Learnable Pooling Methods for Video Classification

Sebastian Kmiec[0000−0003−3116−5302], Juhan Bae[0000−0001−7600−9560], and Ruijian An[0000−0002−2102−0209]

University of Toronto, Toronto, Canada
{sebastian.kmiec, juhan.bae, ruijian.an}@mail.utoronto.ca

Abstract. We introduce modifications to state-of-the-art approaches to aggregating local video descriptors by using attention mechanisms and function approximations. Rather than using ensembles of existing architectures, we provide an insight on creating new architectures. We demonstrate our solutions in the "The 2nd YouTube-8M Video Understanding Challenge", by using frame-level video and audio descriptors. We obtain testing accuracy similar to the state of the art, while meeting budget constraints, and touch upon strategies to improve the state of the art. Model implementations are available in https://github.com/pomonam/LearnablePoolingMethods.

Keywords: Video Classification · Youtube-8M · NetVLAD · Attention · Pooling · Aggregation

1 Introduction

The problem of summarizing local descriptors is highly investigated and encompasses many domains in machine learning such as image retrieval. The goal of aggregating local descriptors is to construct a single global descriptor that encodes useful information. Although progress exists in the context of local video descriptor aggregation [2], [12], few works adequately provide solutions for differentiable descriptor aggregation, such as NetBoW [12] and SMK [14], [16].

Aggregation of local descriptors extends to the task of video classification. Many existing models focus on learning temporal relations within a video. In particular, recurrent neural networks (RNN) with the help of Long Short Term Memory (LSTM) or gated rectified units (GRU) can capture the long-term temporal patterns in between frames. In this paper, however, we question the usefulness of learning temporal relationships in video classification. This is also largely motivated by the work of NetVLAD [2] and attention clusters [11], producing a dominant result compared to recurrent methods [18], [3].

Despite the success of NetVLAD for the task of video understanding [12], many design choices are left unexplained. For example, the individual contribution of cluster center residuals to the global representation is unclear. The model also exhibits several weaknesses. For instance, the global representation generated by NetVLAD is projected to a significantly lower dimensional space

for classification, which we suspect is difficult for a single layer to manipulate. To accommodate these issues, inspired by [5], [17], we propose a model capable of learning inter-feature relationships before and after the NetVLAD layer.

2 Related Work

In this section, we outline previous works that heavily influenced the creation of our learnable pooling architectures

2.1 Attention Mechanisms

Our work is largely inspired by [11], where for a given set of local descriptors, an attention representation is created via a simple weighted sum of local descriptors. These weights are computed as a function of the local descriptors to exclusively obtain information from useful descriptors in the attention representation. This attention representation is termed an attention cluster, and multiple clusters are concatenated to form a final global representation. A novel shifting operation, as in Equation (1), enables each attention cluster to diverge from each other, while keeping scale invariance. Below is the output of a single attention cluster; $\mathbf{X}$ is a matrix of local descriptors, $\mathbf{a}$ is a vector of attention weights, computed via a simple two layer, feed-forward network.

$$\psi_k(X) = \frac{\alpha \cdot \mathbf{aX} + \beta}{\sqrt{N}\left\|\alpha \cdot \mathbf{aX} + \beta\right\|_2} \tag{1}$$

Although the above mechanism has the capacity to learn the training data, the model heavily overfits, and fails to generalize well. We hypothesize that as you are computing a weighted sum of input descriptors, slight changes in the distribution of inputs will have a large negative impact on performance. Intuitively, to avoid this problem, internal embeddings should be summed instead, and these internal embeddings should be a function of the inputs, to improve generalization.

Following the above logic is the work by A Vaswani et al [17], whom achieved state-of-the-art performance for machine translation tasks, using a novel attention mechanism termed Transformer. In the context of their work, local embeddings are projected to query, key and value spaces. The similarity of keys ($\mathbf{K}$) to queries ($\mathbf{Q}$) is then used to provide weights to the internal embedding vectors (as in Equation (2)) that are passed to feed forward networks. Both [11] and [17] use dot product attention for attention-based representations. However, attention clusters intend to provide a straight-forward global representation, whereas a Transformer creates a new attention-based encoding of equal dimensionality. Our use of Transformers is covered in detail within Section 3.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{\mathbf{QK}^T}{\alpha})\mathbf{V} \tag{2}$$

2.2 Differentiable Pooling

The work of [12] provides a useful baseline for a video classification model, as it provides the highest known accuracy of any single architecture, as of the "Google Cloud & YouTube-8M Video Understanding Challenge". The authors utilized NetVLAD [2], along with a novel network block that is comparable to residual blocks [7], known as gating. Gating has the effect of re-weighting the importance of features detected, and decisions made based on these features.

