Video, How Do Your Tokens Merge?

Sam Pollard
University of Bristol
sam.pollard@bristol.ac.uk

Michael Wray
University of Bristol
michael.wray@bristol.ac.uk

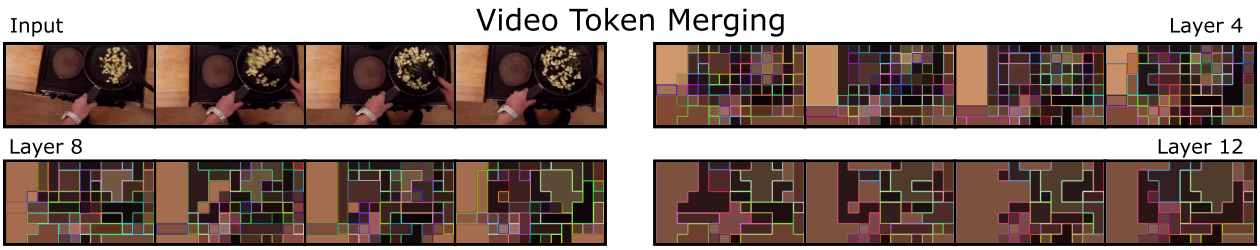


Figure 1. Video Token Merging reduces computation of video transformer models by successively merging tokens without re-training or additional learned parameters. We show how an input video has its tokens merged across different layers.

Abstract

Video transformer models require huge amounts of compute resources due to the spatio-temporal scaling of the input. Tackling this, recent methods have proposed to drop or merge tokens for image models, whether randomly or via learned methods. Merging tokens has many benefits: it can be plugged into any vision transformer, does not require model re-training, and it propagates information that would otherwise be dropped through the model. Before now, video token merging has not been evaluated on temporally complex datasets for video understanding. In this work, we explore training-free token merging for video to provide comprehensive experiments and find best practices across four video transformers on three datasets that exhibit coarse and fine-grained action recognition. Our results showcase the benefits of video token merging with a speedup of around 2.5X while maintaining accuracy (avg. -0.55% for ViViT). Code available at https://github.com/sjpollard/video-how-do-your-tokens-merge.

1. Introduction

Vision transformers [11] (ViTs) have quickly been established as the architecture of choice for solving the majority of computer vision problems. The long range dependencies that are afforded by self-attention significantly improve the flexibility and reasoning of models, when supplied with enough training data. This performance is not free as the quadratic complexity of the attention mechanism leads to exceedingly poor scaling.

This is especially prevalent in the video domain, in which the sequence length of tokens increases with both spatial and temporal dimensions. In the recent push for long video understanding [6, 13, 29], video vision transformers are therefore requiring more and more resources for both training and inference. Assuming that a 5 minute video at 224×224 pixels consists of the typical 16×16 spatial patches, when sampling 1 frame per second, the transformer sequence length is $58,800$ tokens. With this increase in complexity, it's simply not enough to hope that hardware scales up at the same rate: this requires a huge number of resources per video and is significantly longer than the sequence lengths that can be handled in a single window of context for current transformers. It is also worth noting that this sampling rate may not even be enough for fine-grained action understanding, where actions average 2.6s in duration [9].

Recently, token merging of transformers has been proposed for images [4], as a drop-in method for increasing the throughput of ViT models by reducing the token sequence length. This has several benefits over other methods: the method does not require re-training of a pre-trained model and the metric to merge tokens is already calculated as part of the forward pass. In this work, *we explore the implementation of token merging in the video domain without fine-tuning*, allowing tokens to merge freely in and between frames, to showcase its viability on video models that have been built upon ViT.

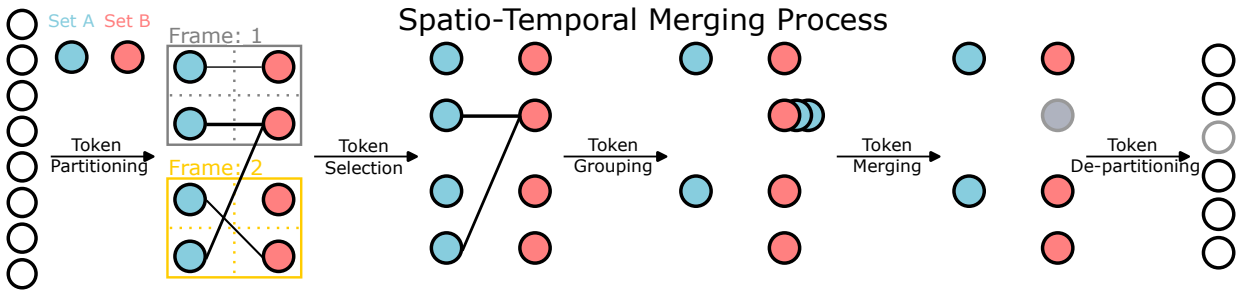


Figure 2. The merging process first separates tokens into two sets. Similarities are calculated and a one-to-many bipartite matching between tokens in each set is found. Finally, the top r edges are kept and these are merged based on the similarity between tokens.

We evaluate video token merging on Kinetics-400 [16], Something-Something v2 [12], and EPIC-KITCHENS-100 [9], which are datasets commonly used for video understanding, using both third and first person videos as well as containing actions with differing granularities.

To summarise, our contributions are as follows: (i) we explore best practices for training-free token merging for video; (ii) we provide a comprehensive evaluation of video transformer models on common action recognition benchmarks, demonstrating a training-free speedup of 2.5X; and (iii) we provide an in-depth analysis of how tokens are merged within spatio-temporal models.

