Beyond ViT: Accelerating ASL Recognition with Convolution-Attention Fusion

Jingshu Liu
Stanford University
jingshu7@stanford.edu

Hengjin Tan
Stanford University
hengjint@stanford.edu

Kai Liu
Stanford University
xkliux@stanford.edu

Abstract

Despite recent advances in vision transformers, their computational demands limit deployment in real-time accessibility applications. We present a systematic study of American Sign-Language (ASL) alphabet and digit recognition using three modern vision backbones—CoAtNet, vanilla Vision Transformer (ViT-Base), and the gesture-tuned HGR-ViT—across two scales of data. After introducing the ASL recognition task and related work, we describe our experimental setup: a 2,520-image pilot corpus and the $104k$*-image DeepASLR benchmark, uniform training protocols, and metrics that couple accuracy with computational cost. We then detail the architectures and their theoretical FLOPs/parameter footprints before reporting results. On the small ASL dataset, CoAtNet emerges as the clear "accuracy-per-compute" winner, achieving* 99.2% *validation accuracy, on the small ASL dataset with only* 8.95 *GFLOPs /* $26.7M$ *parameters, and reaching perfect accuracy on DeepASLR in 3h 15m—representing a 3× speed-up versus both Transformer baselines, which require approximately 9 hours and use* $3.8\times$ *more FLOPs and* $3.2\times$ *more parameters. Error analysis pinpoints a handful of confusable letter pairs, motivating targeted augmentation and specialist heads. Finally, we outline future work that couples efficiency-oriented compression with self-supervised pretraining on unlabeled ASL video, aiming to retain near-ceiling accuracy while driving latency and energy consumption toward real-time edge deployment, potentially improving accessibility for deaf and hard-of-hearing communities.*

1. Introduction

With over 70 million deaf individuals worldwide facing persistent communication barriers, automatic sign-language recognition (SLR) offers a practical route to narrowing that gap. Early deep-learning efforts relied on recurrent neural networks (RNNs) for temporal dynamics [11, 12, 14], while convolutional neural networks (CNNs) provided strong spatial encoders [17, 10, 20, 21]. More recent work introduced spatial-temporal models such as TSTM and Graph Convolutional Networks (GCNs) that explicitly couple motion and appearance [8, 13]; comprehensive surveys chart these trends in depth [2].

The breakthrough Transformer architecture [22] has since proved effective in vision tasks [7] and, specifically, American Sign-Language (ASL) recognition [19]. Yet pure Transformers can struggle with data efficiency and fine-grained local detail, prompting hybrid designs that fuse CNN backbones with attention layers [4]. Parallel to these advances, graph-based methods model the signer's skeleton as a temporal-spatial graph, allowing the network to learn joint relationships more naturally [15].

Building on these ideas, our project integrates CNN features and Transformer attention within a single framework (CoAtNet)—the first systematic evaluation of CNN-attention hybrid models for static ASL recognition. The goal is to retain the inductive biases and computational efficiency of convolutions while benefiting from the global receptive field of self-attention.

On the 2,520-image low-resource corpus, our fine-tuned CoAtNet reached 99.3% test accuracy while consuming only 8.95 GFLOPs and 26.7 million parameters. Both the vanilla Vision Transformer and the gesture-tuned HGR-ViT lagged behind at 98.3% and 98.7% respectively, despite requiring roughly $2.1\times$ the computational budget (33.74 vs 8.95 GFLOPs).

Scaling up to the 104k-image DeepASLR benchmark, all three architectures eventually attained perfect training and validation accuracy. Even so, CoAtNet converged in just 3 hours 15 minutes, whereas the two Transformer baselines needed close to 9 hours. The hybrid model also reduced the cumulative floating-point workload to about 4.5 billion operations, compared with $12-17$ billion for its competitors.

Taken together, the results point to a consistent three-fold advantage for CoAtNet in both wall-clock training time and compute-to-accuracy efficiency, making it the most practical choice across datasets and scales. These efficiency gains are particularly crucial as ASL recognition systems transition from laboratory settings to real-world assistive technologies.

In the remainder of this report, we detail the related literature (Section 2), the experimental methodology (Section 3), datasets (Section 4), an extended analysis of errors and runtime characteristics (Section 5), and directions for future work (Section 6).

2. Related Work

Before presenting our integrated approach in detail, we review the key developments in deep learning for sign-language recognition that inform our work, highlighting the strengths and limitations of each architectural family.

2.1. CNN, RNN, LSTM

Convolutional neural networks (CNNs) have become the workhorse architecture for sign-language recognition, consistently delivering high accuracy on static and isolated gestures yet still facing robustness gaps in continuous, real-world settings.

Early research by [17] proposed a Kinect-based system that ran two parallel CNNs—one targeting hand regions and the other upper-body silhouettes—to recognize 20 Italian Sign-Language gestures. The model achieved 91.7% cross-validation accuracy and generalized well to unseen signers and environments. Subsequent work focused on static or isolated gestures.

[21] showed that a deep CNN could dispense with explicit hand segmentation and handcrafted features, reaching 94.7% accuracy on the NUS Hand Posture dataset and 99.96% on the American Fingerspelling A dataset, despite minimal inter-class variation. In [10], the authors created a 104K image ASL alphabet dataset captured at multiple distances and lighting conditions, then trained a three-layer CNN that attains $\sim$99.4% accuracy on two public benchmarks and 99.38% on their own, more challenging set, and developed a real-time OpenCV interface for robust letter-level ASL recognition.

A survey by [20] synthesized these CNN-based advances, outlining the common pipeline—data capture, hand-region preprocessing, CNN-driven feature learning, and classification. Reported signer-dependent accuracies ranged from 69.0% to 98.0% (mean $\sim$88.8%), with the best static-sign systems exceeding 99%. The authors argued that further progress would require larger, less-constrained datasets to improve robustness in real-world settings.

While CNNs excel at spatial feature extraction and achieve high accuracy on static gestures, they struggle with temporal dependencies and global context modeling—limitations that motivate the exploration of sequential and attention-based approaches.