VLAD itself summarizes a set of local descriptors by presenting these descriptors via a distribution [8]. This distribution is encoded in the sum of distances to cluster centres. Specifically, looking at Equation (3), a_j refers to the cluster soft similarity of descriptor x_k to the jth cluster center, introduced by NetVLAD to avoid non-differentiable hard assignments. Here, $\boldsymbol{x}_k - \boldsymbol{c}_j$ refers to a single distance of a local descriptor to a cluster centre. This is a sound technique that is used in many areas [6], [20], while achieving state of the art performance in the benchmarks such as [15].

$$VLAD(i,j) = \sum_{k=1}^{N} a_j(x_k)\,(x_k(i) - c_j(i)) \tag{3}$$

Despite the success and prominence of VLAD, numerous design choices in [12] are left unexplained. For instance, the cluster similarities, a_j, are computed via a simple linear transformation, where each local video descriptor is compared via dot product with a key per cluster centre, followed by a softmax layer. It is a straightforward idea to consider multiple keys per cluster centre, or to use multiple temporally close local descriptors per cluster similarity prediction.

Further on the notion of design choices, another weakness arises in the architecture after the NetVLAD block. Aside from the use of gating, all outputs of the NetVLAD module are simply squeezed and/or projected to a low dimensional space for classification, this is too demanding of a task for a single layer to perform optimally. A final criticism is that this model does not attempt to leverage inter-feature relationships prior to the use of the NetVLAD module.

2.3 Regularized Aggregation

Following the success of VLAD is the work of [9]. The authors split the problem of local descriptor pooling into problems of local descriptor embedding and aggregation. We specifically utilize the ideas for local descriptor embeddings. The work of [9] creates a new embedding technique by focusing on overcoming pitfalls when using NetVLAD, by L2 normalizing the distances from cluster centers (residuals) before summing to help give an equal contribution to each residual. These embeddings are known as Triangulation embeddings, or T-embeddings. Further, the authors suggest whitening the residuals by removing a bias and decorrelating the residuals per cluster center, for a given local descriptor, before summing.

We do utilize the above two ideas. However, the suggestion of using democratic weights before summing [9] is avoided. The use of democratic weights is intended to give each local descriptor an equal contribution to the similarity of the class they belong to. Nevertheless, it is not clear how to make an easily parallelizable implementation of the Sinkhorn scaling algorithm to obtain a solution for these weights. The use of weights is described in Equation (4), where $\boldsymbol{X}$ is the set of local descriptors belonging to a given class, and $\boldsymbol{\phi}_i$ is an embedding per local descriptor, as displayed in Equation (5). In Equation (6), $\boldsymbol{\Sigma}$ is the covariance matrix of $\mathbf{R(X)}$, where $\mathbf{R(X)}$ is a random variable representing $\boldsymbol{R}(x_i)$ in the set of $\mathbf{X}$, per class. A single $\boldsymbol{R}(x_i)$ is the concatenation of normalized residuals to K cluster centers for a single local descriptor, as in (6).

$$\boldsymbol{\psi}(\boldsymbol{X}) = \sum_{i=1}^{N} \lambda_i \boldsymbol{\phi}_i \tag{4}$$

$$\boldsymbol{\phi}_i(\boldsymbol{x}_i) = \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{R}(\boldsymbol{x}_i) - E[\boldsymbol{R}(\boldsymbol{X})]) \tag{5}$$

$$\boldsymbol{R(x_i)} = [\boldsymbol{r}_1(\boldsymbol{x}_i), \ldots, \boldsymbol{r}_K(\boldsymbol{x}_i)] \tag{6}$$

2.4 Function Approximations

The work of [5] builds upon [19], which intends to provide an encoding per image, for the sake of image classification, using a weighted sum of local tangents at anchor points. Despite the dissimilarity of use cases for VLAD and tangent encoding, the authors of [5] provide a mathematical formulation, to describe how given a certain set of cluster similarity weights, tangent encoding is a generalization of VLAD.