2. Related Work

Efficiency of transformers is a popular topic, with a variety of possible approaches. Recent works in natural language processing (NLP) have focused on the creation of more computationally efficient attention modules to reduce the overhead of the quadratic complexity [32], attention modules that make better use of modern hardware [10], and new architectures that scale linearly with input size [14]. Others make use of domain specific knowledge to reduce the range of tokens that self-attention is applied to in vision problems [2, 26]. Alternatively, in [31], knowledge distillation is applied to make transformers more efficient learners at training time. Transformers are resilient to the dropping (or pruning) of input tokens, attention heads, and even entire blocks, which [25] has demonstrated. The dropout of input tokens is of particular interest for vision transformers because it exploits the data redundancy present in the natural image and video domain. Accordingly, we now introduce works whose focus is to reduce the token sequence length, whether by dropping them directly or merging them into more compact features.

2.1. Token Dropout

A simple method to reduce the sequence length in transformers is to drop tokens, i.e. token dropout. This has been applied to vision problems in multiple ways: tokens can be dropped at random [15, 23] or via a learned module [7, 27, 34]. In particular, [23] introduces a scheme whereby image classification training is expedited by applying random token dropout to vanilla ViTs, however, at inference time the whole token sequence is used. Similarly, [15] trains a partial masked auto-encoder for video understanding by dropping out significant portions of the input and then partially reconstructing the video. Dropout is well established in the literature, but emphasis has not been placed on the speedup of transformers at inference time *without* re-training.

2.2. Token Merging

Tokens have been merged in a variety of ways to reduce the information lost by dropout. In [22] innattentive tokens are fused into a single "background" token, while others exploit semantics to improve features of interest [35, 37]. Other works [20] optimise token merging through trainable parameters. Most recently, [19] trains models with an extra stream of compressed tokens to reduce attention overhead. Alternatively, there are many works that make use of token merging, as established in [3, 4], where image tokens are merged via a weighted average by reusing attention keys as a similarity metric. The primary motivation of the method is that it can be applied to ViTs *without re-training or additional learned modules*. This has been extended to VLMs [5, 8, 28, 33, 36], semantic segmentation [17], and video editing [21]. In [18], the merging method is improved to bridge the gap between dropout and merging. We note that prior works have not explored video token merging on challenging video datasets.

3. Method

Our implementation of spatio-temporal token merging for video transformers is an extension of image based token merging [4]. We first explain token merging for videos in Sec. 3.1 before describing how token merging can be applied to methods which explicitly incorporate attention across the temporal dimension in Sec. 3.2.

3.1. Spatio-Temporal Token Merging

The goal of token merging is to reduce the number of tokens within the model, which reduces the overhead of attention calculation between all pairs of tokens, increasing throughput of the method. Tokens are merged instead of dropping them, so that complementary information is preserved but redundant information is reduced, with the aim of preserving accuracy of the model. The merging process is applied within each layer in the transformer successively, to ensure that information is not lost too quickly. We give an overview of the process in Fig. 2 which is explained in greater depth below.

In detail, at the i^{th} layer of the transformer, we define the video token sequence as $x \in \mathbb{R}^{B \times S_i^V \times D}$, where B is the batch size, S_i^V is the current number of tokens in the video sequence for the i^{th} layer, and D is the channel depth of the features. We first *partition the tokens* x, into two sets $\mathbb{A}$ and $\mathbb{B}$, both of size $S_i^V/2$. The tokens are partitioned in an alternating manner, as is established in [4]. Next, we *select the most similar token* in $\mathbb{B}$ for every token in $\mathbb{A}$, creating a one-to-many matching between the two sets as a bipartite graph G. Note that these connections are drawn *across all* frames. The similarity is defined as the cosine distance between the keys (K) of two tokens from the Query-Key-Value self-attention and this is used as the edge weighting in G.

We define a hyperparameter for the number of tokens to merge as r_i, where $0 \leq r_i \leq S_i^V/2$, which represents the number of tokens to merge within the ith layer. Accordingly, r_i edges with the highest similarity (the edge weight) are kept and all other edges are dropped. The remaining edges denote *tokens to be merged* and their features are then averaged, weighted by their size. The size of each token is tracked as $n \in \mathbb{R}^{S_i^V}$, a vector where each value represents the total number of tokens that have been merged into each location in the sequence.

After the *tokens are de-partitioned*, each token in the sequence is now the weighted average of an arbitrary number of image patches. This has been demonstrated by [4] to affect the attention output of the next layer; when tokens with similar keys are merged, they'll have a smaller impact on the softmax output. To help alleviate this problem, *proportional attention* is introduced:

$$\text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}} + \log n\right) \tag{1}$$

ViViT [1] and VideoMAE [30] extend the ViT architecture [11] to incorporate the temporal dimension by jointly attending across space and time. There are few architectural differences between these video models and image ViTs, with the additional temporal dimension making them computationally expensive. To reduce this overhead, the patch embedding is 3D instead of 2D, meaning each token spans multiple frames.