While CNNs established strong baselines for spatial feature learning, the sequential nature of sign-language motivated researchers to explore temporal modeling approaches, leading to the adoption of recurrent architectures. Recurrent neural networks—especially long short-term memory (LSTM) variants—have become an important mechanism for modeling the temporal evolution of gestures once spatial features (e.g., hand shapes) have been extracted by CNNs or other encoders. Across a growing body of work, they consistently boost recognition accuracy on isolated words and make sentence-level sign-language translation feasible.

The study [11] built a Leap-Motion-based pipeline in which 30 hand-motion features were passed to an LSTM-RNN whose hidden states were finally classified with a k-NN layer; the system correctly identified all 26 ASL letters with 99.4% accuracy (91.8% under 5-fold cross-validation), underlining how even low-cost depth sensors paired with LSTMs can achieve near-perfect alphabet recognition.

[14], the authors combined an Inception CNN with a one-layer RNN to recognize 46 Argentine Sign-Language (LSA) words in real-time; spatial features from successive video frames feed the RNN, yielding 95.2% top-1 accuracy and demonstrating that a shallow recurrent head is enough when visual embeddings are strong.

In [12], the authors took a different route, replacing CNN features with handcrafted motion descriptors but using a deeper LSTM stack to capture long-range dependencies; their model surpassed traditional HMM and SVM pipelines on a 200-word Chinese-SL dataset, signaling the LSTM's advantage for sequential modeling even without deep spatial encoders. Pushing beyond isolated gestures,

In [9], the authors proposed a hierarchical encoder–decoder in which a 3D-CNN mines variable-length "viseme" clips, and three stacked LSTM layers first summarize those clips and then decode them into full sentences. The framework outperformed CTC and HMM hybrids on signer-independent tests with unseen sentences, showing that multi-level LSTMs with attention can handle the reordering and co-articulation inherent in continuous sign-language translation.

In summary, RNN/LSTM modules have matured into a "classical baseline" for temporal modeling in sign-language recognition: they lift isolated-word accuracies into the mid-90s or higher, and—when organized hierarchically—enable direct video-to-sentence translation. Remaining challenges include scaling to large, signer-independent vocabularies, integrating non-manual cues, and maintaining low latency; these issues motivate emerging hybrids that pair LSTMs with 3D-CNNs, graph networks, or Transformers as the next research frontier.

2.2. Transformers

Transformers have only recently migrated from language to vision, but the seminal Vision Transformer (ViT) paper [7] showed that simply treating an image as a sequence of fixed-size patches and feeding them to a vanilla encoder can rival—or surpass—deep CNNs once large-scale pre-training is available. This "patch-token" view removes the locality bias of convolutions and gives the model global context from the first layer, inspiring a wave of work that re-thinks sign-language pipelines around attention rather than kernels.

The first end-to-end application to continuous signing came with Sign-Language Transformers by [3]. A CNN front-end extracted video features, but all sequence modeling was handled by a Transformer encoder–decoder that was trained jointly for two objectives: (i) gloss-level recognition via a CTC loss and (ii) spoken-language translation via cross-entropy. On the RWTH-PHOENIX-14T benchmark, the model doubled the previous BLEU score for translation quality and set new state-of-the-art word-error rates.

Attention models have also pushed static hand-gesture tasks to ceiling performance. In [19], the HGR-ViT slices each hand image into 16×16 patches, adds learnable positional embeddings that explicitly encode orientation and location, and feeds the sequence through a pure ViT encoder followed by an MLP head. Despite its simplicity, the network reaches 99.98% on the ASL alphabet, 99.36% on ASL-Digits, and 99.85% on the NUS posture set—essentially saturating these benchmarks and confirming that Transformers can discover subtle inter-finger cues without handcrafted pre-processing.

One open question is how to retain the data-efficiency inductive biases of convolutions while benefiting from the global receptive field of attention—crucial for sign-language corpora that are still orders of magnitude smaller than ImageNet. CoAtNet [5] tackled exactly that by interleaving MBConv blocks with relative-attention Transformer layers; in image classification it matches or beats ViT while using dramatically less data and compute. Although not yet evaluated on SLR, its hybrid philosophy offers a promising blueprint for future signer-independent models that must generalize from limited footage.

As transformer architectures continue to evolve rapidly, recent developments have focused on making attention mechanisms more efficient for specialized applications like gesture recognition, reinforcing the importance of the hybrid approaches we investigate in this work.

In summary, the transformer era of SLR is unfolding along two tracks: (1) sequence-to-sequence models that drop recurrent stacks in favor of multi-head attention and jointly learn recognition plus translation, and (2) vision-transformer front-ends that treat each video frame—or even each tracked hand—as a bag of patches to be globally reasoned over. Early results already outperform long-standing CNN/RNN baselines, but challenges such as small ASL datasets, fine-grained non-manual cues, and real-time latency remain—motivating continued exploration of hybrid designs and spatial-temporal transformer variants.

Despite these promising results, Transformers face several challenges in SLR: (1) high computational cost limiting real-time deployment, (2) data hunger requiring large training corpora, and (3) lack of inductive biases for spatial locality that CNNs naturally provide.

While Transformers excel at global reasoning, graph-based approaches offer complementary benefits through their explicit modeling of anatomical structure, as the next section describes.

2.3. Graph-based Representation

Graph-based SLR treats the signer's body as a skeleton graph whose joints are nodes and bone connections are edges, letting neural networks reason directly over the spatial–temporal structure of a sign.

Adapted from action-recognition, the Spatial-Temporal Graph Convolutional Network (ST-GCN) in [6] encoded every video as a stack of skeleton graphs and applied shared graph kernels across both space and time. Without any RGB appearance cues, the model already exceeded earlier CNN/RNN baselines and came with an ASLLVD-Skeleton dataset (2,745 signs, $\sim$10k clips), demonstrating that joint-level dynamics alone are highly discriminative for isolated signs.