As tangent encoding is a technique to linearly approximate a high dimensional function [5], the authors naturally extend VLAD to a second order approximation to obtain a unique local descriptor embedding, while incorporating ideas from other methods such as [9] for aggregation. An embedding per local descriptor is described in Equation (7), where $\boldsymbol{\phi}_i(x_i)$ is the concatenation of three vectors, per cluster center j. The $a_j(\boldsymbol{x}_i)$ can be thought of as cluster similarities, as in Equation (3).

What is newly introduced to VLAD is the flattened vector of $a_j(\boldsymbol{x}_i) \cdot F(\boldsymbol{v}_{i,j}\boldsymbol{v}_{i,j}^T)$ (in this equation $F(\cdot)$ is the flattening operation), this provides the second order approximation of our hypothetical function, derived from a Taylor expansion [5]. In addition, objective functions are provided to help regularize the learning of weights, so as to ensure a valid function approximation, this is further discussed in Section 3.3.

$$\boldsymbol{\phi}_i(\boldsymbol{x}_i) = [a_j(\boldsymbol{x}_i);\ a_j(\boldsymbol{x}_i) \cdot \boldsymbol{v}_{i,j};\ a_j(\boldsymbol{x}_i) \cdot F(\boldsymbol{v}_{i,j}\boldsymbol{v}_{i,j}^T)]_{j=1}^{K} \tag{7}$$

$$\boldsymbol{v}_{i,j} = (\boldsymbol{x}_i - \boldsymbol{c}_j) \tag{8}$$

3 Learnable Pooling Architectures

In this section, we describe the architecture of our proposed learnable, pooling methods, for the purpose of video classification. As inputs to our pooling methods, we have audio and video features already extracted at the frame level per second of video, from the "YouTube-8M Dataset" [1]. Our pooling methods aggregate all local descriptors per frame into a single global representation that describes a video, with possible post-processing afterwards. The final global video descriptor is then passed to a Mixture-of-Experts (MoE) [13] for classification, where probabilities are output across possible video labels.

3.1 Attention Enhanced NetVLAD

Our first approach is to use a transformer encoder before and after a NetVLAD module, as in Figure 1. Our local descriptor pooling is based largely on the work of [12]. As motivated in Section 2.2, NetVLAD with context gating is the current (completely differentiable) state of the art for video pooling, as per benchmarks [15], and the result of the "Google Cloud & YouTube-8M Video Understanding Challenge". Similarly, we have already motivated the use of Transformers in Section 2.1.

The first block is a mapping of $f_1 : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{N \times F}$, where N is the number of frame features sampled, and F is the feature size. The first transformer operates on the level of frames. As we uniformly sample frames per video as input, information relating to the relative positions of local descriptors inherently exists within the attention mechanism.

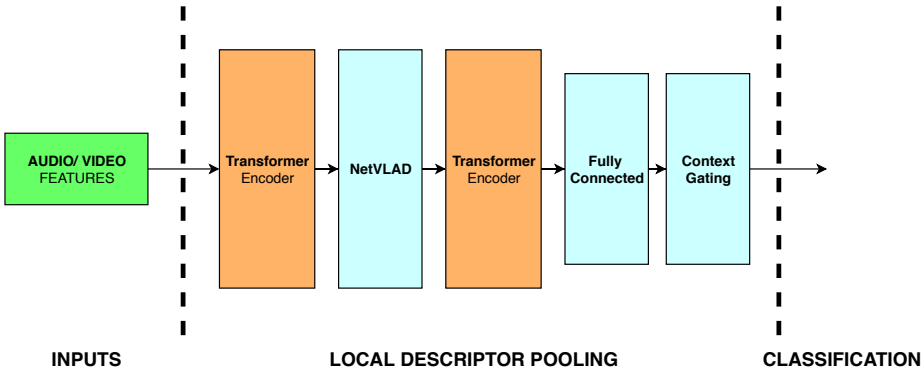

Fig. 1. A block diagram of an attention enhanced NetVLAD model. A modified Transformer Encoder [17] is placed before and after the NetVLAD module.

The second block is a mapping $f_2 : \mathbb{R}^{C \times F} \rightarrow \mathbb{R}^{C \times F}$, where C is the number of NetVLAD cluster centers. The second block operates on the level of clusters.

Our belief is that using separate query/key/value projection per cluster (as per the Transformer encoder), should be easier to learn than a single fully connected layer, that must perform the function of attention, decision making and dimensionality reduction all at once. We effectively spread the responsibility of these crucial functions across multiple layers, increasing the capacity of the initial work of [12].