3.2. Divided Space-Time Token Merging

The works of TimeSformer [2] and Motionformer [26] forgo joint space-time attention and instead experiment with methods to fuse information in a more efficient manner. However, this requires the token sequence to have temporal structure, as the temporal attentions are calculated through frames, using the spatial position as an anchor point. Because of this, merging tokens between frames causes temporal relationships to be broken and so we apply the token merging operation to each frame independently, as if they are unrelated images. Note, that these tokens still encode temporal information through the attention process. Accordingly, the hidden states are $x \in \mathbb{R}^{BF \times S_i^I \times D}$, where B is the batch size, F is the number of frames in the video, S_i^I is the current number of tokens in the image sequence for the i^{th} layer. To the best of our knowledge, we are the first work to implement token merging in this manner with divided space-time transformers.

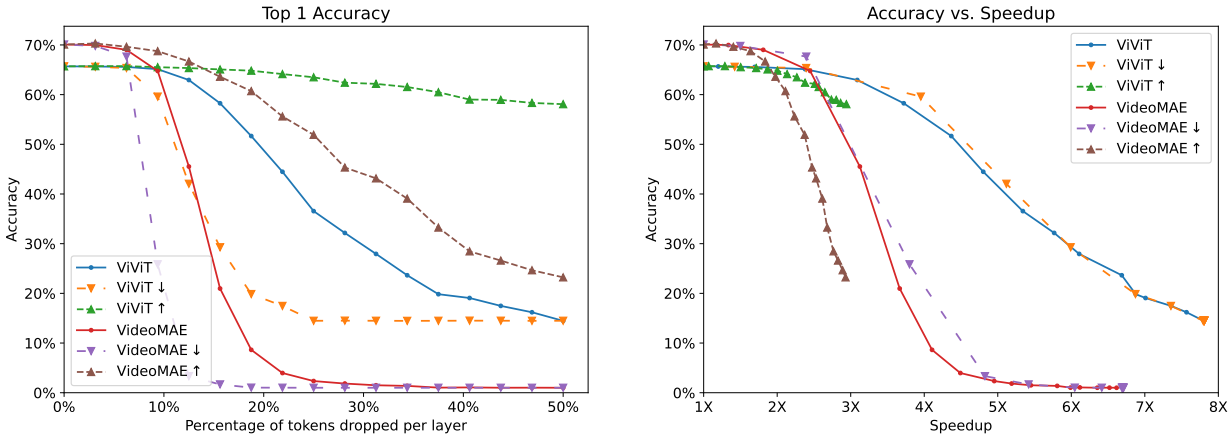
4. Results

Within this section we present comprehensive experiments of video token merging on different video transformer methods across different datasets. We aim to provide a consistent view of these methods and how the accuracy/speedup trade-off of each method can vary depending on the granularity of actions within each dataset.

4.1. Implementation Details

We test token merging in video across a selection of four different transformer models. The models are as follows: TimeSformer with 8 frames, Motionformer with 16 frames, VideoMAE with 16 frames and ViViT ("Model 1" from [1]) with 32 frames. In all of our experiments we use the base size models with default number of frames. This has the benefit of being a fair testbed whilst also evaluating token merging performance across a variable number of frames. Where existing checkpoints were not available online (EK-100 on TimeSformer, ViViT, and VideoMAE and SSv2 on ViViT), we finetune our own from existing K400 checkpoints on four H100 GPUs [24]. Proportional attention is used in all models except VideoMAE, due to the model being a masked autoencoder derivative [4]. For fair comparison, we use the same data loading/data augmentation techniques across all methods and each clip is only used for *one* view, in contrast with the multiple views often used in the original papers.

Merging Schedule. In practice, when r is mentioned in this section, we are referring to setting r_i for each layer, according to one of the following schedules: **Constant:** fixed r per layer, denoted as ViViT. **Decreasing:** $2r \rightarrow 0$ per layer, denoted as ViViT$\downarrow$. **Increasing:** $0 \rightarrow 2r$ per



(a) Accuracy as r increases.

(b) Accuracy against speedup

Figure 3. (Left) curves corresponds to accuracy with ViViT and VideoMAE on K400 when increasing r (the number of tokens merged) up to its limit. The x-axis is the percentage (relative to the original total) of tokens dropped *per layer*. (Right) figure displays the accuracy against speedup gained for these r values.

layer, denoted as ViViT↑. For the decreasing and increasing schedules, values are linearly interpolated between the two listed numbers. For the divided space-time models, the r value is a tuple of the r value for each frame and the effective number of frames in each clip.

Baselines. We re-implement previous methods for token merging and dropout with our combination of models and datasets as baselines to compare with spatio-temporal token merging. **Random dropout [23]:** Randomly drop tokens (instead of merging). **Dropout [4]:** Use attention metric to choose which tokens to drop (instead of merging). Finally, we also introduce a naive baseline to determine how important the attention metric is when merging: **Random merge:** Randomly merge tokens instead of using attention to choose.

4.2. Dataset Details

We evaluate on three action recognition datasets: Kinetics-400 (K400) [16], Something-Something v2 (SSv2) [12] and EPIC-KITCHENS-100 (EK-100) [9]. K400 is a coarse-grained action recognition dataset collected from YouTube, focusing on a large range of general human actions, typically from a third-person perspective. Comparatively, SSv2 is more fine-grained and studies interactions between hands and objects. Most notably, the action classes are object agnostic, to encourage understanding of temporal cues. Lastly, EK-100 consists of unscripted egocentric footage gathered in participants' kitchens, which introduces increased visual noise in the form of motion blur and occlusions. Actions are annotated with a "verb" and "noun" class, making the dataset considerably more fine-grained.

4.3. Quantitative Results

Scaling with r. We first investigate the trade-off between accuracy and throughput from token merging by varying r. The results can be seen in Fig. 3, which compares the Kinetics-400 accuracy and speedup gained for these schedules on ViViT and VideoMAE. Note that we plot r as a percentage of tokens dropped for easy comparison between the two models.

