Supplementary material for

What have we learned from deep representations for action recognition?

Christoph Feichtenhofer
TU Graz
feichtenhofer@tugraz.at

Axel Pinz
TU Graz
axel.pinz@tugraz.at

Richard P. Wildes
York University, Toronto
wildes@cse.yorku.ca

Andrew Zisserman
University of Oxford
az@robots.ox.ac.uk

This document is best viewed offline in Adobe reader where all optical flow figures should play as videos. All these visualizations either play automatically (on repeat), or need user interaction (click). If interaction is required, it will be listed above the respective figures.

The supplementary material is structured in two main blocks. In the first block, A, we show additional visualizations for the VGG-16 Two-Stream Fusion architecture [3], under different strengths of spatiotemporal variation regularization; this corresponds to the experiments shown in Sect. 4.2 of the main paper. Specifically, we show early layers with different visualization styles in Sect. A.1 and under isotropic regularization (Sect. A.2), followed by fusion layers in Sect. A.3, global layers in Sect. A.4 and classification layers in Sect. A.5.

In the second block, B, we visualize various other deep architectures: Temporal Segment Networks [13] using BN-Inception [6] (*i.e.* a batch-normalized GoogLeNet [10]) (in Sect. B.1), Spatiotemporal Residual Networks [2] using ResNet50 [5] streams (in Sect. B.2), as well as Inception_v3 [11] two-stream networks [8] (in Sect. B.3). These models are of increasing depth (BN-Inception<ResNet50<Inception_v3) and were trained on multiple action recognition datasets (after ImageNet pre-training): UCF101 [9], HMDB51 [7] and Kinetics [1] to discuss both network depth and dataset scale, which varies widely; the number of video clips is approximately: 6K in HMDB51, 13K in UCF101 and 300K in Kinetics; the number of classes are 51, 101 and 400, respectively.

Besides the difference in model capacity, the visualized architectures also differ in the numbers of frames at the motion stream input: The inception networks (shown in B.1 & B.3) use a stack of 5 frames of optical flow, while the VGG-16 fusion architecture (shown in A) and the spatiotemporal ResNet (shown in B.2) use 10 frames of optical flow as motion input.

Contents

A Additional results for varying spatiotemporal regularization **2**
A.1. Comparison of different flow encodings . . . 2
A.2. Visualization of early layers under isotropic spatiotemporal regularization 3
A.3. Convolutional fusion layer visualizations . . 3
A.4. Global layer visualizations 3
A.5. Prediction layer visualizations 3

B Visualizing multiple architectures and datasets **27**
B.1. Visualization of Inception networks 27
B.2. Visualization of Spatiotemporal Residual Networks 43
B.3. Visualization of Inception_v3 networks . . . 48

List of highly interesting figures

A.10 Visualizations of classification units at the last layer of the Two-Stream Fusion network [3]. 14
B.21 Visualizations of convolutional layer inception˙5b˙double˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13]. . . 42
B.25 Visualizations of classification units at the last layer of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. 44
B.34 Visualizations of convolutional fusion and motion filters of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. 50
B.40 Visualizations of convolutional layers of an Inception_v3 [11] two-stream network trained on Kinetics [1]. 55
B.43 Visualizations of classification units at the last layer of an Inception_v3 [11] two-stream network trained on Kinetics [1]. . . 58

A. Additional results for varying spatiotemporal regularization

The results shown in this block (A) explore the layers of a VGG-16 Two-Stream Fusion architecture [3] under various spatiotemporal regularization strengths as defined in equation (3) of the main paper. An interactive web-based schematic of the VGG-16 (ImageNet) architecture, showing the layer names with respective number of parameters (ch) and output resolution (width×height) for a 224×224 sized input, can be found here.

A.1. Comparison of different flow encodings

We first visualize filters of the early convolutional layers from the appearance and motion streams of the VGG-16 architecture. While we can simply visualize the appearance input as an RGB image showing the color channels, the question of what is the best way to visualize the optical flow stream arises. In Fig. A.2 we show the convolutional layers of VGG-16 (conv1_2 to conv5_3) in different optical flow visualization encodings. We compare three alternatives for visualizing the optical flow stream: A magnitude plot that plays the optical flow inputs with the RGB channels representing horizontal, vertical and magnitudes of the flow; the classic optical flow encoding used in several optical flow benchmarks that encodes flow in the HSV color space, as explained in Fig. A.1; and the horizontal and vertical optical flow vectors plotted as grayscale videos. These visualizations correspond to the unconstrained temporal case, $\chi = 0$; *i.e.* we do not use a variational regularizer in the temporal domain.

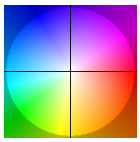


Figure A.1. Flow field coding for the HSV visualization. The displacement of a point corresponds to the vector from the centre of the figure, marked with a cross.

Since we think the magnitude plot, which visualizes the motion as colour videos with the RGB channels corresponding to the horizontal, vertical and magnitudes of the optical flow vectors, is perceptually easier to interpret than the other variants, such as grayscale images for the individual flow components or HSV colour encodings, we use this style of visualization throughout the main submission as well as the remainder of this supplement.

From the content of Fig. A.2, we observe that when going deeper in the network hierarchy, filters get more specific capturing complex structures at the last convolutional layers. Also interesting is that temporal variation increases when going deeper in the network hierarchy. For example, in Fig. A.2, the bottom-right filters at conv1_2 and conv2_2 show no temporal variation whereas we see increased temporal variation at the deeper network layers (*e.g.* at conv3_3 and conv4_3).

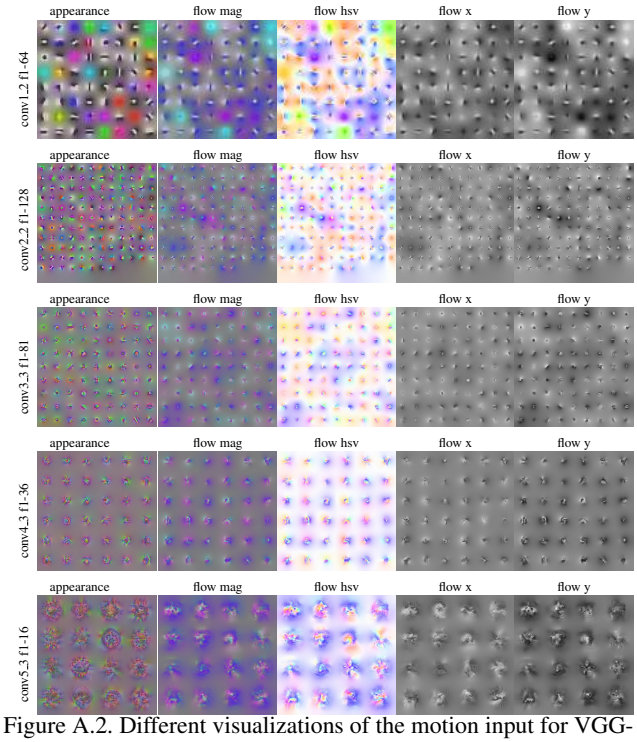


Figure A.2. Different visualizations of the motion input for VGG-16 layers conv1_1 to conv5_3. The temporal dimension is unpenalized (*i.e.* $\chi = 0$).