Later work in [16] focused on efficiency and scale. It introduced bottlenecked residual GCN blocks that reduce parameters (0.6 M vs 12 M in I3D) yet push pose-only Top-1 accuracy to 74% to 83% (+8.9 pp) on WLASL-100, 49% to 76% (+27.6 pp) on WLASL-300, and 29% to 56% (+27 pp) on WLASL-1000—while hitting 100% on the small LSA64 set. It also ran an order of magnitude faster (0.04 s vs 0.50 s per clip) thanks to lightweight graph convolutions.

Recognizing that most linguistic content sits in the fingers, [18] Hand-Aware GCN HA-GCN re-weights the skeleton to give hands higher resolution and adds a dedicated hand-topology attention branch before merging with the global body graph. The model reports state-of-the-art accuracies (e.g., 96 – 98% on AUTSL-226 and CSL-500) while using fewer parameters than multi-scale CNN–GCN hybrids, highlighting the value of fine-grained, anatomy-aware graph design.

Graph representations have evolved from vanilla ST-GCNs that simply mirror the kinematic tree to specialized, hand-centric or residual architectures that boost both robustness and speed. They excel on large-vocabulary, signer-independent benchmarks without the heavy compute of appearance-based 3D CNNs, but still rely on reliable pose extraction and struggle with subtle non-manual cues (facial expressions, mouthing). Current trends mix graph cues with transformers or multi-modal features, aiming to close that remaining gap while keeping the structural advantages of graph learning intact. These emerging hybrid directions highlight the potential of combining complementary architectural strengths.

Our review of the literature reveals several key research gaps. First, while CNNs excel at spatial feature extraction, RNNs at temporal modeling, and Transformers at global attention, few approaches have successfully integrated all three architectural paradigms. Second, graph-based methods demonstrate the power of structural priors but have rarely been combined with the representational capacity of Transformers. Third, the fragmentation of ASL datasets has limited cross-corpus evaluation, making it difficult to assess generalization across different signing conditions. Our work specifically addresses these gaps through a unified multi-architecture approach and harmonized benchmark creation, seeking to combine the strengths of each method while mitigating their individual weaknesses.

3. Methods

3.1. Fine-tuned CoAtNet for ASL data

While Transformers offer high model capacity, they often generalize less well than convolutional networks because they lack the inductive biases that favor vision tasks—an issue that is amplified on modest-sized datasets such as ours for ASL. To mitigate this, we fine-tune the CoAtNet architecture [5], which explicitly fuses convolution and self-attention, thereby combining the strengths of both families.

During block-level fusion, the depth-wise convolution and self-attention share an identical "weighted-sum" form. For a location i,

$$y_i = \sum_{j \in \mathcal{L}(i)} w_{i-j} x_j \text{ (depth-wise conv)}, \tag{1}$$

$$y_i = \sum_{j \in \mathcal{G}} \frac{\exp(x_i^T x_j)}{\sum_{k \in \mathcal{G}} \exp(x_i^T x_k)} x_j \text{ (self-attention)}. \tag{2}$$

Convolution provides translation equivariance because its weights depend only on the offset $i - j$. Self-attention, in turn, provides input-adaptive weights and a global receptive field. CoAtNet unifies all three properties by adding a scalar convolution kernel w_{i-j} to the attention logits:

$$y_i = \sum_{j \in \mathcal{G}} \frac{\exp(x_i^T x_j + w_{i-j})}{\sum_{k \in \mathcal{G}} \exp(x_i^T x_k + w_{i-k})} x_j. \tag{3}$$

This equation is a standard soft-max attention, so computational cost is unchanged, yet the $w_{i=j}$ term restores translation equivariance.

In the network-level stacking, CoAtNet downsamples aggressively and has the following layers.

- S0: two-layer convolutional stem
- S1: MBConv + SE
- S2-S4: either MBConv (C) or the hybrid Transformer block (T), with all C stages preceding any T stages.

The paper tests and compares five layouts:

C-C-C-C, C-C-C-T, C-C-T-T, C-T-T-T, VIT-REL

Balancing generalization, capacity, and efficiency, we adopt C-C-T-T as our backbone. All pre-trained weights up to the penultimate layer are retained, and the classifier is replaced by a fully-connected head with

$$\text{output dim} = \begin{cases} 26 & \text{ASL letters only,} \\ 36 & \text{letters + digits.} \end{cases} \tag{4}$$

Algorithm 1 is described below. We also show the ASL-adapted CoAtNet architecture in Figure 1.

Algorithm 1 Construction of the CoAtNet Fine-tuned Model

```
Require: num_classes, device
     Instantiate CoAtNet-BN0 backbone pre-trained on ImageNet-1k
  1: model ←
  2: CREATE_MODEL("CoAtNet_bn_0_rw_224.sw_in1k")
     Replace the default head by an average-pooled linear classifier
  3: model.head ←
  4: CLASSIFIERHEAD(in_features, num_classes, pool_type
     = "avg", drop_rate = 0.0)
     Move model to the specified device
  5: return model.TO(device)
```

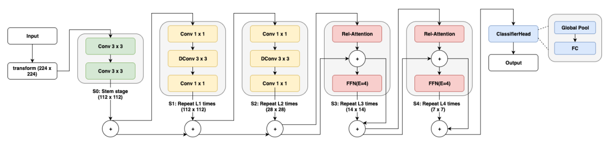


Figure 1: ASL Adapted CoAtNet Architecture

3.2. Baseline Models

We compare the above fine-tuned model on two baseline models.

The first one is the ViT model built in [7]. The ViT treats an image $x \in R^{H \times W \times C}$ as a $1 - D$ token sequence by cutting it into $N = \frac{H \times W}{P^2}$ non-overlapping $P \times P$ patches, flattening each to a vector of length P^2C and mapping these vectors to a common latent size D via a learned linear projection E. The resulting patch embeddings are prepended with a learnable [CLS] token and augmented by learned

$1-D$ positional embeddings before being fed to the encoder. The encoder is a stack of L identical Transformer blocks, each comprising pre-norm LayerNorm, multi-head self-attention, and a two-layer GELU-MLP, with residual connections after every sub-block; its output at the [CLS] position is layer-normed and passed to a classification head (a two-layer MLP during large-scale pre-training, a single linear layer for downstream fine-tuning). We display the architecture in Figure 2.