In Fig. 3a, looking at the constant schedule, we see that both ViViT and VideoMAE are resilient to merging up to 10% of their original tokens *per layer*, as after this point the accuracy (especially for VideoMAE) begins to decrease significantly. Nevertheless, this still represents dropping a total of 60% tokens throughout the entire network. At the other extreme, when r is 20% of the original tokens, VideoMAE sinks to almost random accuracy, while ViViT retains much more of its original performance. The increasing schedule displays a significantly slower drop in accuracy, while the decreasing schedule drops significantly faster becaused merging is loaded towards the front of the model. However, when using the decreasing schedule, there is still a period before r hits 10% of the tokens where the accuracy matches that of the more conservative schedules.

We compare model accuracy to the corresponding speedup gained via merging in Fig. 3b, where we see an elbow for each curve, indicating the points at which merging stops being economical in terms of throughput. The increasing schedules with loosely dashed lines are gathered together, showing little gain in speedup, while the tightly dashed lines for the decreasing schedule show quicker gains in speedup. The constant schedule is a middle-ground between the two, reaching the same end-

Model	r	Reduction	K400	SSv2	EK-100			FPS	Speedup (X)
					Action	Verb	Noun		
TimeSformer [2]	0	-	76.63	50.66	31.32	55.48	47.23	117.78	1.00
	18×8	random drop	65.58	17.18	12.78	34.03	28.31	240.13	2.04
		drop	68.30	22.97	16.09	38.24	33.02	240.13	2.04
		random merge	38.41	5.46	2.82	21.47	8.57	234.58	1.99
		merge	**71.14**	**25.11**	**18.59**	**40.47**	**35.73**	240.16	2.04
Motionformer [26]	0	-	70.50	61.39	35.02	61.09	46.72	99.79	1.00
	18×8	random drop	63.80	**24.50**	14.27	37.38	27.57	218.40	2.19
		drop	63.41	22.46	**15.92**	**39.54**	**30.60**	216.73	2.17
		random merge	49.30	17.76	7.88	31.56	16.77	210.30	2.11
		merge	**65.05**	24.10	15.60	38.20	30.15	218.11	2.19
VideoMAE [30]	0	-	62.09	64.58	35.70	61.49	46.89	186.72	1.00
	150	random drop	55.02	57.29	28.53	55.45	39.45	481.45	2.58
		drop	**56.65**	60.33	31.02	57.70	**42.43**	483.04	2.59
		random merge	20.64	22.89	5.44	26.86	10.07	471.74	2.53
		merge	56.10	**61.10**	**31.27**	**58.00**	42.39	476.28	2.55
ViViT [1]	0	-	63.43	50.63	35.82	58.19	51.59	106.00	1.00
	300	random drop	59.95	46.71	30.36	54.24	45.51	262.04	2.47
		drop	58.00	45.36	30.12	53.20	46.90	262.34	2.47
		random merge	28.88	19.15	5.78	28.88	14.82	259.92	2.45
		merge	**63.08**	**50.15**	**35.11**	**57.24**	**51.33**	260.72	2.46

Table 1. Performance of token merging with a constant schedule when compared to alternative methods of reducing token sequence length. Bold indicates the reduction methods that achieve highest accuracy on a given dataset. Grey rows correspond to the upper bound accuracy of the original model.

point as the decreasing schedule at a slower rate. Finally, in terms of picking optimal r and schedule, the point at each elbow indicates that for the constant and decreasing schedules, merging 10% of the original tokens can produce speedups of roughly 2.5X and 4X respectively.

From inspection of these results, we select r for all other experiments such that 10% of tokens are dropped per layer, equating to 60% of all tokens across all layers in the network being merged. This value gives a good trade-off between retaining accuracy but increasing throughput of the methods.

Comparison with token reduction methods. In Tab. 1, we show results across all models *without* any reduction and *with* reduction at a specified r value, when using a constant schedule. We compare upper bound accuracy (in grey) with four reduction methods which don't require re-training, including our spatio-temporal merging.

Firstly, we investigate the importance of the merging aspect by comparing between reduction methods. For all methods but Motionformer, token merging remains able to produce the highest accuracies when reducing token sequences. On all datasets tested, ViViT gains at least 4% over dropout that uses the same attention metric. For VideoMAE, the gains are less significant, yet consistently outperform other methods on SSv2 and EK-100. An interesting result is that randomly merging tokens is significantly destructive to performance, being noticeably worse than random dropout. In particular, on EK-100, the verb accuracy tends to halve, while the noun accuracy roughly quarters. The poor performance can be explained by the fact that potentially dissimilar tokens are having their features averaged together, creating a more negative effect than averaging the features of tokens that the model already determines to be similar—in this case dropping the tokens altogether is preferable. We note that there are not significant differences between the throughputs of these variations.

Secondly, we directly compare the upper bound model performance to our merged implementations. For the divided space-time models, accuracy on K400 drops by roughly 5%, while performance on SSv2 and EK-100 drops by much larger margins. In particular, the accuracies on SSv2 are roughly half what the vanilla model can achieve even with a speedup of just over 2X. Both EK-100 and SSv2 are more fine-grained and reliant on temporal reasoning, *token merging is impairing the models' ability to fuse information across frames*, while high accuracy can be attained on K400 with spatial reasoning.