A.2. Visualization of early layers under isotropic spatiotemporal regularization

In Fig. A.3 we first show what excites the convolutional filters of a two-stream architecture when varying the isotropic spatiotemporal regularization (this is in comparison to the anisotropic variation in Sect. 4.2 of the main paper). The motion signals are reconstructed under spatiotemporal regularization with different regularization strengths, κ, and $\chi = \kappa$. Varying the regularization in spacetime reveals interesting properties of the underlying representation. We discuss Fig. A.3 from two perspectives: First, from the temporal perspective, we see that the early layer filters are more robust to regularization in spacetime, whereas higher layers show larger dependency on the regularization strength, κ. This dependence originates from the temporally consistent nature of the early filters that exhibit temporal low-pass characteristics. Second, from the spatial perspective, we see that with decreasing spacetime regularization strength, κ, high-frequency inputs become dominant. Especially for low regularization factors $\kappa \leq 2.5$, we see high-frequency patterns dominating and reconstruction artifacts appearing in the background. These artifacts can be explained by the linear amplification in higher dimensional spaces and directly relate to adversarial examples [4].

In Fig. A.4 we show the evolution of internal filters of the VGG16 motion stream during training. After 10 epochs we see very minor temporal variation of the filters which increases when going towards the end of learning at epoch 60. Interestingly, the spatial shape of the early layer filters at conv2_2 do not change while the temporal dimension builds up over epochs. The higher layers, in contrast, undergo large changes of spatiotemporal structure from epoch 10 to epoch 60.

A.3. Convolutional fusion layer visualizations

The next visualizations shown in Figures A.5 - A.8 alternately show samples from the conv5_fusion layers under isotropic (varying κ & $\chi = \kappa$), and non-isotropic (varying χ, while holding κ constant), spacetime TV norms as defined in equation (3) of the main paper. It is recommended to view the two regularization forms in a two-page, side by side, view to see the full variability of these local filters (each unit is shown in 8 different motion and 2 different appearance visualizations that is shown in the rows of Figures A.5 - A.8). Interesting patterns appear that act as a local, distributed representation for multiple abstract features at the higher layers. In Fig. A.9 we also show the maximum test set activity of corresponding filters at the fusion layer, for the 10 highest activating videos across the test set. Most of the filters fire for classes with similar appearance and motion, but also strong activity over a broad set of classes can be observed.

A.4. Global layer visualizations

This section shows additional visualizations for the fully-connected layers in the motion stream of the VGG-16 fusion architecture, that have global receptive fields by operating on pooled local fusion features. In particular, Fig. A.10 shows filters of the fully-connected 6 (fc_6) layer while Fig. A.11 shows filters of the subsequent fully-connected_7 (fc_7) layer. We again show the maximum test set activity of corresponding units at the fully connected layers.

A.5. Prediction layer visualizations

In Figures A.12-A.28, we show additional visualizations for the learned VGG-16 fusion model on the classes present in UCF101. Again, we visualize appearance and optical flow inputs for different temporal regularization strengths (χ). Notably, in some cases, the fast temporal variation case ($\chi = 0$) reveals large scale motion that is not present in the slower cases. For example, for the Billiards class the fastest visualization indicates player movement around the pool table, etc.

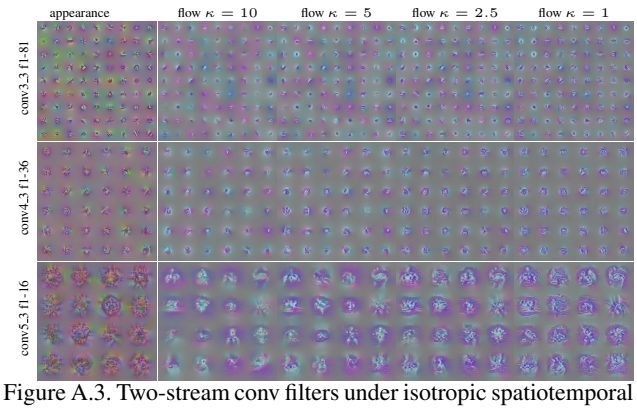


Figure A.3. Two-stream conv filters under isotropic spatiotemporal variation regularization. We show appearance and the optical flow inputs for slowest $\kappa = 10$, slow $\kappa = 5$, fast $\kappa = 2.5$, and faster $\kappa = 1$ spatiotemporal variation.