The Transformer Encoder in Figure 1 refers to the work of [17], with batch normalization added to inner layers within the Multi-Head Attention block, to make learning a simpler process.

3.2 NetVLAD with Attention Based Cluster Similarities

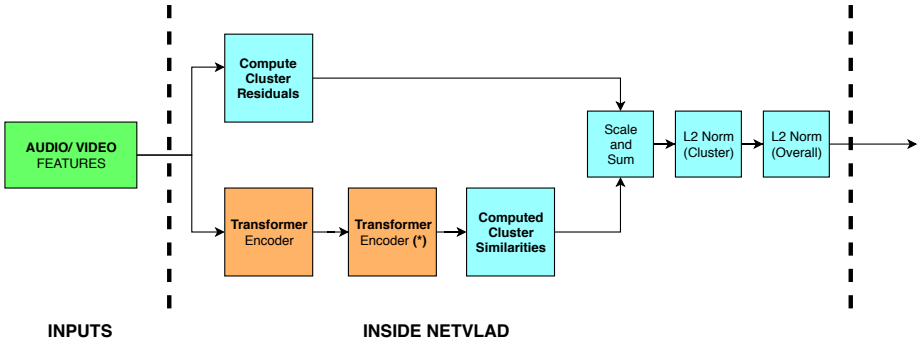

Fig. 2. A block diagram of a modified NetVLAD module. A pair of modified Transformer Encoders [17] is used to compute cluster similarities.

Along with the aforementioned Transformer based model, we propose the following model, using modified NetVLAD modules. As in Figure 2, the first Transformer encoder is a mapping of $g_1 : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{N \times F}$, whereas the second encoder is modified to be a mapping from $g_2 : \mathbb{R}^{N \times F} \rightarrow \mathbb{R}^{C \times F}$.

Similar to Section 3.1, we argue that determining cluster similarities per local descriptor via a simple dot product with one key per cluster, followed by a softmax, is too demanding of a task (although it is simple to learn). Furthermore, cluster similarity is a task well suited for a Transformer-based attention mechanism, as the goal is simply to remember what input descriptors are highly correlated to (or similar) to which cluster centers, by using keys that are dependent on the inputs themselves. In addition, by using a Transformer, our prediction of input descriptor similarity to a cluster now receives information from other input descriptors, further improving the capacity of the initial work of [12].

The Transformer Encoder (*) in Figure 2 is similar to the Transformer encoder discussed in Section 3.1, however, the final feed forward layer projects to

a dimension of size C instead of F, and hence, the final residual connection is removed.

3.3 Regularized Function Approximation Approach

Given our heavy reliance on NetVLAD [2] in our previous two models, our final model attempts to address issues that lie within NetVLAD and adaptations. The work of [9] already provides useful suggestions regarding possible pitfalls when using NetVLAD, by L2 normalizing the distances from cluster centers (residuals) before summing, as well as whitening these residuals. Unfortunately, we do not make use of democratic weights for aggregation of intermediate T-embeddings, as proposed in [9], as even simplified versions of the Sinkhorn scaling algorithm (with assumptions made) are slow to train with.

To account for the discriminative power lost by missing democratic weights, we unite the works of [9], [5] and [19], as illustrated in Figure 3. By adding second order terms, we now introduce more useful information in our global representation by adding multiplicative residual terms. To the point, these multiplicative terms cannot be computed by a simple linear transformation that follows our global representation, as is the case in [12].

Notice that for the second order terms, we first project the inputs to a lower space, as the second order cluster residuals are elements of $\mathbb{R}^{B\times N\times C\times F\times F}$, where B is the batch size, N is the number of local descriptors per video, C is the number of clusters and F is the input feature size. This is too large to fit even on multiple high-end commercial GPUs, given a feature size of 1024 (video features). Also note that as we perform a simple linear projection, as well as separate cluster centre and similarity computations for these squeezed features, it is expected that performance is to be lost.

Furthermore, we do not utilize the suggested regularizer terms within cost functions, as in [5]. The reason being that regularization requires tuning in order to provide a valid contribution. Given the large amount of time required to train this model, we avoid additional regularization terms. Although regularizer terms such as in [5], or in [4] (for the sake of cluster center sparsity), may be necessary to overcome generalization issues discussed in Section 4.

