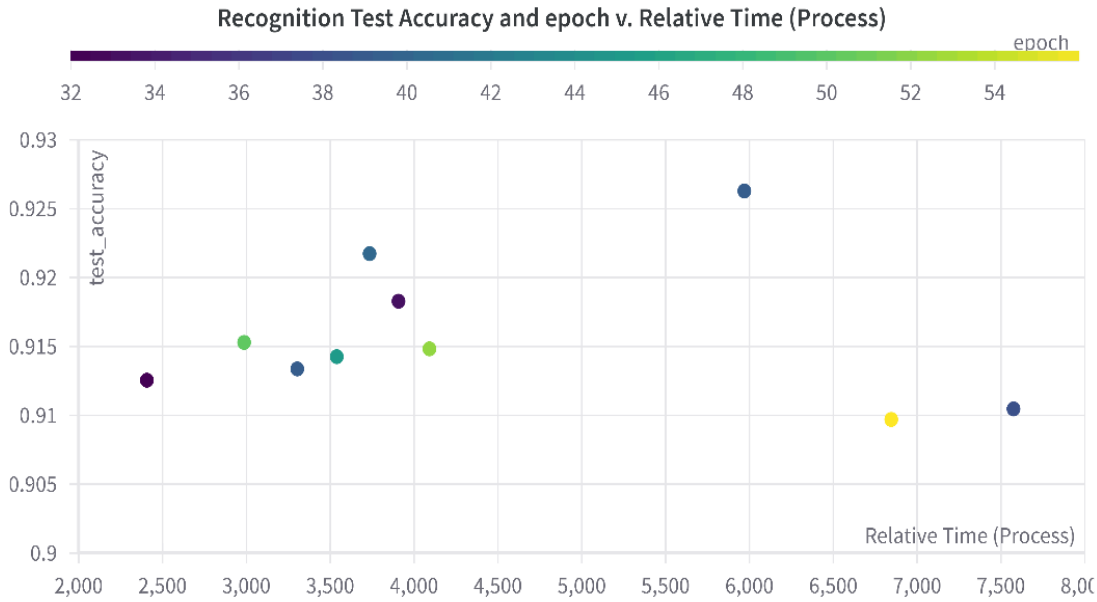

Fig.8. Comparison of Performance of Recognition Accuracy vs. Model Relative Time Process for 10 variants of EPT-MoE model

Computation efficiency / Memory Allocation using GPU model Performance:

Plotting GPU usage across several variants of EPT-MoE model explains what's happening. Fig.9 and Fig 10 show the Memory Allocation (in Bytes) of GPU utilization percentage using GPU Model (Tesla V100-SXM2-16GB) and the Training epoch average time for 10 variants of EPT-MoE model. For the best high performance of recognition accuracy, we observe the best model is Model 3 with memory allocation percentage about 15.55%.