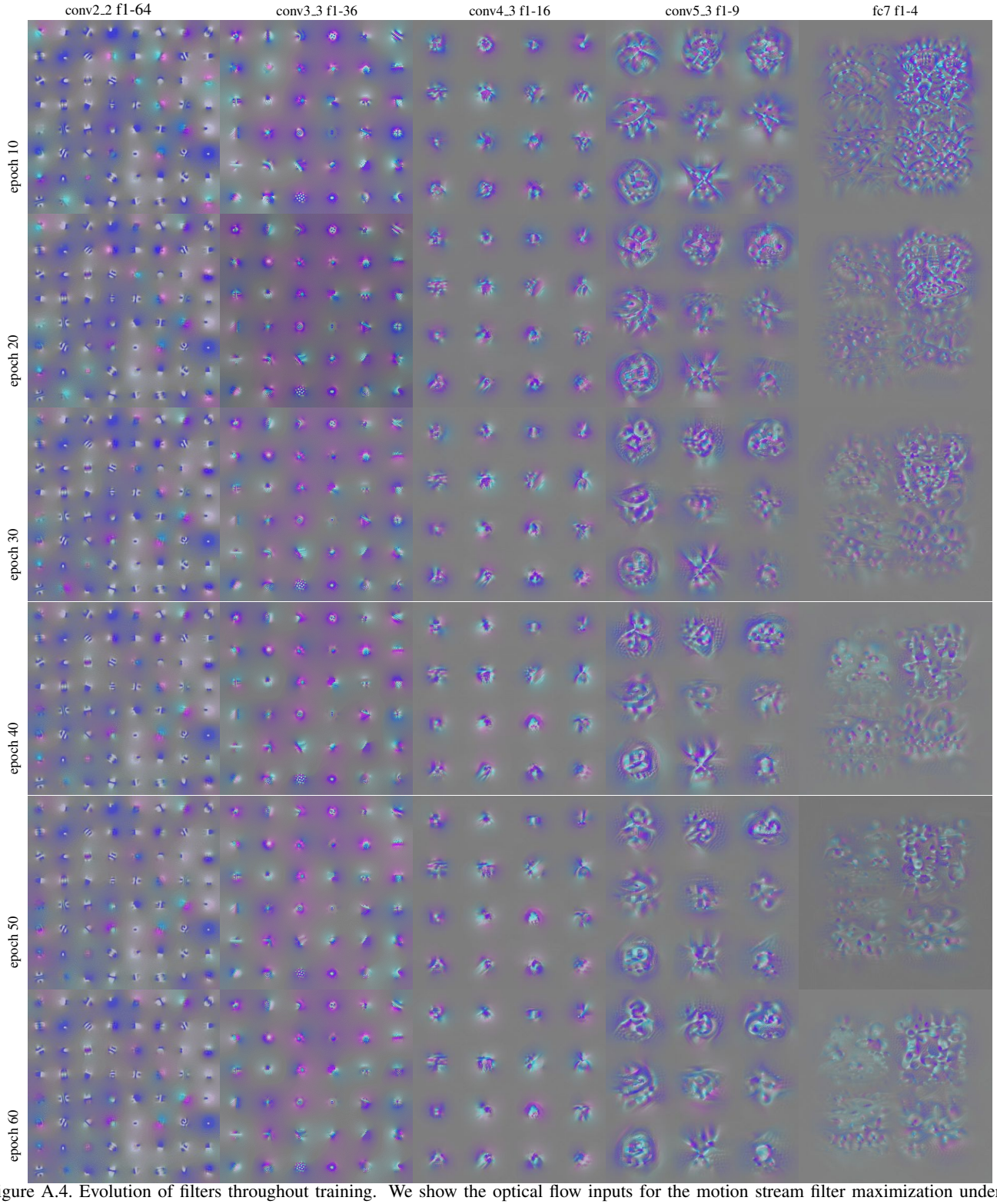


Figure A.4. Evolution of filters throughout training. We show the optical flow inputs for the motion stream filter maximization under spatiotemporal variation. Filters at early layers adapt fast whereas filters at the higher layers emerge later on in training, especially over the temporal dimension.

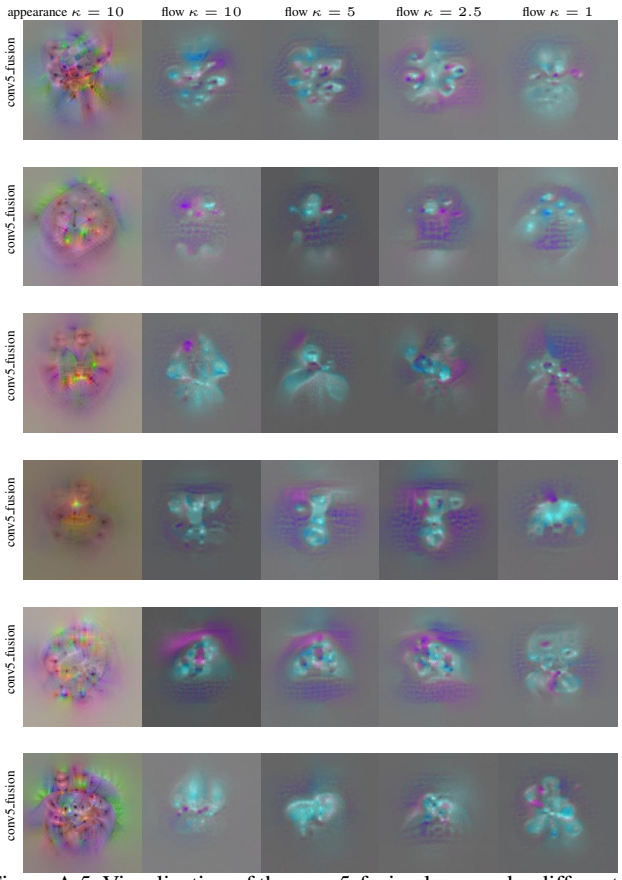


Figure A.5. Visualization of the conv5_fusion layer under different 3D spacetime TV regularization. We show appearance and the optical flow inputs for slowest $\kappa = 10$, slow $\kappa = 5$, fast $\kappa = 2.5$, and faster $\kappa = 1$, spatiotemporal variation.

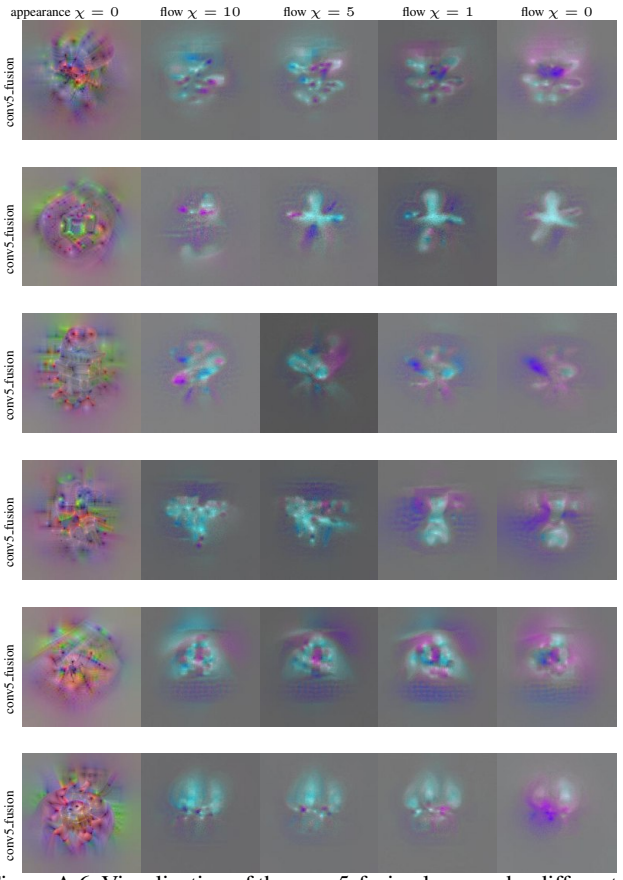


Figure A.6. Visualization of the conv5_fusion layer under different temporal TV regularization. We show the appearance input and the optical flow inputs for slowest $\chi = 10$, slow $\chi = 5$, fast $\chi = 1$, and unconstrained $\chi = 0$, temporal variation regularization.

