# ARDA VSLNet: FiLM-Enhanced Knowledge Distillation for Efficient Natural Language Query Localization in Ego4D Videos

Rino Di Chiara,[†] Daniele Famà,[†] Alice Fico,[†] Andrea Lolli[†]
[†]Data Science and Engineering, Politecnico di Torino
{s314931, s345153, s347322, s346165}@studenti.polito.it

## Abstract

*Natural Language Query (NLQ) is one of the tasks of the Ego4D benchmark, which aims to localize the temporal segment of an untrimmed egocentric video that best matches a query written in natural language. In this paper, we conduct an empirical evaluation of two model compression techniques, knowledge distillation and quantization, applied to VSLNet, a span-based question-answering model. To weaken accuracy degradation due to compression , we introduce a lightweight architectural modification using a FiLM layer and analyze its effects across different configurations. All variants of the model are evaluated in terms of memory usage, inference latency, and localization accuracy with the goal of maximizing the trade-off between accuracy and computational efficiency. Although the distillation strategies yielded promising and insightful results, we were unable to complete the quantization pipeline due to technical limitations.*

## A. Introduction

The field of egocentric vision has seen rapid progress in recent years, driven by the increasing availability of first-person video data included in famous datasets as Ego4D [1]. One of the core challenges in this domain is Natural Language Query (NLQ), which requires localizing temporal segments within egocentric videos that correspond to a given prompt. VSLBase and VSLNet [2] serve as foundational baseline models for this task.

However, recent models on Ego4D challenges—such as InternVideo-Ego4D [3], GroundNLQ [4], and OSGNet [5]—commonly rely on large Vision Transformer (ViT) backbones and multi-stream fusion, often resulting in models with hundreds of millions of parameters. These architectures, while effective, are computationally heavy and memory-intensive, posing significant limitations in applications with constrained memory or power consumption.

As part of our project, we first conducted a data exploration phase on the Ego4D datasets, followed by comparative experiments using different video encoders (Omnivore and EgoVLP) and text embedding methods (BERT and GloVe), establishing a baseline for our task and experiments.

Our study highlights the configuration of VSLNet with EgoVLP and GloVe as the most appropriate baseline for exploring model compression. We aim to investigate whether and to what extent this model can be compressed, using knowledge distillation, without significantly degrading its localization accuracy. We hypothesize that introducing a lightweight FiLM layer can enhance the model's representational capacity, allowing it to better retain task-relevant information while achieving substantial reductions in memory usage and inference latency.

Empirical results demonstrate that, for several compression settings, our pipeline yields substantial reductions in model size and latency with only minor accuracy degradation. Owing to tooling limitations for large-scale sequence models and time constraints, we were unable to complete the full quantization pipeline; we leave this component for future work. All source code and intermediate artifacts are openly released in our GitHub repository[1] to support further research on resource-efficient egocentric video grounding.

## B. Methodology

We first evaluate VSLBase and VSLNet on the NLQ task to identify the most effective setup. After selecting the best-performing variant, we introduce the FiLM layer, and subsequently the two knowledge distillation strategies to reduce model complexity.

### B.1. Text and Video Representation

In our experimental study, we follow the framework of the NLQ challenge in EGO4D by treating the task as

[1]https://github.com/AndreaLolli2912/Ego4D-NLQ

a Natural Language Video Localization (NLVL) problem. This approach allows us to use span-based question answering methods, where the video is treated like a text passage, following the idea proposed in [2]. Eventually, we adopt both VSLBase and its enhanced variant VSLNet, which are specifically designed for this task. Although we conduct experiments with both models, our primary focus lies on VSLNet, since it extends VSLBase by incorporating query-guided highlighting (QGH), which focuses the model's attention on a region surrounding the video-span target moment. This strategy provides a richer temporal context and reduces the search space, allowing for more accurate localization in continuous video streams.

For the video modality, we use visual features extracted from two pre-trained encoders — Omnivore and EgoVLP. Omnivore is a unified vision transformer trained on a mixture of images, videos, and 3D data. It breaks videos and images into small patches and analyzes them across space and time using attention. This allows it to learn useful visual representations without needing the inputs to be perfectly aligned or matched [6]. EgoVLP, on the other hand, is a multimodal encoder specifically tailored for egocentric video understanding. It uses contrastive learning to make the features of videos and text more similar when they refer to the same content, helping them to align in a shared representation space [7].