We find that spatio-temporal models are much more resilient to merging. VideoMAE sees losses of around 4% across all datasets, though it retains strong accuracy on SSv2 and EK-100, either outperforming or being comparable to $r = 0$ TimeSformer and Motionformer.

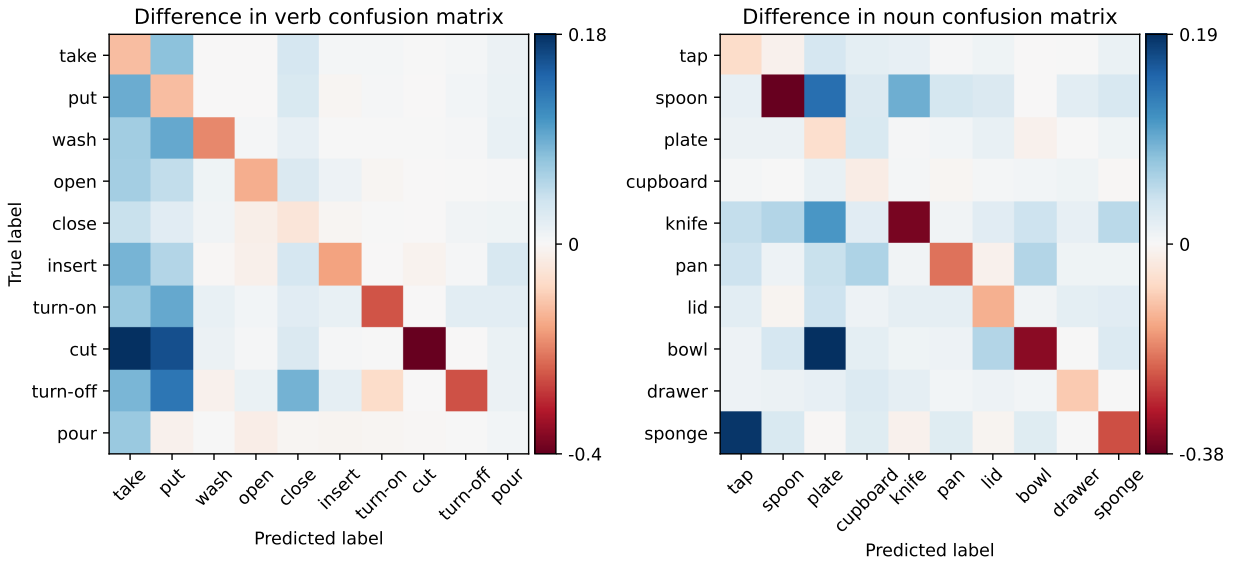
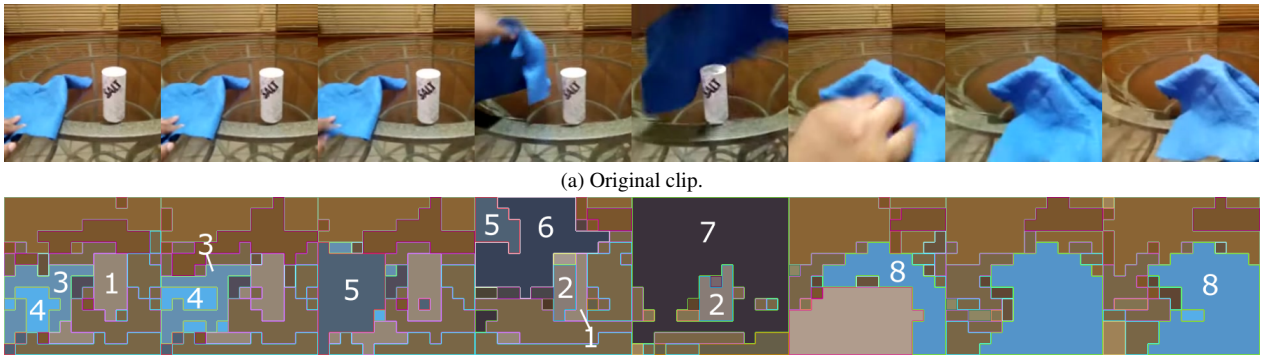


Figure 4. Impact on confusion matrices from Token Merging a VideoMAE model. The first 10 verb and noun classes are displayed left and right respectively from VideoMAE on EK-100. Red indicates less predictions and blue indicates more predictions.



(a) Original clip.

(b) Merged clip.

Figure 5. Visualisation of the final merged tokens for an SSv2 clip of "covering salt shaker with a towel", produced with VideoMAE. Tokens 1 and 2 capture the white salt shaker. The model struggles more with the blue towel, with it splitting into tokens 3 – 8.