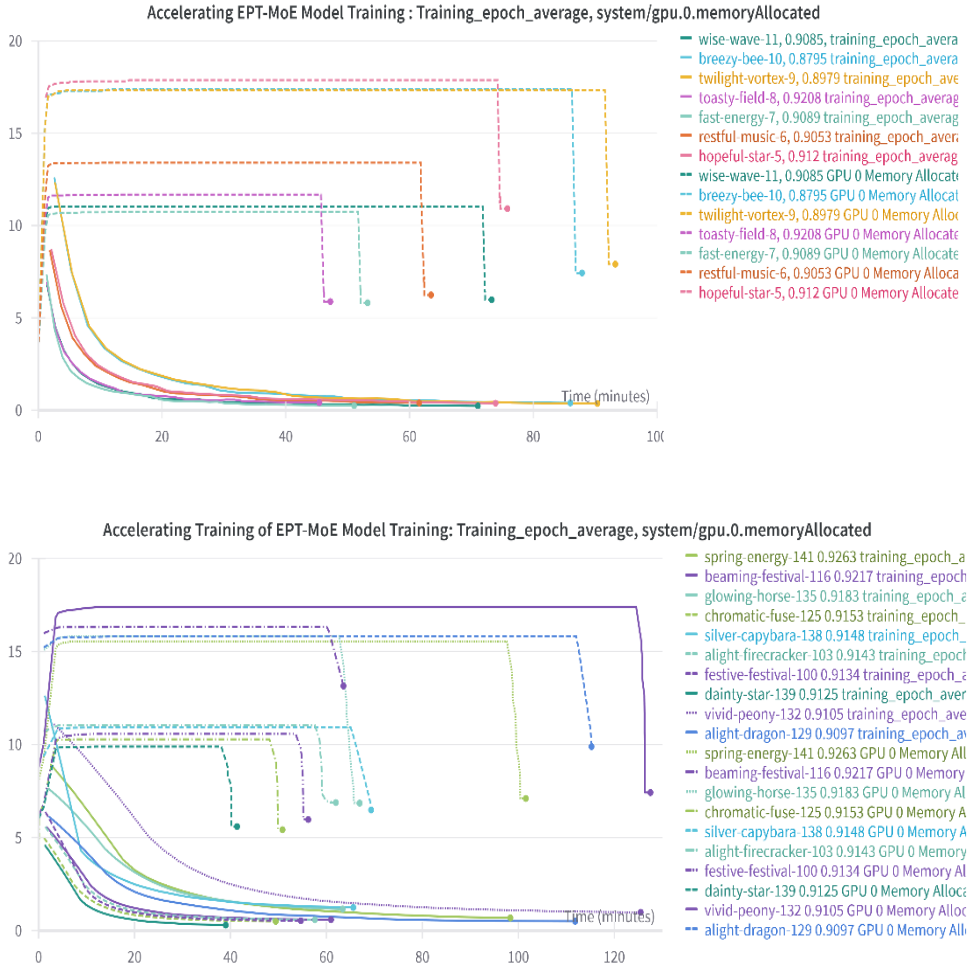

Fig.9. Accelerating Training and GPU Model Performance (GPU Memory Allocation utilization percentage) of 17 variants of EPT-MoE model (See Appendix: Reports generated using Wandb).