For the embedding part, we used two models: BERT (Bidirectional Encoder Representations from Transformers) and GloVe (Global Vectors for Word Representation). BERT is a deep model that uses a multi-layer bidirectional Transformer to create contextual word embeddings, meaning that the representation of a word depends on the context around it [8]. On the other hand, GloVe generates word vectors by analyzing how often words appear together across a large text corpus, relying on global co-occurrence statistics to capture meaning [9].

For evaluation, we used the Recall@k, tIoU=m metric, commonly adopted in Natural Language Query (NLQ) tasks, as suggested in [1]. This metric measures the percentage of query instances for which at least one prediction among the top k has a temporal Intersection over Union (tIoU) greater than or equal to m. Table 1 reports the baseline performance of the NLQ task, which we used as a reference point for our own experiments.

Our results are shown and discussed in Table 2 in the C.1 Section.

We chose VSLNet with EgoVLP and GloVe as our reference configuration as it offers a good trade-off between accuracy and efficiency, making it a solid basis for applying our compression techniques. Therefore, all subsequent experiments are conducted using this setup.

Table 1. Performance of the NLQ baselines

| **Model (Split)** | **IoU=0.3 (%)** | | **IoU=0.5 (%)** | |
|---|---|---|---|---|
| | r@1 | r@5 | r@1 | r@5 |
| VSLNet (Val) | 5.45 | 10.74 | 3.12 | 6.63 |
| VSLNet (Test) | 5.47 | 11.21 | 2.80 | 6.57 |

For more details on this choice, please refer to Section C.1.

## B.2. FiLM Layer

At this point of the work, we show how the introduction of the FiLM layer impacts the performance of VSLNet model. Feature-wise Linear Modulation (FiLM) layers modify the internal computation of neural networks by applying a simple, feature-wise affine transformation based on conditioning information derived from the features extracted from the natural language queries.

More formally, FiLM learns functions f and h which output $\gamma_{i,c}$ and $\beta_{i,c}$ as a function of input $x_i$

$$\gamma_{i,c} = f(x_i), \quad \beta_{i,c} = h(x_i) \tag{1}$$

where $\gamma_{i,c}$ and $\beta_{i,c}$ modulate the activations $F_{i,c}$ of the network for the $i$-th input and $c$-th feature map via the affine transformation:

$$\text{FiLM}(F_{i,c}|\gamma_{i,c}, \beta_{i,c}) = \gamma_{i,c}F_{i,c} + \beta_{i,c} \tag{2}$$

In this way FiLM facilitates early interaction between the video representation and the natural language query, improving the model's capacity to align multimodal semantics while preserving a compact architecture. In the reference model [10] the computation of the FiLM parameters $\gamma$ and $\beta$ is conditioned on the output context vector obtained from the query through a Recurrent Neural Network.

In our approach, we propose a simplified and computationally lighter alternative. Instead of relying on recurrent processing, we applied a mean pooling operation to the sequence of token embeddings produced before. This produces a fixed vector that represents an aggregate of the query information. We then apply to it a linear layer and duplicate its dimension, after that we use the output obtained to calculate the final vectors $\gamma$ and $\beta$.

Finally, we apply Equation (2) to compute the FiLM-modulated features using the obtained parameters. This choice reduces computational cost and demonstrates that simple architectural designs can still support efficient multimodal interaction when combined with strategic placement and tuning.

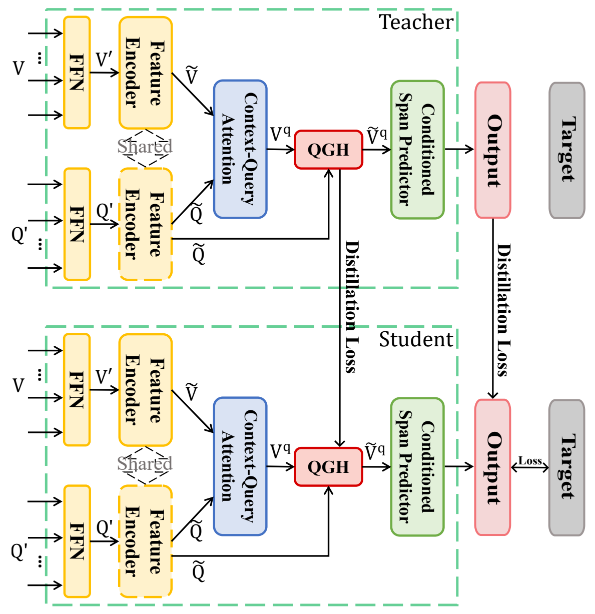
Figure 1. Illustration of the Knowledge Distillation process of VSLNet [2], where the teacher model transfers knowledge to the student model.