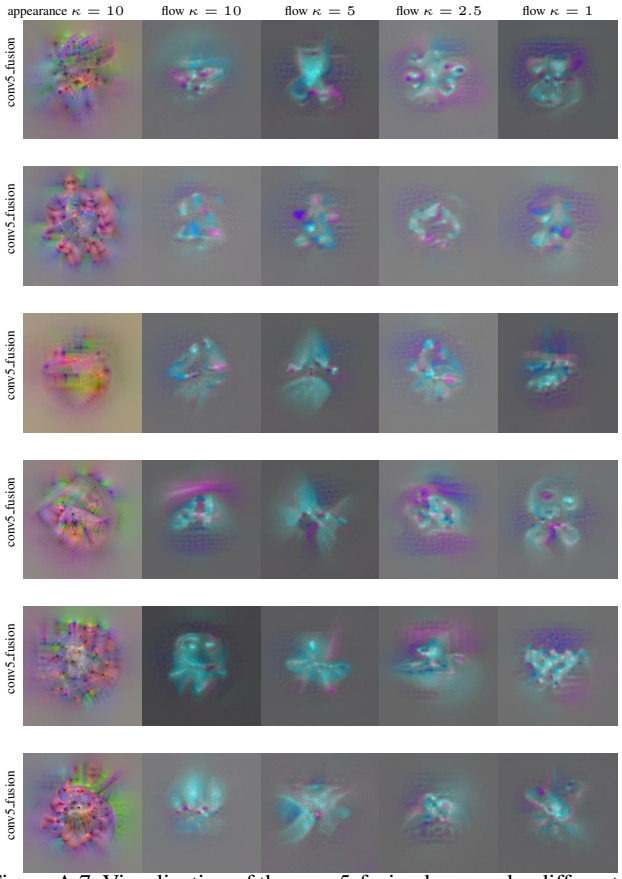


Figure A.7. Visualization of the conv5_fusion layer under different 3D spacetime TV regularization. We show appearance and the optical flow inputs for slowest $\kappa = 10$, slow $\kappa = 5$, fast $\kappa = 2.5$, and faster $\kappa = 1$, spatiotemporal variation.

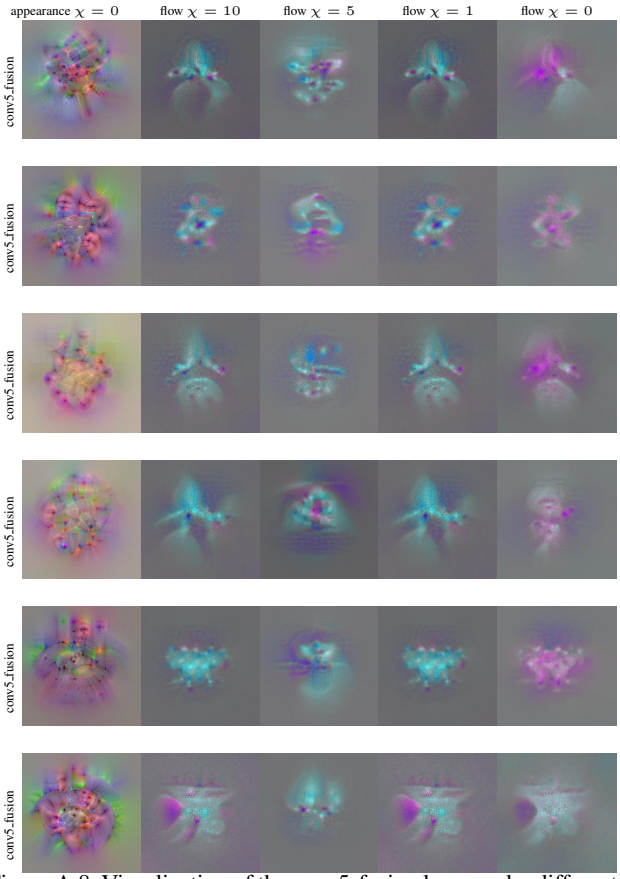


Figure A.8. Visualization of the conv5_fusion layer under different temporal TV regularization. We show the appearance input and the optical flow inputs for slowest $\chi = 10$, slow $\chi = 5$, fast $\chi = 1$, and unconstrained $\chi = 0$, temporal variation regularization.