The second baseline model is the HGR-ViT model, which adapts the ViT transformer for hand-gesture recognition by retaining ViT-Base/32's backbone while tailoring the data pipeline and training schedule to small, gesture-centric datasets. Each RGB frame is first resized to 256×256 so it can be split into (32×32) patches—yielding $N = 64$ tokens—that are flattened and passed through a learned linear projection to a D-dimensional embedding (the authors keep ViT-Base's $D = 768$). A learnable [CLS] token is prepended and standard 1-D learned positional embeddings are added, after which the sequence traverses the 12-layer Transformer encoder (pre-norm LN $\rightarrow$ Multi-Head Self-Attention $\rightarrow$ LN $\rightarrow$ two-layer GELU-MLP, with residual connections). The normalized [CLS] vector emerging from the final layer feeds a single fully-connected head followed by soft-max, producing class probabilities for the gesture set.

As no model or code is available for the HGR-ViT model, we implement the model to replicate the logic mentioned in the paper.

Algorithm 2 Construction of the HGR-ViT Fine-Tuning Model

```
Require: num_classes, mlp_ratio ($\alpha$), dropout ($p$), device,
    pretrained_backbone
    Create ViT-Base/16 backbone without classification head
 1: model ← TIMM.CREATE_MODEL("vit_base_patch16_224",
    pretrained_backbone, num_classes, global_pool)
    Derive dimensions for the two-layer MLP head
 2: $d_{\text{in}} \leftarrow model.num\_features$          ▷ $d_{\text{in}} = 768$
 3: $d_{\text{hid}} \leftarrow \lfloor \alpha \times d_{\text{in}} \rfloor$
    Attach custom classification head
 4: model.head ← SEQUENTIAL(
 5:     LAYERNORM($d_{\text{in}}$),
 6:     LINEAR($d_{\text{in}}, d_{\text{hid}}$),
 7:     GELU(),
 8:     DROPOUT($p$),
 9:     LINEAR($d_{\text{hid}}$, num_classes) )
    Move to target device (mps, cuda, etc.)
10: return model.TO(device)
```

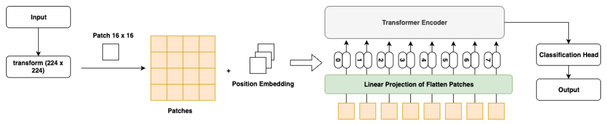


Figure 2: ASL Adapted ViT Architecture

3.3. Fine-tuning

For CoAtNet fine-tuning we start from the public ImageNet-21k checkpoint (C-C-T-T layout), replace the final classifier with a fresh linear head of 26 (letters-only) or 36 (letters + digits) outputs, and optimize the entire network end-to-end on our ASL corpus. The training set is split into train/validation/test folds; images are resized to 224×224, randomly augmented (crop + horizontal flip), and channel-wise normalized. An Optuna Bayesian search explores four hyper-parameters—learning-rate $10^{-5} - 5 \times 10^{-3}$, weight-decay $10^{-6} - 10^{-2}$, batch-size $\{16, 32, 48, 64\}$, and early-stopping patience (3–8 epochs). Each trial trains for at most 20 epochs with AdamW on a cross-entropy loss, a ReduceLROnPlateau scheduler (factor 0.2, patience 2), and class-balanced sampling; validation accuracy is the optimization target while early-stopping and Optuna's pruning API terminate unpromising runs early. Once the study converges, the best hyper-parameter set is fixed and the model is re-trained from scratch for the full epoch budget, after which the weights achieving the highest validation accuracy are saved and finally evaluated on the held-out test fold, providing the checkpoint used in all downstream ASL experiments.

Beyond early stopping, we implement multiple regularization strategies to prevent overfitting on our modest-sized dataset. These include L2 weight decay (optimized via Optuna) and dropout regularization in the classification heads, horizontal flipping, and color jittering to increase training diversity. The ReduceLROnPlateau scheduler with factor 0.2 and patience 2 provides adaptive learning rate reduction when validation performance plateaus. Additionally, we employ class-balanced sampling to ensure equal representation of all ASL letters during training, preventing the model from developing biases toward more frequent classes in any given batch.

To enable a fair comparison, we perform the same fine-tuning processes for the pre-trained ViT model and our built HGR-ViT algorithms.

All experiments were conducted on a MacBook Pro M3 with 8-core CPU, 10-core GPU, and 16GB unified memory, utilizing Metal Performance Shaders (MPS) for GPU acceleration through PyTorch 2.7.0. To ensure reproducibility, we fixed all random seeds, enabled deterministic operations, and version-pinned all dependencies in our requirements file. Training times averaged 3h 15m for CoAtNet and approximately 9 hours for ViT variants on this hardware configuration. Hyper-parameter configurations, and training scripts are made available with detailed documentation to facilitate reproduction of our results.

All experimental code, trained model checkpoints, and preprocessing pipelines are publicly available at CS231N-ASL-Deep-Learning.

4. Dataset

In this paper, we explore three complementary ASL datasets, illustrated in Figure 3, which together form the foundation of our unified benchmark.

4.1. Data Description

ASL Alphabet: The ASL Alphabet corpus comprises 1,820 color images (RGB, 200 × 200 px) depicting 26 static gestures: the 26 letters A–Z. Each class contains exactly 70 samples captured with consumer cameras under varied illumination and background conditions. The hands belong to multiple volunteers and vary in skin tone, scale, and orientation. As shown in Figure 3, these images provide excellent diversity in hand positioning. The dataset is organized into 26 folder-based labels for training, with a separate validation set (training ratio: 0.64, validation ratio: 0.16 and test ratio: 0.2) for hold-out testing. Released on Kaggle [1] under a permissive license, this dataset is particularly well-suited for benchmarking static gesture classifiers.

ASL Digits: The ASL Digits Dataset contains 700 raw RGB images representing signs for digits 0-9 (70 files per digit) and their corresponding MediaPipe-processed versions. This dataset is already formatted with extracted hand keypoints, making it particularly valuable for our skeleton-based approach. An example is shown in Figure 3.