## B.3. Knowledge Distillation

In the following paragraphs, we introduce two strategies to distill VSLNet. The first is a standard knowledge distillation approach, while the second aims to distill the model in a block-wise manner.

### B.3.1 Standard Distillation

In this first distillation strategy, the student network is trained using a composite weighted loss that integrates two key components: the soft predictions produced by the teacher network and the ground-truth classification labels [11]. This formulation encourages the student not only to match the final outputs of the teacher but also to maintain fidelity to the original supervised task. To further enhance knowledge transfer, we introduce an additional loss term between the internal representations of the teacher and student, specifically focusing on the QGH (Query-Guided Highlighted) module. This weighted loss contributes to aligning the intermediate feature spaces of the two models.

$$\mathcal{L} = \mathcal{L}_{\text{span}} + \mathcal{L}_{\text{QGH}} \tag{3}$$

where:

$$\mathcal{L}_{\text{span}} = \alpha_{\text{stud}}\mathcal{L}_{\text{ce\_stud}} + \alpha_{\text{s\_t}}\mathcal{L}_{\text{stud\_teacher}} \tag{4}$$

$$\mathcal{L}_{\text{QGH}} = \beta_{\text{stud}}\mathcal{L}_{\text{QGH\_stud}} + \beta_{\text{s\_t}}\mathcal{L}_{\text{QGH\_stud\_teacher}} \tag{5}$$

The distillation process involves architectural compression through reduction of layers within the Feature Encoder; The number of depthwise separable convolutional blocks and attention heads in the multi-head attention block. Additionally, we reduced the dimensionality of the feature embedding space, which determines the size of vector representations used throughout the network architecture. By leveraging both output-level supervision and intermediate feature-level alignment (Fig. 1), our approach facilitates a comprehensive distillation process that captures both semantic predictions and deeper representational cues learned by the teacher.

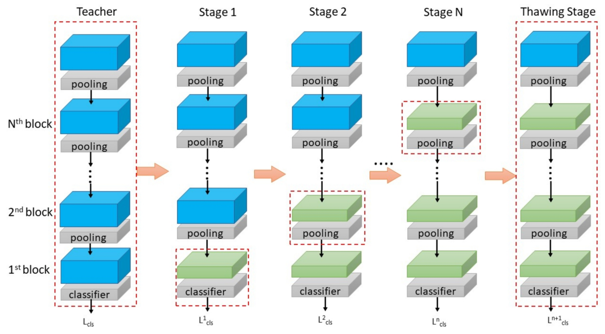
Figure 2. Overview of CBKD.

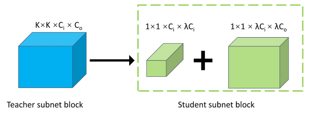

Figure 3. Illustration of the student subnetwork block design.

### B.3.2 Counterclockwise block-by-block knowledge distillation (CBKD)

The research by Gotmare et al. theoretically demonstrates that teacher models and student models exhibit capacity differences in their intermediate hidden layers, leading to different feature representation abilities. Therefore, merely acquiring the teacher's output feature knowledge is insufficient. By transferring the intermediate layer knowledge of the teacher model, it helps bridge this gap [12].

Counter-clockwise Block-wise Knowledge Distillation (CBKD) tackles the teacher-student "generation gap" by slicing both networks at each down-sampling layer [13]. It gradually replaces teacher subnet blocks with corresponding student subnet blocks, proceeding from deep to shallow layers. During each stage of the block replacement process, the other subnet blocks within the teacher network remain unchanged, enabling the student model to acquire intermediate layer feature knowledge from the teacher model. At each stage, the structural similarity between the teacher and student models narrows the generational gap. The comprehensive process of CBKD is vividly depicted in Fig. 2. The red dotted box indicates the subnet block trained for the current stage. For VSLNet we divide the architecture into four consecutive blocks: (i) the visual-projection and embedding layer, (ii) the feature encoder layer, (iii) the context–query interaction module—encompassing attention, concatenation, and highlight operations—and (iv) the conditioned predictor head.