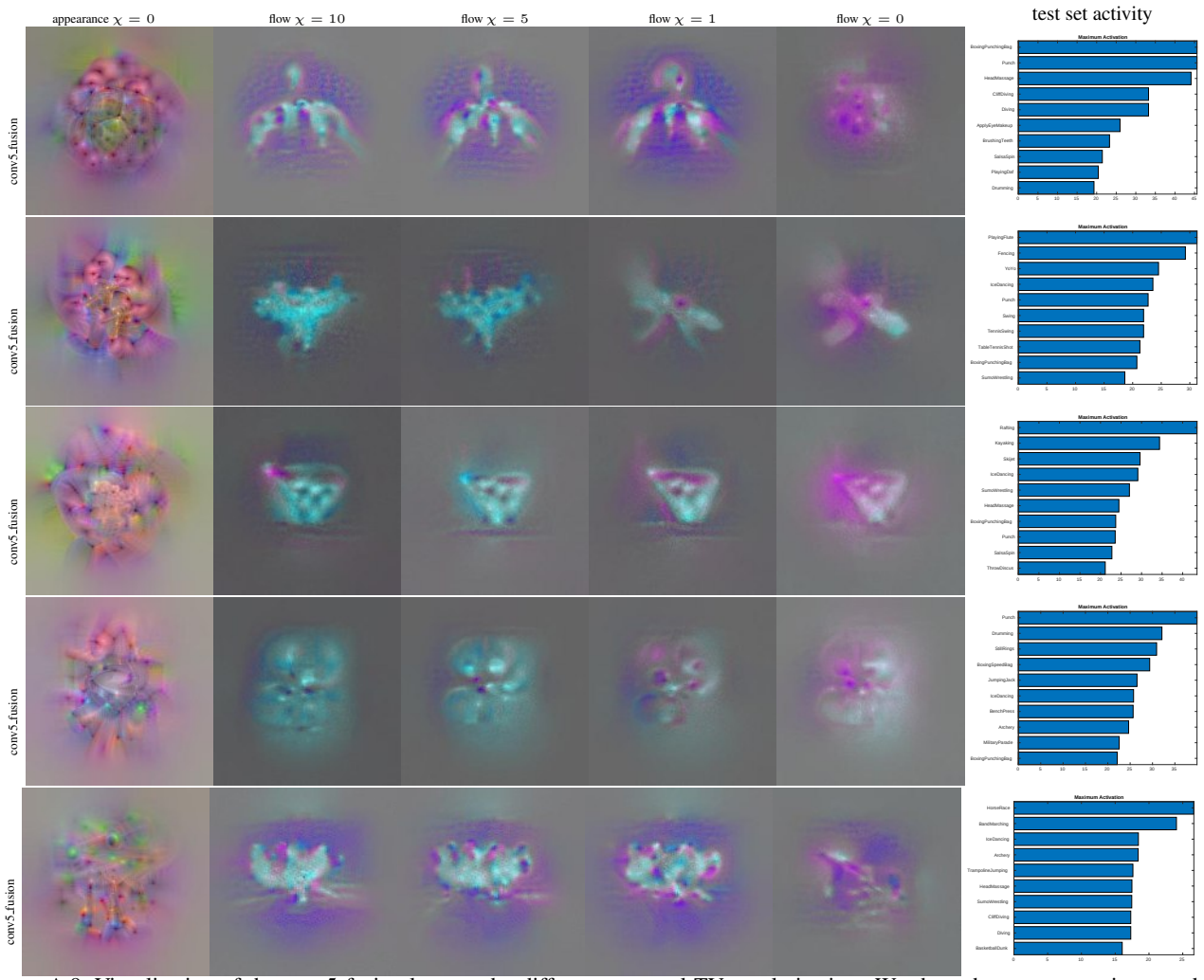


Figure A.9. Visualization of the conv5_fusion layer under different temporal TV regularization. We show the appearance input and the optical flow inputs for slowest $\chi = 10$, slow $\chi = 5$, fast $\chi = 1$, and unconstrained $\chi = 0$, temporal variation regularization. The last column shows the maximum activity observed for classes in the test set.

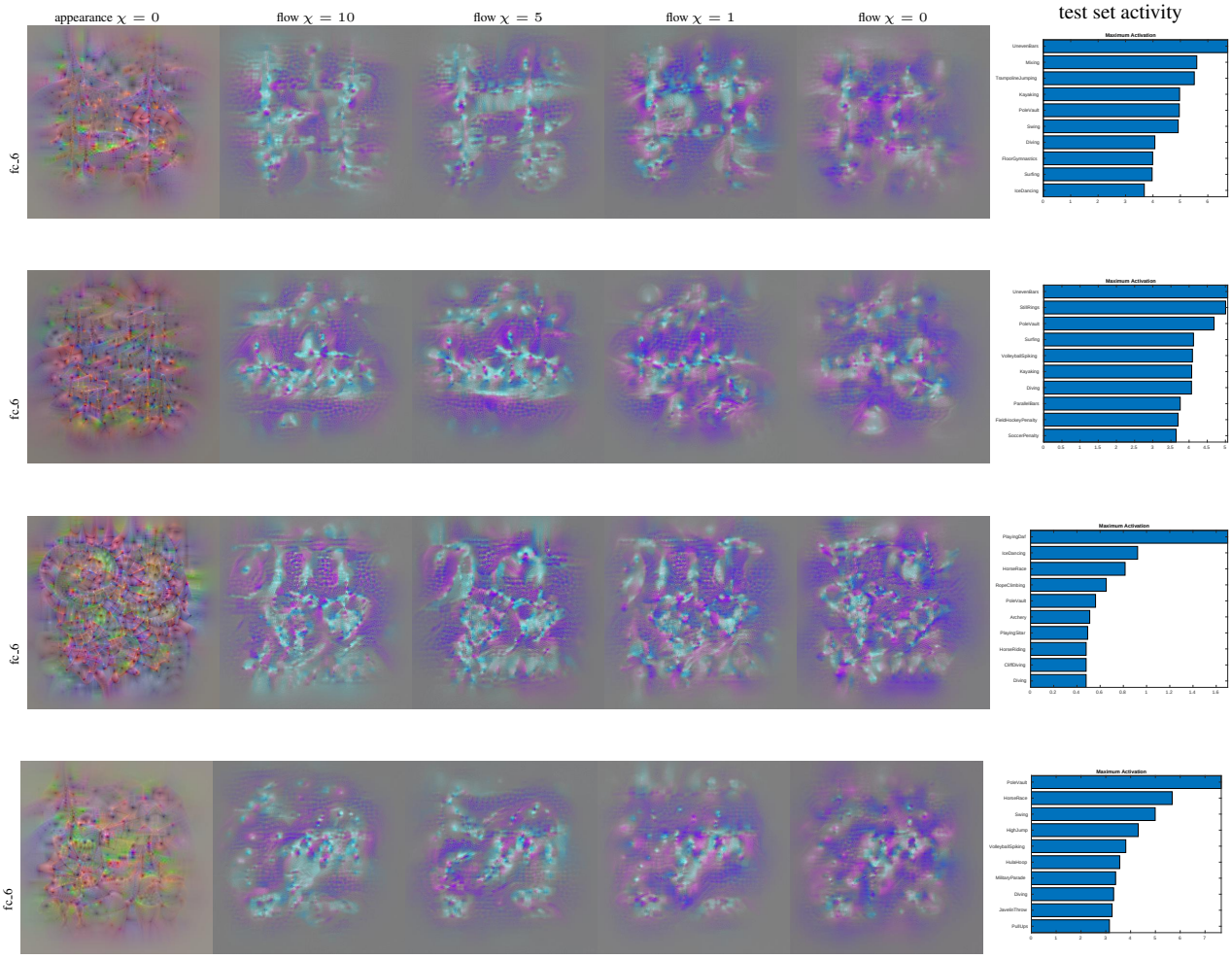


Figure A.10. Visualization of filters of the fc_6 layer under different temporal regularization. We show the appearance input and the optical flow inputs for slowest $\chi = 10$, slow $\chi = 5$, fast $\chi = 1$, and unconstrained (fastest) $\chi = 0$, temporal variation regularization. The last column shows the maximum activity observed for classes in the test set.