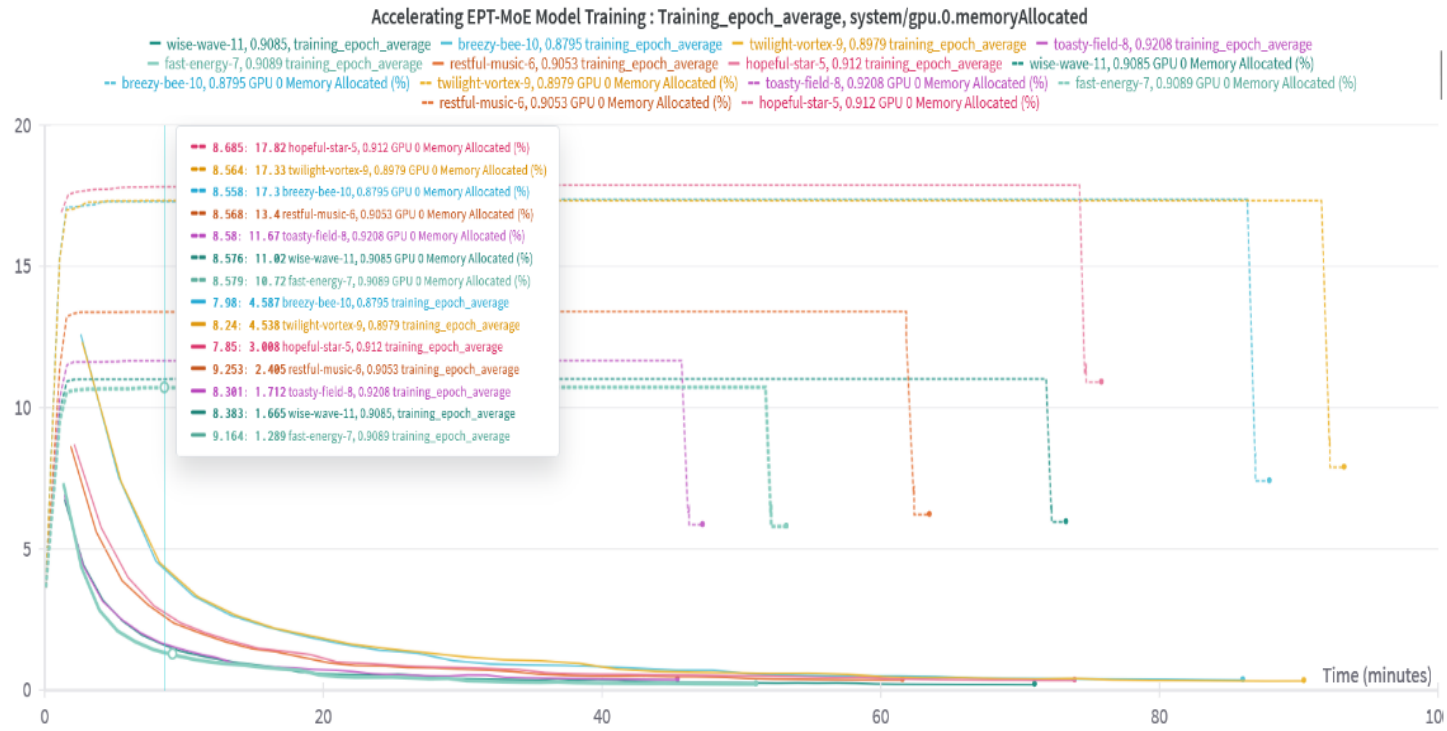

Fig.10. Accelerating Training and GPU Memory Allocation utilization percentage of 10 variants of EPT-MoE model (See Appendix: Reports generated using Wandb).

Energy usage and consumption using GPU model Performance:

Plotting GPU usage across several variants of EPT-MoE model explains what is happening in terms of energy/power usage and consumption. Fig.11 show the energy usage and consumption (in Watt) of GPU utilization using GPU Model NVIDIA (Tesla V100-SXM2-16GB) and the Training epoch average time for 17 variants of EPT-MoE model. For the best high performance of recognition accuracy, we observe the best model is Model 3 with energy/power usage about 46.15 Watt.

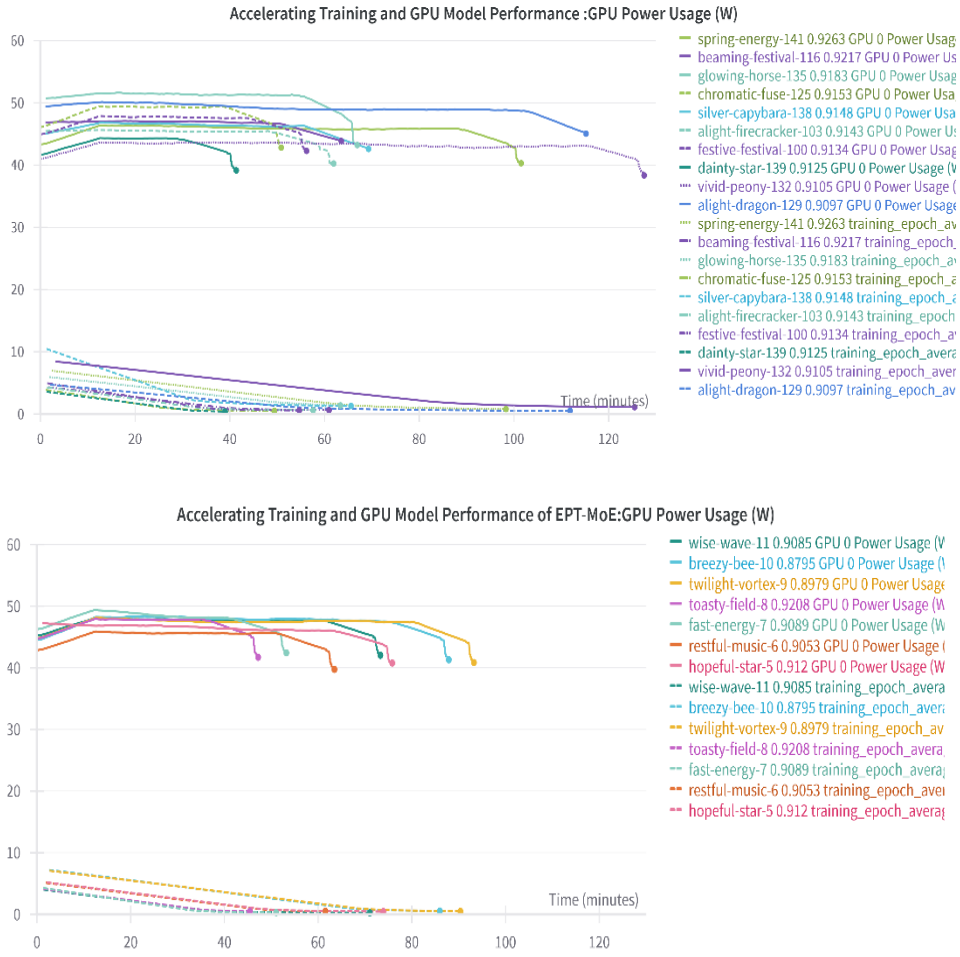

Fig.11. Accelerating Training and GPU Model Performance (GPU Power usage in Watt) of 17 variants of EPT-MoE model (See Appendix: Reports generated using Wandb).