The shallowest segment of the teacher network is directly preserved within the corresponding shallow layer of the student model, reflecting a direct inheritance of foundational knowledge. To derive the student subnet block, we systematically reduce the number of channels of the convolutional layers within the corresponding teacher subnet block. To facilitate the integration of the student subnet block into the overall network architecture, we employ a convolutional layer or linear layer to downscale / upscale the number of channels over the batch of data, in order to adjust the input channel number of the student subnet block to ensure compatibility with the surrounding layers, and maintaining dimensional consistency throughout the substitution process, as vividly illustrated in Fig. 3. Let $C_i, C_o$ and $\lambda$ represent the input channels, output channels and compression ratio of the teacher subnet block. This design choice ensures that the student subnet block can seamlessly replace its teacher counterpart without disrupting the overall flow of information within the network. The ratio of the number of channels in the student subnet block to that in the teacher subnet block serves as a quantifiable metric of the compression applied.

CBKD is segmented into N+1 phases. The transfer occurs progressively, moving from the deepest teacher subnet blocks to the shallowest, with each student subnet block mirroring its teacher counterpart in terms of knowledge acquisition. In the first N phases, only one student subnet block is actively trained, while the remaining networks within the student model remain in a frozen state. At the final phase, all parameters within the student model are unfrozen, marking the beginning of a comprehensive training session designed to fine-tune the entire student model, reducing its reliance on the teacher model promoting the discovery of a more suitable parameter distribution.

At every distillation stage CBKD minimises a single objective computed only at the network output, thereby avoiding any ill-posed layer-wise alignment between heterogeneously pruned teacher–student blocks. Let $\mathbf{s}^T, \mathbf{e}^T, h^T$ (teacher) and $\mathbf{s}^S, \mathbf{e}^S, h^S$ (student) denote the start-logits, end-logits and highlight scores, and let $\tau > 1$ be the distillation temperature. Following the seminal work of [14], the temperature $\tau$ is used to soften the soft-max distribution of both teacher and student. Dividing the logits by $\tau > 1$ spreads probability mass across non-argmax classes, thereby transferring the teacher's so-called dark knowledge—its relative preference among incorrect options—to the student. To preserve gradient magnitude, the resulting KL term is multiplied by $\tau^2$, compensating for the $1/\tau^2$ scale factor introduced by the division. This simple trick has become standard practice in knowledge-distillation pipelines and is implemented unchanged in our CBKD loss.

The loss is

$$\mathcal{L}_{\text{CBKD}} = \big(\mathcal{L}_{\text{loc}} + \lambda_{\text{hl}}\mathcal{L}_{\text{hl}}\big) + \\ + \alpha_{\text{KD}}\tau^2\Big(\text{KL}_s + \text{KL}_e + \beta_{\text{hl}}\,\text{BCE}\Big) \quad (6)$$

where $\text{KL}_s = \text{KL}\big(\mathbf{s}^T/\tau \parallel \mathbf{s}^S/\tau\big)$ and $\text{KL}_e = \text{KL}\big(\mathbf{e}^T/\tau \parallel \mathbf{e}^S/\tau\big)$ are the start- and end-logit divergences, while $\text{BCE} = \text{BCE}\big(\sigma(h^T), \sigma(h^S)\big)$ matches highlight probabilities. Temperature $\tau > 1$ softens both posteriors, and the factor $\tau^2$ restores gradient magnitude. The coefficients $\lambda_{\text{hl}}, \alpha_{\text{KD}}, \beta_{\text{hl}}$ balance highlight supervision, distillation strength, and highlight-KD, respectively.

## C. Experiments and results

Each experiment altered a single component while keeping the rest of the pipeline fixed:

### C.1. Baseline Model Configurations

Based on the configurations and techniques described in Section B.1, we now present the experimental results evaluating the impact of our compression strategies.