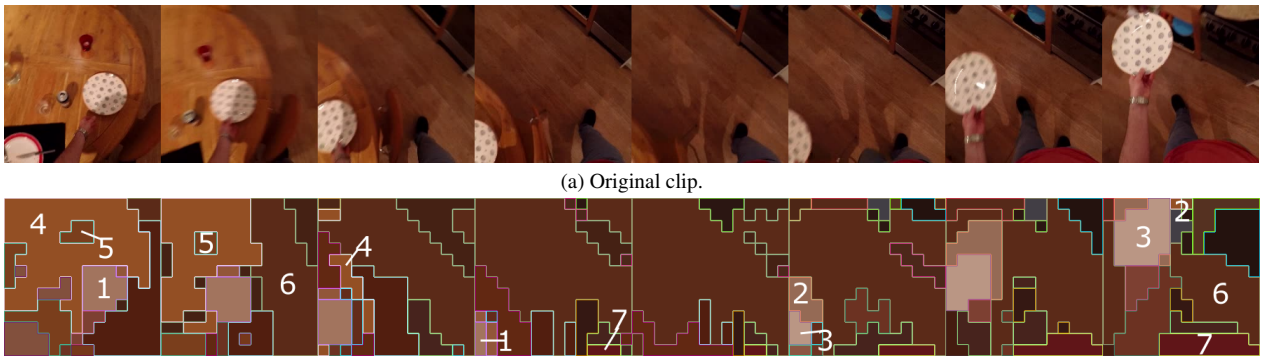
ViViT with token merging *performs extremely well on all datasets*, never dropping by more than 1%. Both models see speedups of around 2.5X, with ViViT reaching comparable performance, *increasing the inference throughput for free*. While ViViT and VideoMAE have a similar structure, there are a few differences that could account for the disparity: ViViT uses twice as many input frames whereas VideoMAE does not use a class token and is pretrained on masked input.

Comparison of class confusion. We investigate the errors introduced from merging VideoMAE and produce confusion matrices for the most frequent 10 verb and noun classes in EK-100. Figure 4 shows the difference in performance between a model *with* and a model *without* token merging. In Fig. 4 (left) we can see that when a high r value is used, the merged model becomes more likely to predict the highly common "take" and "put" classes, exacerbating the errors of the original model.

The nouns in Fig. 4 (right) are more resilient to this collapse into the most common classes. However, we can see that certain related nouns are increasingly being misclassified, particularly those that relate to smaller objects. For example, "spoon" and "knife" are more frequently misclassified as the wrong piece of cutlery, or the action-related "plate". Similarly, "bowl" is mistaken for "plate" and "sponge" is mistaken for "tap". Larger objects like "drawer", "tap", and "cupboard"—which occupy portions of the background—are broadly unaffected by the merging, suggesting that the features of smaller objects are more likely to be lost by merging.

4.4. Qualitative Results

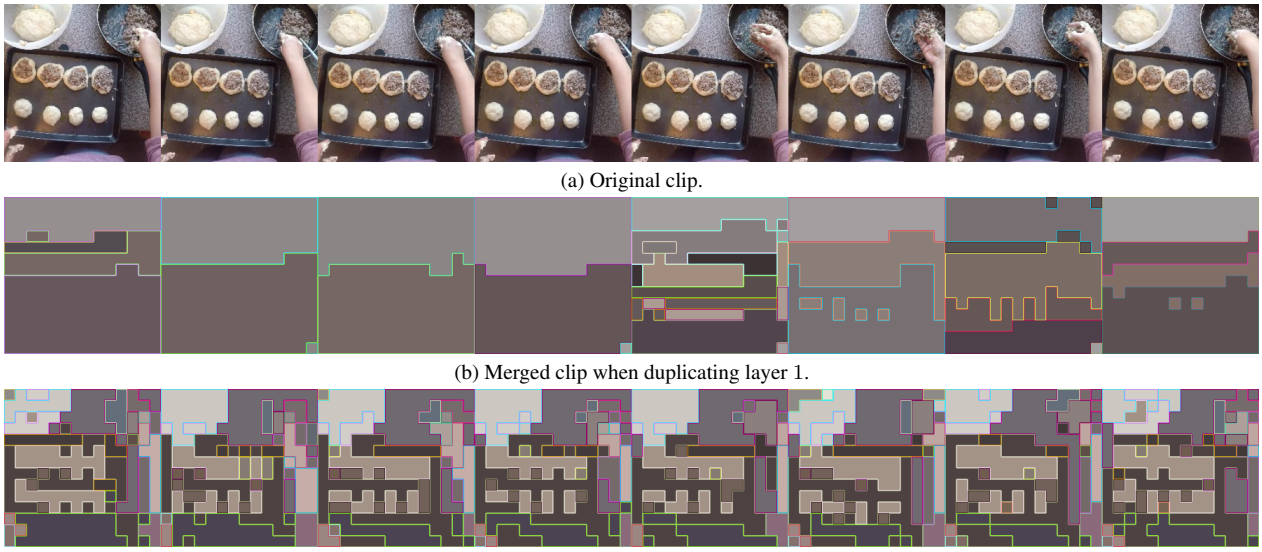
In this section, we explore the token merging from a visual standpoint, answering the following questions: i)



(a) Original clip.

(b) Merged clip.

Figure 6. Visualisation of the final merged tokens for an EK-100 clip of "take plate", produced with VideoMAE. The first 4 frames, merge the plate into token 1, even with motion blur whereas in the last 3 frames it splits into tokens 2 and 3.



(a) Original clip.

(b) Merged clip when duplicating layer 1.

(c) Merged clip when duplicating layer 12.