DeepASLR: The DeepASLR dataset [10] contains 104,000 RGB images of ASL letters (4,000 per letter) captured under systematically varied conditions. Unlike other datasets with fixed camera distances (typically 0.5m, 0.75m, or 1m), DeepASLR includes images at all three distances. Additionally, it was collected at different times of day to incorporate diverse lighting conditions. A notable feature of this dataset is its handling of dynamic letters like Z and J, which typically require movement; DeepASLR represents these with static gestures at the movement's start or end position, allowing for a complete alphabet within a static-gesture recognition framework. An example is shown in Figure 4.

4.2. Data Preprocessing

Our preprocessing pipeline consists of two main stages.

Data Aggregation and Corpus Harmonization: We combined the the first two ASL datasets described above into a unified corpus of 2,520 images across 36 semantic classes. This process involved mapping folder names to a canonical label set (A–Z, 0–9) and generating a master manifest with absolute path, class label, and dataset origin for each image. This gives us a collection of small data, which we call the "small ASL dataset". We also train models on the DeepASLR dataset, which we call the "large DeepASLR dataset". Using stratified sampling, we created disjoint train/validation/test partitions (64/16/20 ratio) while preserving per-class frequencies within each source corpus. This approach yields a balanced, signer-agnostic benchmark that better represents real-world deployment scenarios.

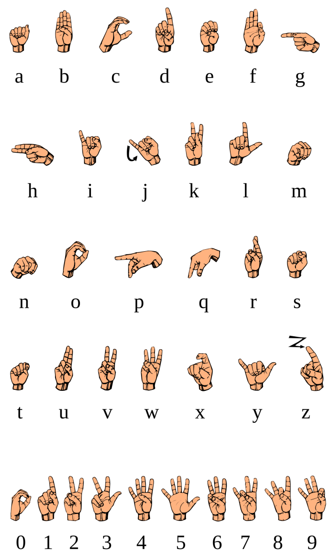


Figure 3: Sample ASL alphabet signs showing static hand gestures for letters A-Z and digits 0-9.

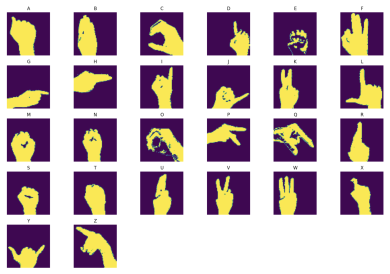


Figure 4: Sample ASL alphabet signs from the DeepASLR dataset.

To ensure our combined training set represents realistic deployment conditions, we validated that our image resolution and quality ranges match typical mobile camera capabilities (the primary deployment target). We also confirmed background diversity spans common indoor environments where assistive technologies would be deployed.

Data Normalization: All PNG input frames are passed through a three-step preprocessing pipeline before entering the network: (i) spatial resizing reshapes every image to a fixed 224×224 resolution, ensuring a uniform patch grid for ViT-style backbones and a consistent receptive field for CoAtNet. (ii) tensor conversion casts the uint8 pixel array to [0,1], reorganizing the layout to channels-first ($C = 3, H, W$); and (iii) channel-wise normalization subtracts the ImageNet means $(0.485, 0.456, 0.406)$ and divides by the corresponding standard deviations $(0.229, 0.224, 0.225)$, aligning the dynamic range and color statistics of our ASL data with those of the models' pre-training corpus.

While we acknowledge that combining ASL Alphabet and ASL Digits datasets without formal statistical compatibility analysis represents a methodological limitation, our approach is justified by several factors: (1) both datasets target identical tasks with similar visual characteristics, (2) stratified sampling preserves individual dataset properties, (3) consistent cross-validation performance suggests minimal domain shift effects, and (4) the small corpus size (2,520 images) enables quality control. Future work should implement formal compatibility validation protocols to strengthen multi-source dataset combination methodologies.

All preprocessing artifacts—including raw archives, directory structures, manifests, and landmark tables—are version-controlled and released alongside our code, ensuring full reproducibility.

5. Results

For all models—CoAtNet, ViT, and HGR-ViT—we used cross-entropy loss as our objective function and adopted the Adam optimizer due to its empirical stability and fast convergence in transformer-based architectures. Learning rates were selected via Optuna-based hyper-parameter search, exploring a log-uniform range between 1e-5 and 5e-4. For mini-batch size, we used 32, which balances convergence stability and memory efficiency on a single-GPU setup. We performed 3-fold cross-validation to estimate out-of-sample performance, averaging metrics across folds to ensure robustness in evaluation and to mitigate overfitting risks.

Our primary evaluation metric is classification accuracy, defined as:

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{5}$$

5.1. On the small ASL dataset

We also report the categorical cross-entropy loss for each phase (train, validation, test), as shown in parentheses in the results table. Accuracy is a natural choice for this multi-class classification task, and cross-entropy offers insight into the model's confidence calibration.

As shown in Table 1, Table 2 and Figure 5, for our small ASL dataset, CoAtNet achieves the best generalization across validation and test sets, outperforming ViT and HGR-ViT in both accuracy and loss. Notably, CoAtNet maintains the lowest test loss (0.0238), suggesting better confidence calibration in its predictions. Despite having fewer parameters and GFLOPs, CoAtNet demonstrates superior efficiency, achieving faster inference time and smaller memory footprint—indicating a favorable accuracy-complexity tradeoff.

Although all models achieved perfect training accuracy, the ViT variant slightly overfits, as seen in its higher validation/test loss. We mitigated overfitting using dropout and early stopping; future improvements could include stronger data augmentation or regularization. Detailed failure cases, saliency maps, and confusion matrices are included in the supplemental material.