As observed in Table 2, both VSLBase and VSLNet achieve their best performance when combining EgoVLP visual features with BERT-based text embeddings. Although it was not the best performing configuration, we chose to adopt the EgoVLP, GloVe settings as a reference for the following experiments. This choice is driven by a trade-off between performance and computational efficiency: GloVe embeddings are significantly lighter than BERT (pre-trained static vectors instead of transformer-based contextual representations), performance degradation is relatively limited - on average a 1 point drop in Recall@K, corresponding to roughly 9-10%. This lightweight configuration better supports our goal of building a compact, computationally efficient model—an advantage that becomes even more important in the distillation experiments presented in the next sections. Moreover, across several trials we found that adding FiLM conditioning layers produced a larger performance boost when combined with GloVe embeddings than with BERT. A plausible explanation is that the static GloVe vectors leave more representational "headroom" for FiLM to inject video-specific modulation, whereas BERT's already-contextualized embed-

Table 2. Performance of the NLQ task after fine-tuning

| Model | Visual Feature | Embedding | mIoU@0.3 |  | mIoU@0.5 |  |
|---|---|---|---|---|---|---|
|  |  |  | R@1 | R@5 | R@1 | R@5 |
| VSLNet | Omnivore | BERT | 6.09 | 13.09 | 3.33 | 7.61 |
|  |  | GloVe | 5.37 | 11.44 | 2.79 | 6.92 |
|  | EgoVLP | BERT | **10.12** | **18.95** | **6.14** | **12.31** |
|  |  | GloVe | 9.10 | 16.96 | 5.16 | 11.03 |
| VSLBase | Omnivore | BERT | 6.27 | 14.04 | 3.79 | 9.11 |
|  |  | GloVe | 6.45 | 13.63 | 3.92 | 8.62 |
|  | EgoVLP | BERT | **9.06** | **17.35** | **5.52** | **11.64** |
|  |  | GloVe | 8.23 | 16.13 | 4.90 | 10.71 |

dings are highly expressive; additional FiLM transformations can therefore disrupt rather than enhance their semantics, leading to smaller gains.

These results were obtained by fine-tuning the models with the following optimal hyperparameters: an embedding dimension of 128, 30 training epochs, a maximum position length of 128, and an initial learning rate of 0.0006.

## C.2. FiLM

Building on the experimental setup described above, we use the EgoVLP + GloVe model as the reference architecture to evaluate the impact of inserting a lightweight FiLM layer. This model, introduced in the previous section, was selected due to its favorable balance between performance and computational efficiency. Before compressioning model we introduce a lightweight FiLM layer and explore several configuration strategies. In particular we evaluate 3 possible positions of this layer with respect to the encoder: before, after, and inside the encoder. Inside the feature encoder, we examine multiple variants that differ in the exact placement of the FiLM layer; separately after the *PositionalEmbedding*, *DepthwiseSeparableConvBlock* and *MultiHeadAttentionBlock*, or simultaneously in the *multi* configuration. The results of these experiments are reported in the table below 3, which also includes the baseline model without any FiLM layer to provide a comprehensive overview. Based on the analysis of the results, we selected the configuration where the FiLM layer is placed inside the encoder, specifically the multi variant, which consistently provided the best performance.

## C.3. Distillation

Experiments on distillation involve a trade-off between minimizing model size and maximizing performance accuracy. In both standard knowledge distillation (KD) and CBKD, we distilled a teacher model with a feature embedding size of 256, intended to enrich the learning space and provide a more expressive feature representation. Figure 4 illustrates the model's performance as a function of GFLOPs. Table 4 reports the results obtained from the different distilled models configurations, which are further analyzed in the following paragraphs.

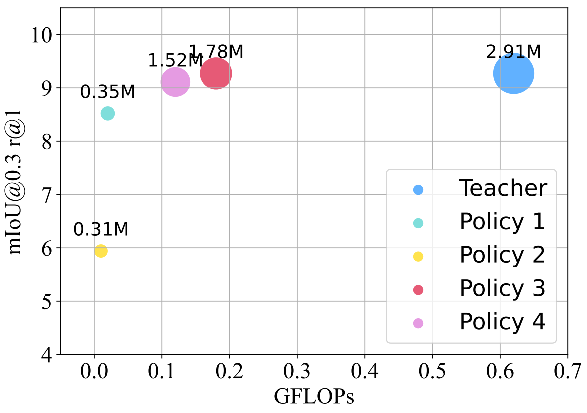

Figure 4. Different model size, comparing gflop and miou.