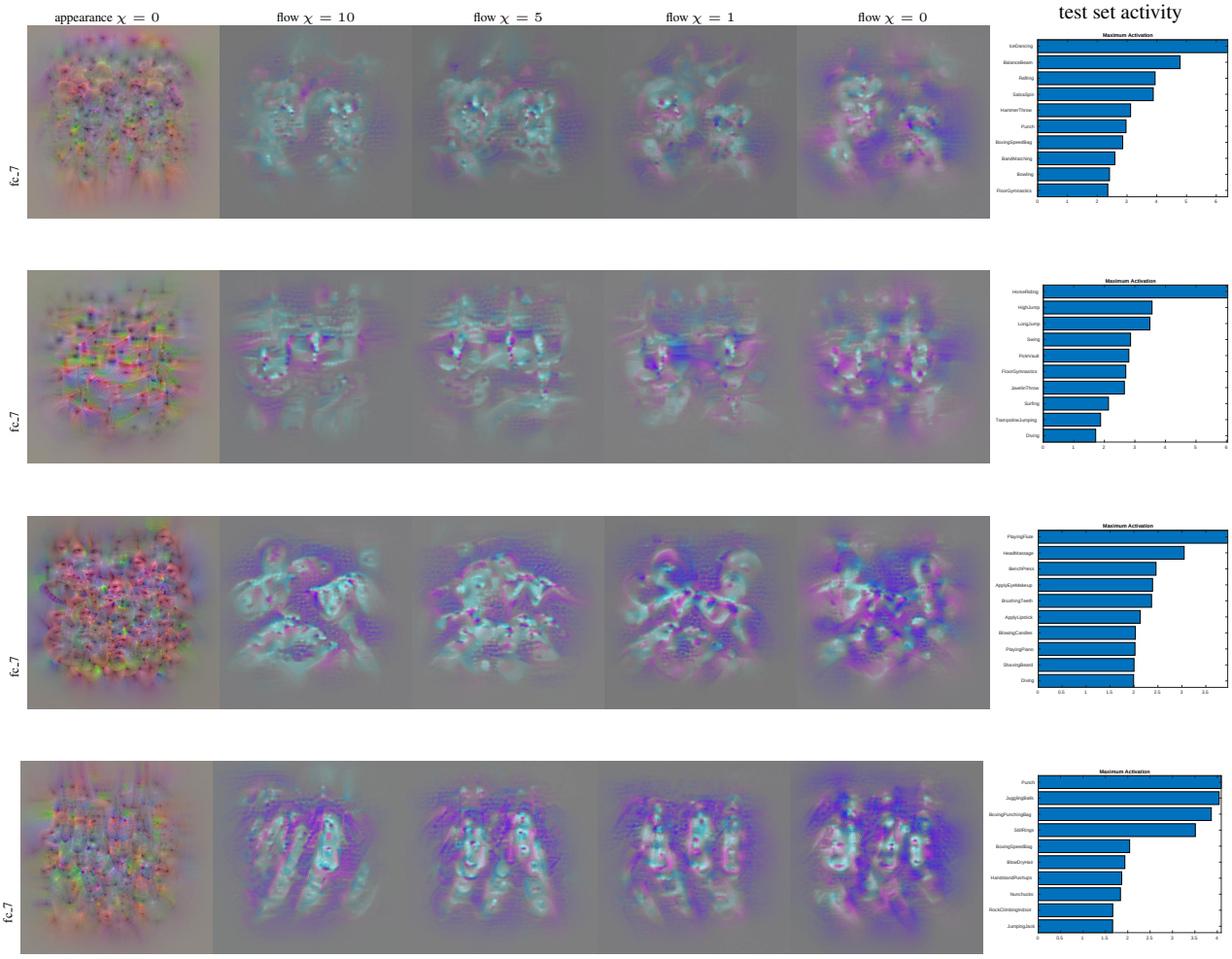


Figure A.11. Visualization of filters of the fc_7 layer under different temporal regularization. We show the appearance input and the optical flow inputs for slowest $\chi = 10$, slow $\chi = 5$, fast $\chi = 1$, and unconstrained (fastest) $\chi = 0$, temporal variation regularization. The last column shows the maximum activity observed for classes in the test set.