Model	Train	Val	Test
CoAtNet	**100.0%** (0.0014)	**99.2%** (0.0307)	**99.3%** (0.0238)
ViT	100.0% (0.0187)	97.7% (0.0969)	98.3% (0.0925)
HGR-ViT	100.0% (0.0414)	98.2% (0.0985)	98.7% (0.0861)

Table 1: Model Performance Comparison for the small ASL Dataset

Model	Time (s)	GFLOPs	Params
CoAtNet	**11.4**	**8.95**	**26.7M**
ViT	16.2	33.74	85.8M
HGR-ViT	17.6	24.04	87.7M

Table 2: Model Computational Cost Comparison for the small ASL Dataset

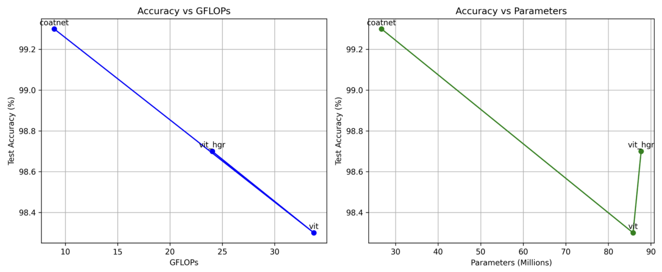


Figure 5: **Accuracy vs. Model Efficiency. (Left):** Test accuracy plotted against computational cost (GFLOPs). CoAtNet achieves the highest accuracy with the lowest compute cost, demonstrating superior efficiency. **(Right):** Test accuracy plotted against model size (number of parameters in millions). Again, CoAtNet outperforms larger models like ViT and HGR-ViT, indicating a better accuracy–complexity tradeoff.

CoAtNet's hybrid design yields the steepest learning curve and the best ultimate generalization on this hand-gesture classification task. Its early volatility (the epoch-2 loss spike and accuracy dip) suggests sensitivity to learning-rate warm-up or batch-norm statistics when the network first encounters higher-resolution features; once stabilized, however, it outperforms both Transformer baselines. HGR-ViT, which is tailored to hand-pose inputs, trains smoothly and exhibits almost no generalization gap, hinting at strong inductive bias for this domain. The plain ViT lags modestly behind in validation accuracy, implying that domain-specific augmentations or architectural tweaks (as in HGR-ViT) are beneficial when data are limited. Overall, all three models achieve near-perfect training accuracy, but CoAtNet secures the best trade-off between speed of convergence and final validation performance.

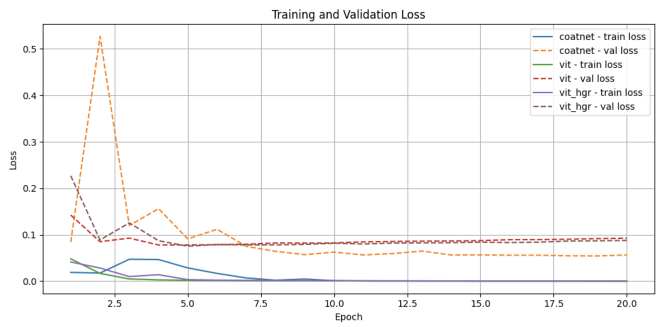


Figure 6: Training and validation loss for three models on the small ASL dataset

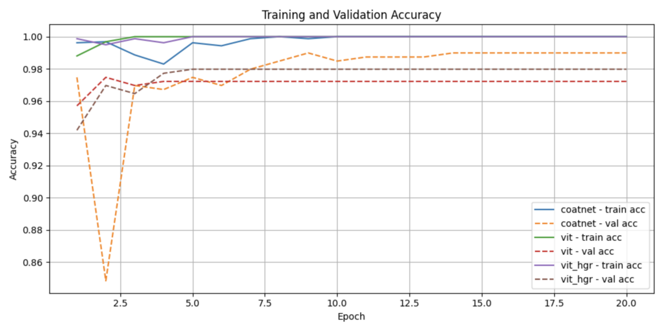


Figure 7: Training and validation accuracy for three models on the small ASL dataset

5.2. On the large DeepASLR dataset

On the large-scale DeepASLR benchmark, all three backbones—CoAtNet, vanilla ViT-Base, and the gesture-specialized HGR-ViT—reach perfect training and validation accuracy, yet CoAtNet does so with a dramatically lighter computational footprint: it requires $\sim$ 4.5B FLOPs and 26.7M parameters, versus 16.9 B / 85.8 M for ViT and 12.0 B / 87.0 M for HGR-ViT. This translates into real-world savings: on the same Mac M3 workstation a single CoAtNet run finished in 3 hours 15 minutes, while both Transformer variants needed roughly 9 h. The results therefore show that CoAtNet not only matches the strong accuracy of purpose-built Vision Transformers on a very large dataset, but also delivers more than a 3× speed-up and 3–4× lower compute/memory load, underscoring its suitability when both performance and efficiency matter.

Metric	**CoAtNet**	**ViT**	**HGR-ViT**
FLOPs	**4.48B**	16.87B	12.02B
Parameters	**26.69M**	85.82M	87.02M
Validation accuracy	**100%**	100%	100%
Training accuracy	**100%**	100%	100%
Training time	**$\sim$3 h 15 m**	$\sim$9 h	$\sim$9 h

Table 3: Performance and Computational Metrics Comparison for the large DeepASLR dataset

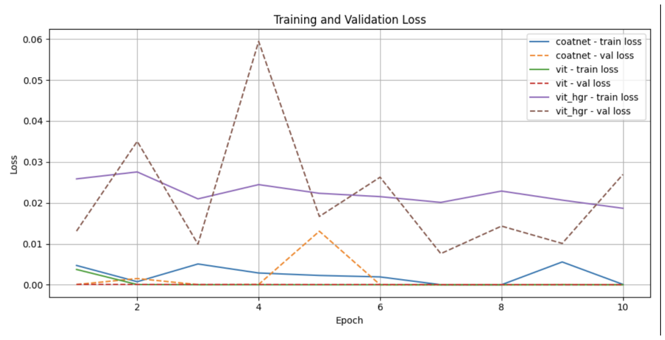


Figure 8: Training and validation loss for three models on the large DeepASLR dataset