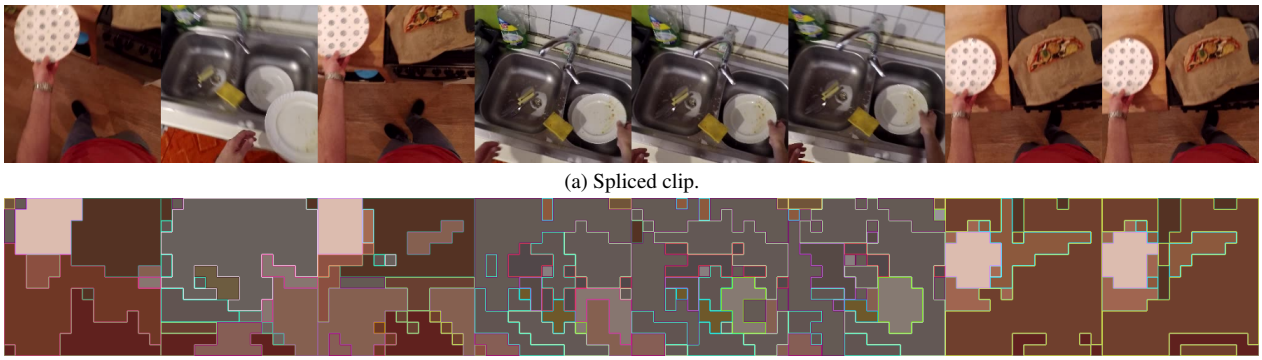
Figure 7. Visualisations of the difference in merging decisions made in layer 1 versus layer 12, produced with VideoMAE. We find that the final layer merges the tokens in a more object focused manner vs. the first layer.

How are tokens merged in video?; ii) How do different layers merge tokens?; and iii) Are tokens being merged semantically or visually? For the visualisations, we extended the implementation in [4] for video—tokens with the same background and edge colour are the same. We number the first and last appearance in each visualisation where specific tokens are referred to in text.

How are tokens merged in video? The action "covering salt shaker with a towel" from SSv2 is depicted in Fig. 5. While it is visible, the white salt shaker is captured well by token 1 and 2, likely split due to the shadow of the towel covering the salt shaker. However, the model struggles with the blue towel, leaving tokens 3–8 unmerged. Tokens 4 and 5 are significantly darker as the towel's underside is not hit by a light source, increasing the visual differences between the tokens, suggesting that tokens are not merged semantically and visual differences will prevent merging across frames.

Next, we show an example from EK-100 of "take plate" in Fig. 6. In the first four frames, the plate is merged to token 1, even with the rapid motion blur. However, when it's back in view in the last three frames it splits into tokens 2 and 3. Additionally, the table and a glass are clearly grouped separately into tokens 4 and 5 respectively. The background is surprisingly well distinguished from the foreground in token 6, even with the unusual head pose, likely because the floor has one common texture. Interestingly, the t-shirt of the participant is visibly grouped into token 7. These examples showcase how tokens are merged effectively spatio-temporally without any learned parameters/fine-tuning.

How do different layers merge tokens? Because the



(a) Spliced clip.

(b) Merged clip.

Figure 8. Merging outcome for a clip that has had half its frames from the most "similar" clip in the same noun class spliced in, produced with VideoMAE. The plates are merged consistently together within each clip but are not merged across the clips.

merging method happens iteratively as the token sequence passes through transformer layers, the outcome is cumulative and biased towards earlier layers. To determine how different transformer layers merge tokens, we conduct experiments where layer i of the model is duplicated 12 times (stripping out everything but the merging modules) and tokens are only merged by instances of layer i. Figure 7 visualises an example from EK-100. In Fig. 7b we see that duplicating the first layer produces merged results that are drastically different compared to earlier examples. We speculate that this layer is merging based on rudimentary shape or basic foreground/background colours. In Fig. 7c, we can see that duplicating the final layer produces similar output to a visual segementation of the clip. The food, tray, bowl, and background are clearly separated consistently across frames. Due to the cumulative nature of the layer based merging, there is currently no simple method to balance the impact that different layers have. However, *biasing merging towards the later layers significantly decreases the gains in throughput* of the model.

Are tokens being merged semantically or visually? Finally, we test whether the models are merged semantically, as much of what we have demonstrated is possible with visual similarity. We do this by taking a video and inserting clips from a different video from the same class, i.e. a clip with the same semantic meaning and objects and check how the tokens are merged spatially and temporally. We find videos that the model believes to be semantically similar by comparing the noun logits of all clips in the EK-100 test set using the Kullback–Leibler (KL) divergence between all possible pairs, and select the nearest video. Figure 8 shows an example of the most "similar" corresponding clip in the same noun class, randomly splicing frames into the source video. From the example, we can see that the model merges the original clip's plate into a single token, which notably does not overlap with any of the second clip's corresponding plate tokens. Even though the objects are in the same class, the difference in lighting and texture is enough to stop their tokens from merging, suggesting that *the merging process is almost entirely visual*. We show more examples within the supplementary material, but only a tiny portion of examples suggest that tokens are being merged with any semantic reasoning. We also find that the background is only merged between the original and spliced frames if the kitchen is visually similar enough or the light levels are similar, indicating that the merging is not explicitly considering foreground and background.

5. Conclusion

The efficiency of transformer models remains an ongoing cornerstone of research in deep learning. In this work, we implemented video token merging for two spatio-temporal vision transformers and two divided space-time transformers to see how effectively the method can be dropped into existing models without retraining. As well as this, we created a testbed to compare token merging across video transformer models.

Using this testbed, we collate various quantitative and qualitative findings, with the aim of exploring how and why tokens are merged in the temporal dimension, as well as the effect that this has on action recognition with fine-grained video datasets. We have determined the margin by which attention driven merging beats dropout alternatives. Furthermore, we have demonstrated that this method of merging has little consideration for the semantics in the inputs that it merges and that the merging decisions made by different layers can vary wildly. The token merging methodology remains a convenient drop-in for vision transformers to increase throughput with a small drop in performance based on the ratio of dropped tokens. We hope that with these results pave the way for token merging to be directly incorporated into future video transformer models.