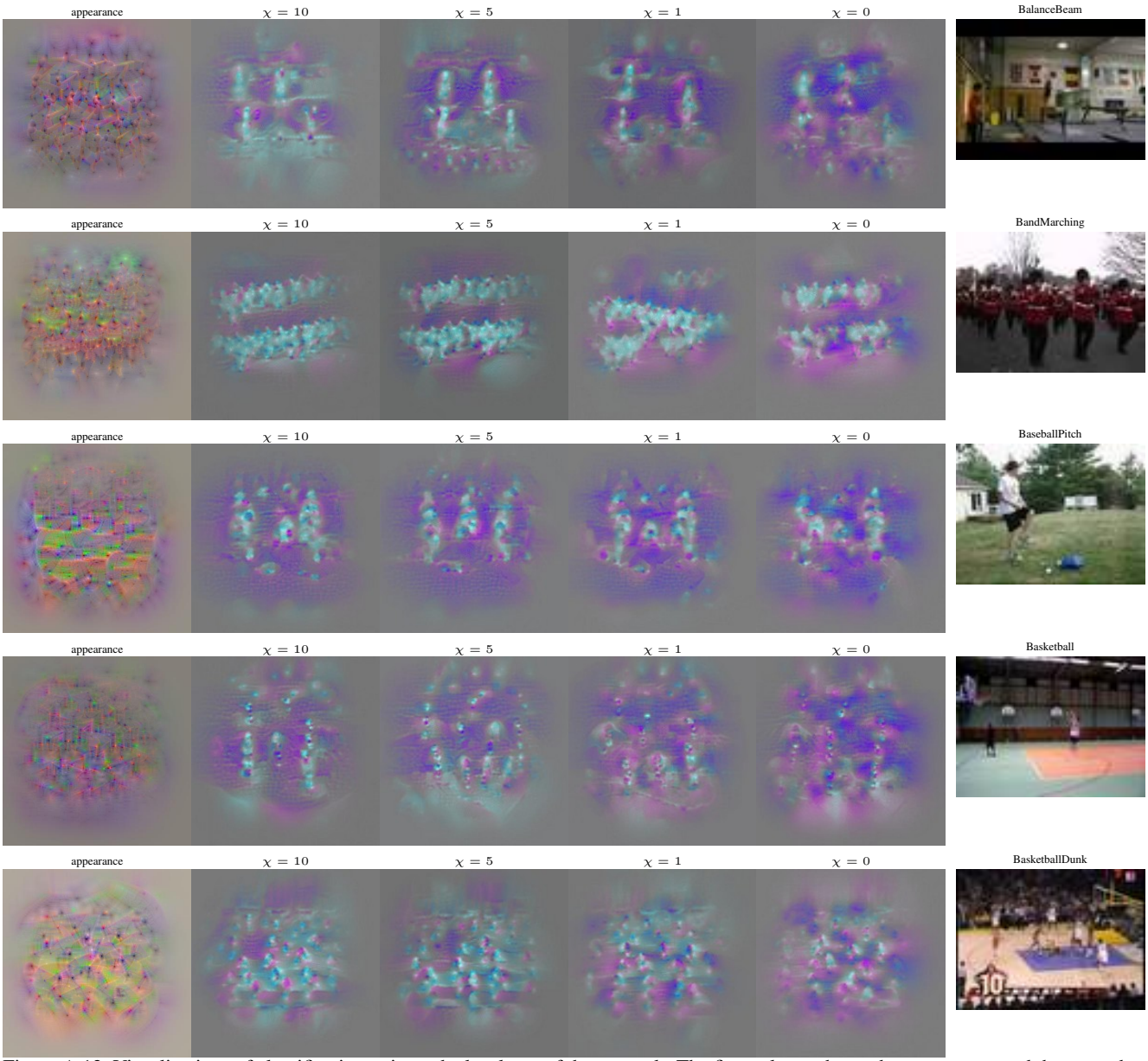


Figure A.12. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

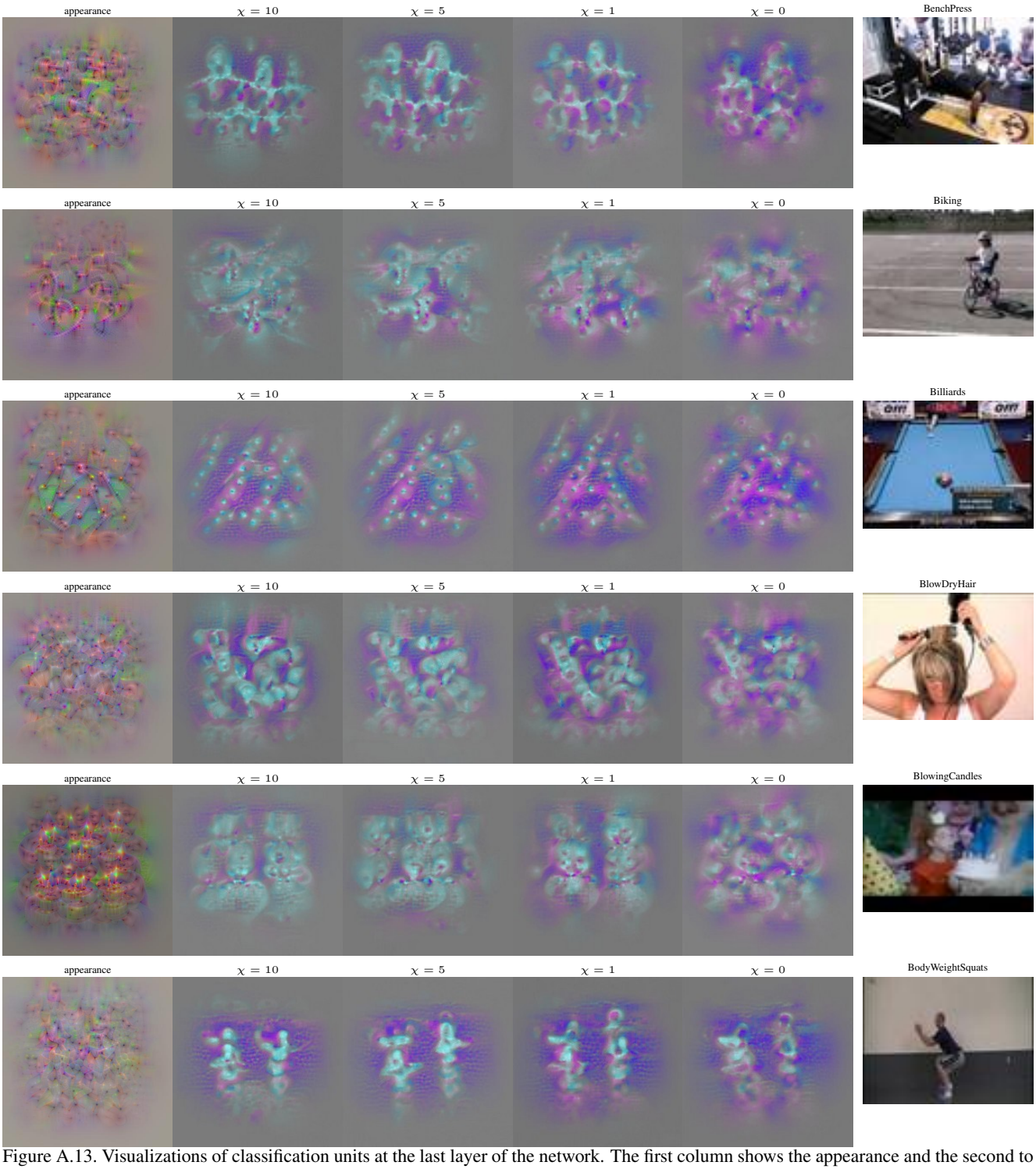


Figure A.13. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