4 Experiments

Herein, we provide implementation details for experiments performed on models found in Section 3. All of the aforementioned models can be found in https://github.com/pomonam/LearnablePoolingMethods, with complete documentation and easily usable modules.

4.1 Training Details

The Youtube-8M dataset is split into training (70%), validation (20%) and testing (10%). For the sake of accuracy, we utilize both training and validation

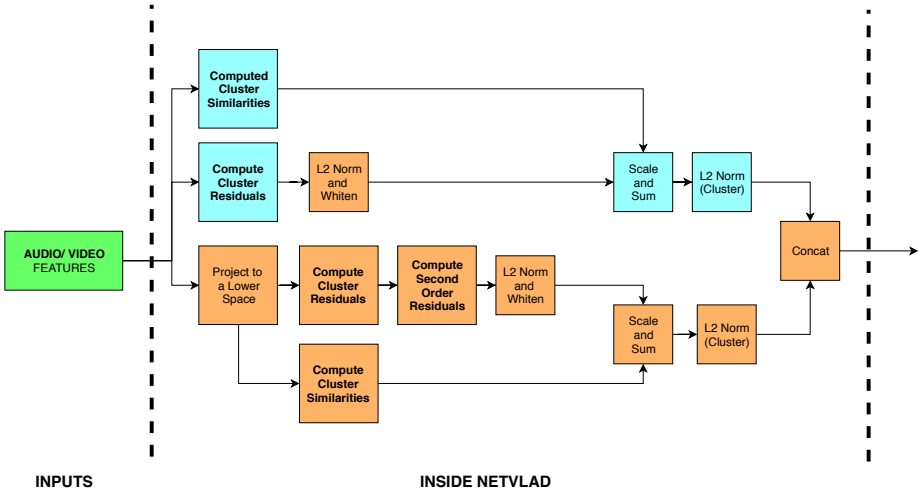

Fig. 3. A block diagram of the second order, function approximation based model. In addition to the usual NetVLAD implementation, we add residual normalization and whitening, along with second order terms based on projected input features.

portions of the Youtube-8M dataset for training, while leaving out a random 2% of data for the sake of local validation testing.

All transformer related experiments are trained using the Adam optimizer [10] with an initial learning rate of 0.0003 and a batch size of 32 (64 in all) on two NVIDIA P100 GPUs. All function approximation related experiments are performed using a batch size of 4 (32 in all) on eight NVIDIA K80 GPUs. For each video, we utilized uniform sampling of 256 frames, to be able evenly select features consistently, while maintaining temporal consistency, for the sake of attention-based models.

We did not train many of our models exhaustively due to time and resource constraints while participating in this competition. We stop training early after roughly 3 epochs (270k steps). For complete implementation details, visit our GitHub repository.

4.2 Testing Results

In the "The 2nd YouTube-8M Video Understanding Challenge", models are evaluated using the Global Average Precision (GAP) metric. In Equation (9), p(i) and r(i) refer to the precision and recall of the top i predictions, respectively.

$$GAP = \sum_{k=1}^{20} p(i) * r(i) \tag{9}$$

Our results are encouraging, but currently, do not improve the state of the art. Our largest issue for the models listed in Table 1, as well as other models

Table 1. A collection of the highest testing scores, as determined by Global Average Precision (**GAP**). Second Order refers to Section 3.3, Attention Enhanced refers to Section 3.1 and Attention NetVLAD refers to Section 3.2

Name	Training Steps	Batch Size	Public Test **GAP**
Baseline NetVLAD	270k	80	**0.870**
Second Order	270k	32	0.865
Attention Enhanced	270k	64	0.856
Attention NetVLAD	220k	64	**0.867**

that exist in our GitHub repository, is the problem of generalization and/or overfitting, which is a relatively poorly understood topic. During training, we clearly have the capacity to learn training regularities, however, we inevitably overfit, even when experimenting with common techniques such as dropout and early stop.

Despite our misfortune, it is possible to extend our second order models by creating a parallelizable Sinkhorn scaling algorithm to make use of crucial democratic weights, or by avoiding the stage of projecting input features into a low dimensional space (given hardware resources for such a model). In addition, exploration of other regularization techniques, such as regularizer cost functions from Section 3.3 are promising.