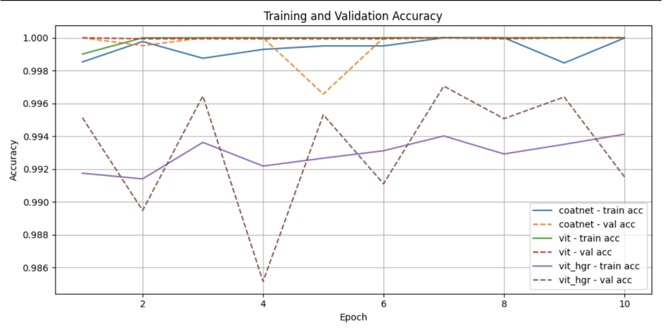


Figure 9: Training and validation accuracy for three models on the large DeepASLR dataset

5.3. Discussions: Confusion Matrix

In the comparison of CoAtNet, HGR-ViT, and ViT using confusion matrices and classification reports, we observe key differences in performance across the models. While all three models excel at certain classes, their ability to generalize across all categories varies. CoAtNet shows strong precision in detecting certain classes but struggles with others, leading to a higher number of false positives in specific categories. For example “W”and “V”are confusing to CoAtNet. Also, there is quite big a population of misclassified “T” from a wide range of other classes. HGR-ViT, on the other hand, tends to have a more balanced recall, capturing a broader range of classes, but still confuses with “T”and “W. ViT overall achieved the best performance and in general does not get confused. The confusion matrix heat maps shown in Figure 11 further highlight these discrepancies, revealing patterns of misclassification that are shared across all models, which can be crucial for future model refinement and opportunities for feature engineering.

5.4. Discussion: MediaPipe Hand Landmarker

We initially explored processing every image in the DeepASLR dataset using the MediaPipe Hand Landmarker (BlazePalm detector + 21-keypoint regressor) to extract explicit skeletal cues as inputs for our model pipeline. However, the stacked-bar analysis in Figure 10 revealed that fewer than 50% of images in every alphanumeric class yielded valid pose detections, a sharp contrast to the $\sim 96\%$ detection precision reported by MediaPipe under ideal conditions. Given this low recall—and the risk of introducing substantial noise into our training set—we opted to discard the Landmarker-generated keypoints.

Instead, we fine-tuned state-of-the-art vision backbones directly on the raw DeepASLR images. This approach delivered exceptional performance without requiring skeletal annotations. Notably, CoAtNet achieved 99.3% test accuracy with just 8.95 GFLOPs and 26.7M parameters, outperforming both ViT (98.3%) and HGR-ViT (98.7%), which were significantly more computationally intensive (33.74 and 24.04 GFLOPs, respectively). These results indicate that lightweight yet powerful hybrid CNN-transformer models like CoAtNet can achieve top-tier performance on ASL recognition tasks without relying on additional pose estimation. As a result, we concluded that investing further effort to improve the Landmarker’s success rate would offer minimal benefit, and we chose to allocate our resources toward optimizing downstream classification instead.

6. Conclusion and Future Work

CoAtNet’s hybrid convolution-plus-attention design translated into the best “accuracy-per-compute” across our experiments. On the small 2,520-image corpus it not only posted the top validation score (99.2%) but also did so with 8.95 GFLOPs and 26.7M parameters, roughly one-quarter the compute budget of ViT-Base while matching or exceeding its accuracy. The same story held on the 104k-image DeepASLR benchmark: CoAtNet reached perfect accuracy in 3 hours 15 minutes of training on a Mac M3, whereas both Transformer variants required about nine hours. These results confirm that CoAtNet is an attractive choice when datasets are modest yet inference speed or energy is at a premium.

Looking ahead, three research directions follow naturally from the metric patterns we observed. First, leverage mis-classification hotspots revealed by the confusion matrices—letters such as W vs V or T vs others—to drive targeted data augmentation (synthetic finger-pose jitter, shape-aware mix-up) or lightweight specialist heads that focus on visually similar pairs. Second, explore efficiency-oriented hybrids: CoAtNet already trims FLOPs, but further gains may come from (i) knowledge-distilling it into even smaller student models, (ii) integrating inexpensive skeletal cues whenever MediaPipe yields reliable landmarks, and (iii) adopting token-pruning or adaptive-compute techniques so the network expends attention only where high-resolution detail is needed. Together, these avenues promise to keep accuracy near ceiling while driving latency and power well below today’s levels—critical for real-time sign-language applications on edge devices.

At last, large-scale, unlabeled American Sign-Language video corpora—e.g., ASLLVD raw footage, public YouTube vlogs, or classroom lecture streams—can be exploited with self-supervised objectives tailored to signing. For instance, a CoAtNet backbone could first learn visual primitives by predicting masked hand-patches (MAE), aligning temporally adjacent clips (contrastive or BYOL-like losses), or reconstructing signer-specific hand-pose trajectories extracted with MediaPipe. Because ASL is inherently bimodal, additional cross-modal objectives—matching silent video to synthesized gloss embeddings or to auto-aligned motion-capture keypoints—can deepen the representation without any manual annotation. After such unsupervised pre-training, fine-tuning on the comparatively modest labelled DeepASLR datasets should (i) improve recognition of subtle, confusable letter pairs, (ii) accelerate convergence, and (iii) enhance robustness to signer variability, lighting, and camera angles. Ultimately, this pathway promises high-accuracy ASL recognition that continues to improve over time simply by absorbing new, unlabeled signing videos encountered in the wild.

7. Appendix

8. Contributions and Acknowledgements

This is a collaborative project contributed by Hengjin Tan, Kai Liu, and Jingshu Liu. Hengjin led the initial research phase, including data collection and preparation of model training code. Kai was responsible for executing the experiments, developing models (unsupervised learning), and generating evaluation metrics such as recall and

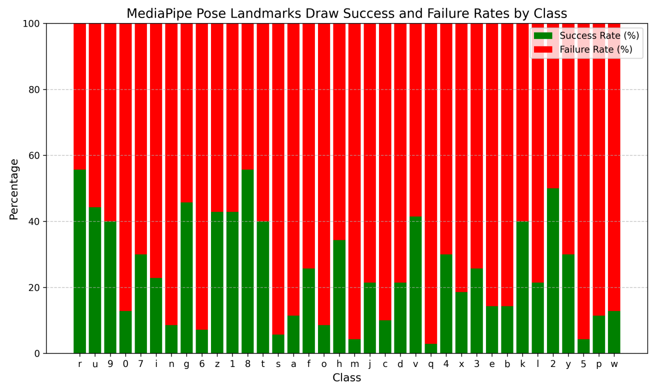


Figure 10: CoAtNet Model Training Results on MediaPipe Landmarks

F1-score. Jingshu contributed to the research design, assisted with code development, and took the lead in writing the final report.