### C.3.1 Standard Distillation

The key hyperparameters introduced by the distillation framework include: the dimensionality of the student's feature embedding space, the number of layers in the *Feature Encoder*, the head size of the multi-head attention mechanism and the weights $\alpha$ and $\beta$ (as defined in Eq. 5) used in the student model loss.

The parameter configurations explored are summarized in Table 5. Additionally, the head size of the multi-head attention mechanism was reduced from 8 to 2. We selected as the best model the one obtained with a projected feature dimension of 32 and a single layer in the Feature Encoder. This configuration achieved an 88.47% reduction in model size, with a 11.98% drop in mIoU@0.3 R@1 metric. Additionally, when com-

Table 3. Performance comparison of FiLM configurations

| **Position** | **Configuration** | **mIoU@0.3** | | **mIoU@0.5** | |
|---|---|---|---|---|---|
| | | R@1 | R@5 | R@1 | R@5 |
| - | - | 9.10 | 16.96 | 5.16 | 11.03 |
| bef. encoder | - | 7.56 | 14.92 | 4.54 | 9.53 |
| aft. encoder | - | 8.75 | 16.52 | 5.37 | 11.07 |
| ins. encoder | multi | 8.96 | **17.37** | 5.45 | 11.20 |
| | after pos | **9.68** | 16.91 | **5.86** | 11.31 |
| | after conv | 9.11 | 17.32 | 5.29 | **11.67** |
| | after attn | 8.93 | 16.52 | 5.39 | 11.10 |

Table 4. Performance comparison of distillations

| **Strategy** | **Policy** | **mIoU@0.3** | | mIoU@0.5 | | **#params (M)** | **GFLOPs** |
|---|---|---|---|---|---|---|---|
| | | R@1 | R@5 | R@1 | R@5 | | |
| | Teacher | 9.68 | 17.63 | 5.27 | 10.97 | 3.043 | 0.62 |
| SKD | Policy 1 | 8.52 | 16.37 | 4.98 | 10.53 | 0.351 | 0.02 |
| | Policy 2 | 5.94 | 13.11 | 3.54 | 8.83 | 0.311 | 0.01 |
| CBKD | Policy 3 | 9.27 | 17.24 | 5.55 | 11.38 | 1.784 | 0.18 |
| | Policy 4 | 9.11 | 17.01 | 5.50 | 11.49 | 1.525 | 0.12 |

Table 5. Standard Distilled model configurations

| **Policy** | **DIM** | **#layers** | **Params (M)** | **Reduction** |
|---|---|---|---|---|
| P-1 | 32 | 1 | 0.351 | 88.47% |
| P-2 | 16 | 2 | 0.311 | 89.78% |

pared to a non-distilled model of comparable size, our approach achieved a 31.48% improvement, increasing the mIoU@0.3 R@1 score from 6.48 to 8.52.

### C.3.2 Compression Ratios for CBKD

To understand how much we can shrink the network without destroying accuracy we evaluate two distinct pruning schedules, *Policy 3* leaves the shallow branch (`Block 2`) moderately compressed and obtains a total parameter reduction of 39.95%, whereas *Policy–4* (P-4) applies an aggressive 0.1 keep ratio to all convolutions in that block achiving a total parameter reduction of 47.62%. Both policies use the same (already aggressive) pruning settings for `Block 3` and `Block 4`, obtaining scores close to baseline. The resulting component-wise parameter reductions, are summarized in Table 6.

## D. Conclusions

In this paper, we tackled the challenge of efficient Natural Language Querying in egocentric videos by exploring model compression techniques on the Ego4D benchmark.

After comparing several combinations of visual and textual encoders, we selected VSLNet with EgoVLP and GloVe as our baseline due to its reduced resource requirements, despite BERT's slightly superior accuracy. As a potential future improvement, we propose replacing GloVe with a compact transformer-based encoder like DistilBERT to better balance performance and efficiency.