Acknowledgements

Research supported by EPSRC Doctoral Training Partnerships (DTP). The authors would like to thank Siddhant Bansal, Prajwal Gatti and Toby Perrett for their comments on the paper. The authors acknowledge the use of resources provided by the Isambard-AI National AI Research Resource (AIRR). Isambard-AI is operated by the University of Bristol and is funded by the UK Government's Department for Science, Innovation and Technology (DSIT) via UK Research and Innovation; and the Science and Technology Facilities Council [ST/AIRR/I-A-I/1023].

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 3, 5

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 2, 3, 5

[3] Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4599–4603, 2023. 2

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your ViT but faster. In *International Conference on Learning Representations*, 2023. 1, 2, 3, 4, 7

[5] Qingqing Cao, Bhargavi Paranjape, and Hannaneh Hajishirzi. Pumer: Pruning and merging tokens for efficient vision language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12890–12903, 2023. 2

[6] João Carreira, Michael King, Viorica Patraucean, Dilara Gokay, Catalin Ionescu, Yi Yang, Daniel Zoran, Joseph Heyward, Carl Doersch, Yusuf Aytar, et al. Learning from one continuous video stream. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 28751–28761, 2024. 1

[7] Lei Chen, Zhan Tong, Yibing Song, Gangshan Wu, and Limin Wang. Efficient video action detection with token dropout and context refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10388–10399, 2023. 2

[8] Joonmyung Choi, Sanghyeok Lee, Jaewon Chu, Minhyuk Choi, and Hyunwoo J Kim. vid-tldr: Training free token merging for light-weight video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18771–18781, 2024. 2

[9] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Evangelos Kazakos, Jian Ma, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision*, pages 1–23, 2022. 1, 2, 4

[10] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022. 2

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. 1, 3

[12] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017. 2, 4

[13] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu 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*, pages 18995–19012, 2022. 1

[14] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 2

[15] Tengda Han, Weidi Xie, and Andrew Zisserman. Turbo training with token dropout. *arXiv preprint arXiv:2210.04889*, 2022. 2

[16] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 4

[17] Daniel Kienzle, Marco Kantonis, Robin Schön, and Rainer Lienhart. Segformer++: Efficient token-merging strategies for high-resolution semantic segmentation. *arXiv preprint arXiv:2405.14467*, 2024. 2

[18] Minchul Kim, Shangqian Gao, Yen-Chang Hsu, Yilin Shen, and Hongxia Jin. Token fusion: Bridging the gap between token pruning and token merging. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1383–1392, 2024. 2

[19] Rajat Koner, Gagan Jain, Prateek Jain, Volker Tresp, and Sujoy Paul. Lookupvit: Compressing visual information to a limited number of tokens. *arXiv preprint arXiv:2407.12753*, 2024. 2

[20] Seon-Ho Lee, Jue Wang, Zhikang Zhang, David Fan, and Xinyu Li. Video token merging for long video understanding. *Advances in Neural Information Processing Systems*, 37:13851–13871, 2024. 2

[21] Xirui Li, Chao Ma, Xiaokang Yang, and Ming-Hsuan Yang. Vidtome: Video token merging for zero-shot video editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7486–7495, 2024. 2

[22] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are what you need: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022. 2

[23] Yue Liu, Christos Matsoukas, Fredrik Strand, Hossein Azizpour, and Kevin Smith. Patchdropout: Economizing vision transformers using patch dropout. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3953–3962, 2023. 2, 4

[24] Simon McIntosh-Smith, Sadaf R Alam, and Christopher Woods. Isambard-ai: a leadership class supercomputer optimised specifically for artificial intelligence, 2024. 3

[25] Lingchen Meng, Hengduo Li, Bor-Chun Chen, Shiyi Lan, Zuxuan Wu, Yu-Gang Jiang, and Ser-Nam Lim. Adavit: Adaptive vision transformers for efficient image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12309–12318, 2022. 2

[26] Mandela Patrick, Dylan Campbell, Yuki Asano, Ishan Misra, Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and Joao F Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. *Advances in neural information processing systems*, 34: 12493–12506, 2021. 2, 3, 5

[27] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. *Advances in neural information processing systems*, 34: 13937–13949, 2021. 2

[28] Shuhuai Ren, Sishuo Chen, Shicheng Li, Xu Sun, and Lu Hou. Testa: Temporal-spatial token aggregation for long-form video-language understanding. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. 2

[29] Mattia Soldan, Alejandro Pardo, Juan León Alcázar, Fabian Caba, Chen Zhao, Silvio Giancola, and Bernard Ghanem. Mad: A scalable dataset for language grounding in videos from movie audio descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5026–5035, 2022. 1

[30] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022. 3, 5

[31] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021. 2

[32] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 2

[33] Yuetian Weng, Mingfei Han, Haoyu He, Xiaojun Chang, and Bohan Zhuang. Longvlm: Efficient long video understanding via large language models. In *European Conference on Computer Vision*, pages 453–470. Springer, 2024. 2

[34] Hongxu Yin, Arash Vahdat, Jose M Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10809–10818, 2022. 2

[35] Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11101–11111, 2022. 2

[36] Yiwu Zhong, Zhuoming Liu, Yin Li, and Liwei Wang. Aim: Adaptive inference of multi-modal llms via token merging and pruning. *arXiv preprint arXiv:2412.03248*, 2024. 2

[37] Xingyi Zhou, Anurag Arnab, Chen Sun, and Cordelia Schmid. How can objects help action recognition? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2353–2362, 2023. 2