5 Conclusions

In this paper, we made modifications to the state-of-the-art approaches for aggregating local video descriptors. We drew a connection between NetVLAD and Transformers to learn cluster similarities per local descriptors. In addition, we had some encouraging results using a function approximation approach. These techniques increased model capacity compared to the state of the art. Experimental results demonstrate that the testing accuracy is similar to that of the state of the art. Due to the time constraints of the competition, we did not fully investigate parameter tuning or overfitting issues. We plan to explore regularization costs, as well as other avenues discussed for accuracy improvement.

References

1. Abu-El-Haija, S., Kothari, N., Lee, J., Natsev, A.P., Toderici, G., Varadarajan, B., Vijayanarasimhan, S.: Youtube-8m: A large-scale video classification benchmark. In: arXiv:1609.08675 (2016), https://arxiv.org/pdf/1609.08675v1.pdf
2. Arandjelovi, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(6), 1437–1451 (June 2018). https://doi.org/10.1109/TPAMI.2017.2711011

3. Bian, Y., Gan, C., Liu, X., Li, F., Long, X., Li, Y., Qi, H., Zhou, J., Wen, S., Lin, Y.: Revisiting the effectiveness of off-the-shelf temporal modeling approaches for large-scale video classification. arXiv preprint arXiv:1708.03805 (2017)
4. Brock, A., Lim, T., Ritchie, J.M., Weston, N.: Neural photo editing with introspective adversarial networks. CoRR **abs/1609.07093** (2016), http://arxiv.org/abs/1609.07093
5. Do, T., Tran, Q.D., Cheung, N.: Faemb: A function approximation-based embedding method for image retrieval. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015. pp. 3556–3564 (2015). https://doi.org/10.1109/CVPR.2015.7298978, https://doi.org/10.1109/CVPR.2015.7298978
6. Girdhar, R., Ramanan, D., Gupta, A., Sivic, J., Russell, B.: Actionvlad: Learning spatio-temporal aggregation for action classification
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR **abs/1512.03385** (2015), http://arxiv.org/abs/1512.03385
8. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local image descriptors into compact codes. IEEE Trans. Pattern Anal. Mach. Intell. **34**(9), 1704–1716 (2012). https://doi.org/10.1109/TPAMI.2011.235, https://doi.org/10.1109/TPAMI.2011.235
9. Jégou, H., Zisserman, A.: Triangulation embedding and democratic aggregation for image search. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014. pp. 3310–3317 (2014). https://doi.org/10.1109/CVPR.2014.417, https://doi.org/10.1109/CVPR.2014.417
10. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014), http://arxiv.org/abs/1412.6980
11. Long, X., Gan, C., de Melo, G., Wu, J., Liu, X., Wen, S.: Attention clusters: Purely attention based local feature integration for video classification. CoRR **abs/1711.09550** (2017), http://arxiv.org/abs/1711.09550
12. Miech, A., Laptev, I., Sivic, J.: Learnable pooling with context gating for video classification. CoRR **abs/1706.06905** (2017), http://arxiv.org/abs/1706.06905
13. moe: Hierarchical mixtures of experts and the EM algorithm. Neural Computation **6**(2), 181–214 (1994). https://doi.org/10.1162/neco.1994.6.2.181, https://doi.org/10.1162/neco.1994.6.2.181
14. Radenovic, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. In: IEEE Computer Vision and Pattern Recognition Conference (2018)
15. Radenovic, F., Iscen, A., Tolias, G., Avrithis, Y.S., Chum, O.: Revisiting oxford and paris: Large-scale image retrieval benchmarking. CoRR **abs/1803.11285** (2018), http://arxiv.org/abs/1803.11285
16. Tolias, G., Avrithis, Y., Jégou, H.: Image search with selective match kernels: aggregation across single and multiple images. International Journal of Computer Vision **116**(3), 247–261 (2016)
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA. pp. 6000–6010 (2017), http://papers.nips.cc/paper/7181-attention-is-all-you-need
18. Xie, S., Sun, C., Huang, J., Tu, Z., Murphy, K.: Rethinking spatiotemporal feature learning for video understanding. arXiv preprint arXiv:1712.04851 (2017)

19. Yu, K., Zhang, T.: Improved local coordinate coding using local tangents. In: Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel. pp. 1215–1222 (2010), http://www.icml2010.org/papers/454.pdf
20. Zhu, Y., Wang, J., Xie, L., Zheng, L.: Attention-based pyramid aggregation network for visual place recognition. arXiv preprint arXiv:1808.00288 (2018)