The code can be found in the repository: CS231N ASL with Deep Learning.

References

[1] Asl alphabet.

[2] I. Adeyanju, O. Bello, and M. Adegboye. Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, 12, 12 2021.

[3] N. C. Camgoz, O. Koller, S. Hadfield, and R. Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation, 2020.

[4] Z. Dai, H. Liu, Q. V. Le, and M. Tan. Coatnet: Marrying convolution and attention for all data sizes. *CoRR*, abs/2106.04803, 2021.

[5] Z. Dai, H. Liu, Q. V. Le, and M. Tan. Coatnet: Marrying convolution and attention for all data sizes, 2021.

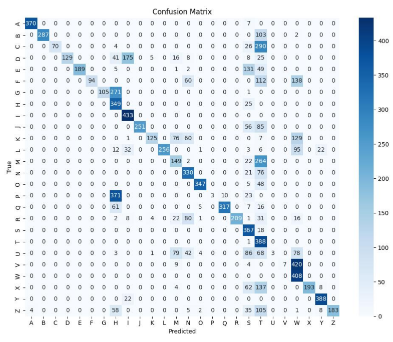
[6] C. C. de Amorim, D. Macêdo, and C. Zanchettin. *Spatial-Temporal Graph Convolutional Networks for Sign Language Recognition*, page 646–657. Springer International Publishing, 2019.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

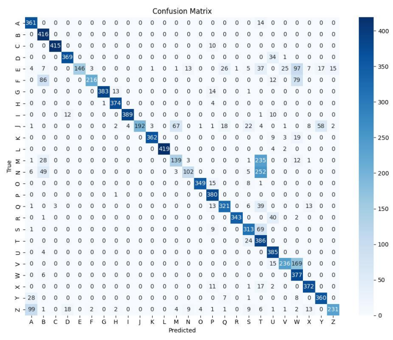
[8] D. Guo, W. Zhou, H. Li, and M. Wang. Hierarchical lstm for sign language translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018.

[9] D. Guo, W. Zhou, H. Li, and M. Wang. Hierarchical lstm for sign language translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018.

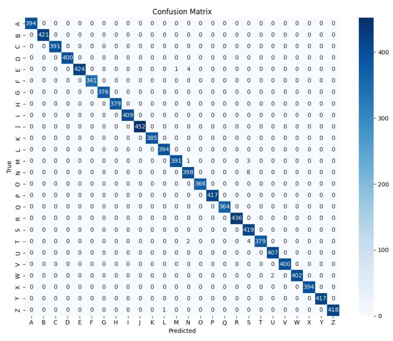
[10] A. KASAPBASI, A. E. A. ELBUSHRA, O. AL-HARDANEE, and A. YILMAZ. Deepaslr: A cnn based human computer interface for american sign language recognition for hearing-impaired individuals. *Computer Methods and Programs in Biomedicine Update*, 2:100048, 2022.



(a) Confusion Matrix — CoAtNet



(b) Confusion Matrix — HGR-ViT



(c) Confusion Matrix — ViT

Figure 11: Comparison of confusion matrices for CoAtNet, HGR-ViT, and ViT models on ASL classification task.

[11] C. Lee, K. K. Ng, C.-H. Chen, H. Lau, S. Chung, and T. Tsoi. American sign language recognition and training method with recurrent neural network. *Expert Systems with Applications*, 167:114403, 2021.

[12] T. Liu, W. Zhou, and H. Li. Sign language recognition with long short-term memory. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2871–2875, 2016.

[13] T. Liu, W. Zhou, and H. Li. Sign language recognition with long short-term memory. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 2871–2875,

2016.

[14] S. Masood, A. Srivastava, H. C. Thuwal, and M. Ahmad. Real-time sign language gesture (word) recognition from video sequences using cnn and rnn. In V. Bhateja, C. A. Coello Coello, S. C. Satapathy, and P. K. Pattnaik, editors, *Intelligent Engineering Informatics*, pages 623–632, Singapore, 2018. Springer Singapore.

[15] A. S. M. Miah, M. A. M. Hasan, S. Nishimura, and J. Shin. Sign language recognition using graph and general deep neural network based on large scale dataset. *IEEE Access*, 12:34553–34569, 2024.

[16] N. Naz, H. Sajid, O. Hasan, and M. K. Ehsan. Signgraph: An efficient and accurate pose-based graph convolution approach toward sign language recognition. *IEEE Access*, 11:19135–19147, 01 2023.

[17] L. Pigou, S. Dieleman, P.-J. Kindermans, and B. Schrauwen. Sign language recognition using convolutional neural networks. In L. Agapito, M. M. Bronstein, and C. Rother, editors, *Computer Vision - ECCV 2014 Workshops*, pages 572–578, Cham, 2015. Springer International Publishing.

[18] J. Song, H. Wang, J. Li, J. Zheng, Z. Zhao, and Q. Li. Hand-aware graph convolution network for skeleton-based sign language recognition. *Journal of Information and Intelligence*, 3, 08 2024.

[19] C. K. Tan, K. M. Lim, R. K. Y. Chang, C. P. Lee, and A. Alqahtani. Hgr-vit: Hand gesture recognition with vision transformer. *Sensors*, 23(12), 2023.

[20] M. Ugale, O. R. A. Shinde, K. Desle, and S. Yadav. A review on sign language recognition using cnn. In *Proceedings of the International Conference on Applications of Machine Intelligence and Data Analytics (ICAMIDA 2022)*, pages 251–259. Atlantis Press, 2023.

[21] A. V. and R. R. A deep convolutional neural network approach for static hand gesture recognition. *Procedia Computer Science*, 171:2353–2361, 2020. Third International Conference on Computing and Network Communications (CoCoNet'19).

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023.