To prepare the model for compression, we introduced a lightweight FiLM layer aimed at improving its representational capacity. Although the accuracy gain was modest, it remains relevant given the task's inherent difficulty and low baseline performance - suggesting a valuable direction for future work. The application of the two distillation strategies mentioned in the above sections proved effective, yielding notable reductions in model size and inference latency with only minor accuracy degradation. These findings validate the effectiveness of compression approaches to improve the efficiency of NLQ in egocentric vision. As part of our effort to explore various compression strategies, we also ex-

| **Policy** | **Block / Component** | **Compression ratios $\lambda$** | **Params (M)** | **Reduction** |
|---|---|---|---|---|
| P-3 | Block 2 — DS stack / Attn | 0.30 / 0.37 | 0.724 | 24.9% |
| | Block 3 — CQA / Concat | 0.111 / 0.20 | 0.199 | 49.7% |
| | Block 4 — Encoder / Heads | 0.05 / 0.05 | 0.502 | 59.1% |
| P-4 | Block 2 — DS stack / Attn | 0.10 / 0.10 | 0.500 | 48.1% |
| | Block 3 — CQA / Concat | 0.111 / 0.20 | 0.199 | 49.7% |
| | Block 4 — Encoder / Heads | 0.05 / 0.05 | 0.502 | 59.1% |

Table 6. Component-level keep ratios and parameter savings for the two pruning schedules evaluated in our CBKD experiments.

perimented with quantization. However, due to tooling limitations in our PyTorch-based pipeline, its integration remains an open challenge. We leave this as future work, with the goal of further optimizing model efficiency.

## References

[1] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu *et al.*, "Ego4d: Around the world in 3,000 hours of egocentric video," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 18 995–19 012. 1, 2

[2] H. Zhang, A. Sun, W. Jing, and J. T. Zhou, "Span-based localizing network for natural language video localization," 2020. [Online]. Available: https://arxiv.org/abs/2004.13931 1, 2, 3

[3] G. Chen, S. Xing, Z. Chen, Y. Wang, K. Li, Y. Li, Y. Liu, J. Wang, Y.-D. Zheng, B. Huang, Z. Zhao, J. Pan, Y. Huang, Z. Wang, J. Yu, Y. He, H. Zhang, T. Lu, Y. Wang, L. Wang, and Y. Qiao, "Internvideo-ego4d: A pack of champion solutions to ego4d challenges," 2022. [Online]. Available: https://arxiv.org/abs/2211.09529 1

[4] Z. Hou, L. Ji, D. Gao, W. Zhong, K. Yan, C. Li, W.-K. Chan, C.-W. Ngo, N. Duan, and M. Z. Shou, "Groundnlq @ ego4d natural language queries challenge 2023," 2023. [Online]. Available: https://arxiv.org/abs/2306.15255 1

[5] Y. Feng, H. Zhang, M. Liu, W. Guan, and L. Nie, "Object-shot enhanced grounding network for egocentric video," 2025. [Online]. Available: https://arxiv.org/abs/2505.04270 1

[6] R. Girdhar, M. Singh, N. Ravi, L. van der Maaten, A. Joulin, and I. Misra, "Omnivore: A single model for many visual modalities," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 16 102–16 112. 2

[7] K. Q. Lin, J. Wang, M. Soldan, M. Wray, R. Yan, E. Z. XU, D. Gao, R.-C. Tu, W. Zhao, W. Kong, C. Cai, W. HongFa, D. Damen, B. Ghanem, W. Liu, and M. Z. Shou, "Egocentric video-language pretraining," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35. Curran Associates, Inc., 2022, pp. 7575–7586. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2022/file/31fb284a0aaaad837d2930a610cd5e50-Paper-Conference.pdf 2

[8] M. V. Koroteev, "Bert: A review of applications in natural language processing and understanding," 2021. [Online]. Available: https://arxiv.org/abs/2103.11943 2

[9] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, and W. Daelemans, Eds. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://aclanthology.org/D14-1162/ 2

[10] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," 2017. [Online]. Available: https://arxiv.org/abs/1709.07871 2

[11] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *arXiv preprint arXiv:1412.6550*, 2014. 3

[12] M. A. Langovoy, A. Gotmare, and M. Jaggi, "Unsupervised robust nonparametric learning of hidden community properties," 2018. [Online]. Available: https://arxiv.org/abs/1707.03494 3

[13] X. Lan, Y. Zeng, X. Wei, T. Zhang, Y. Wang, C. Huang, and W. He, “Counterclockwise block-by-block knowledge distillation for neural network compression,” *Scientific Reports*, vol. 15, p. 11369, 2025. [Online]. Available: https://www.nature.com/articles/s41598-025-91152-3 3

[14] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” 2015. [Online]. Available: https://arxiv.org/abs/1503.02531 4