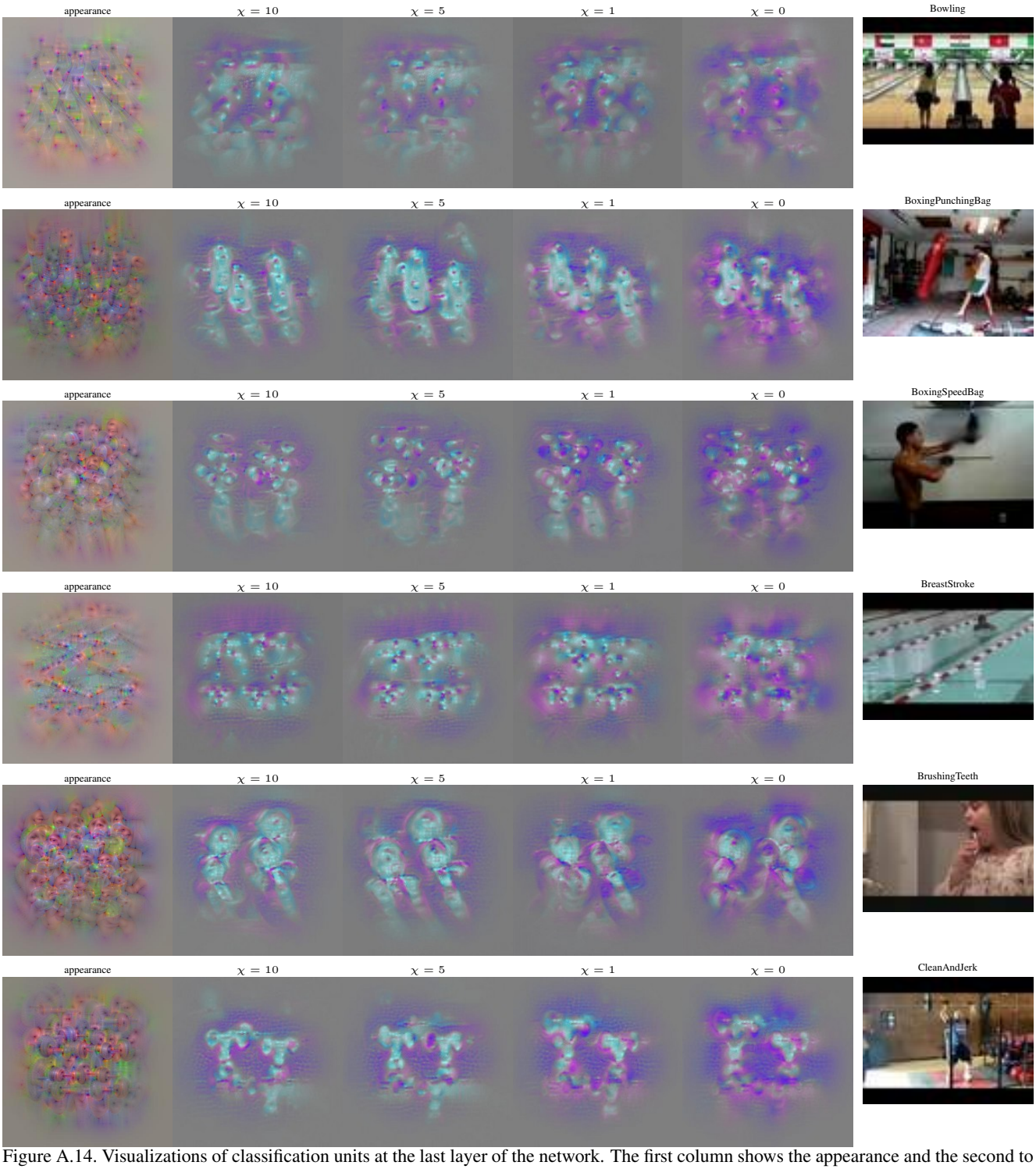


Figure A.14. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

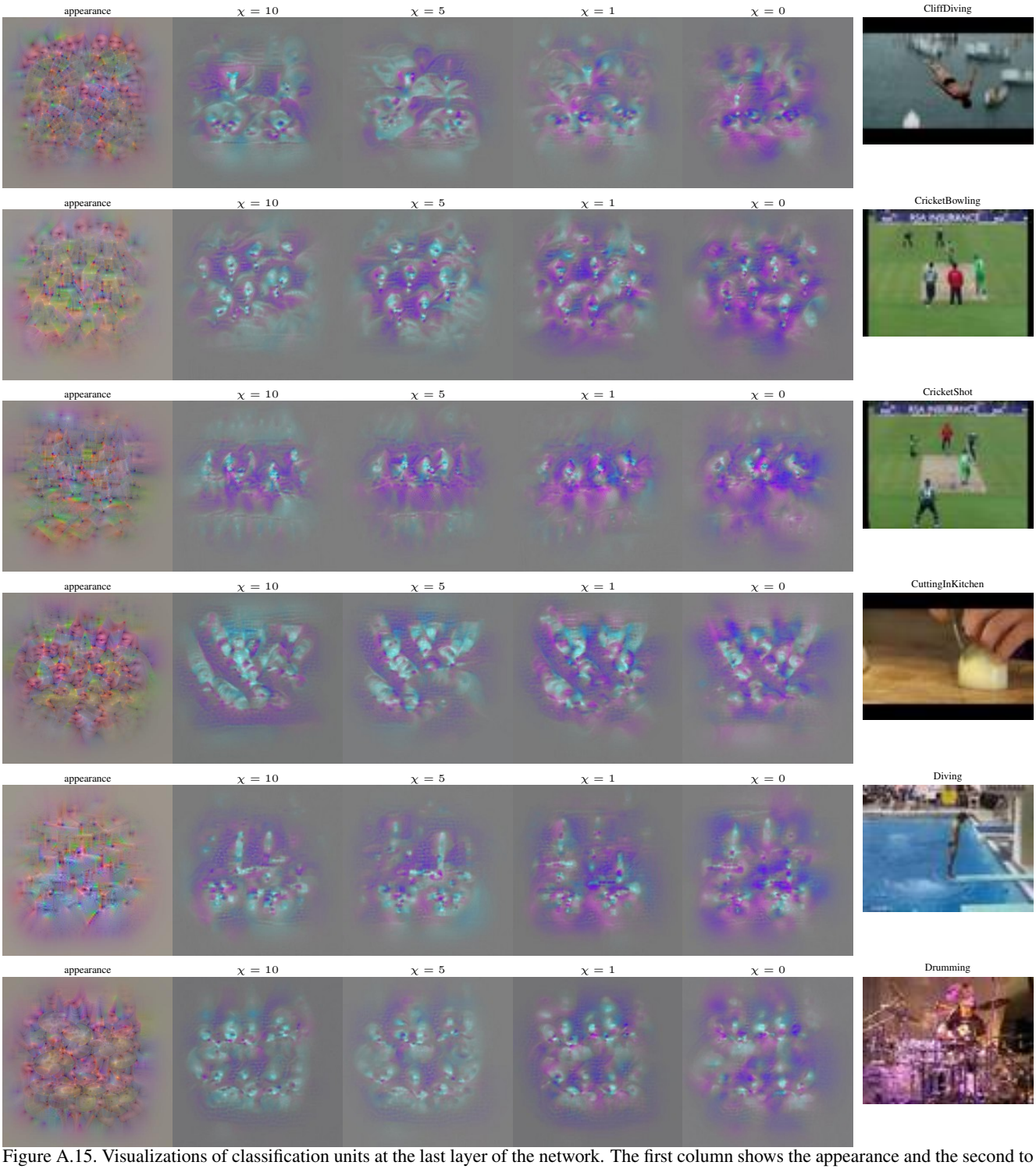


Figure A.15. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

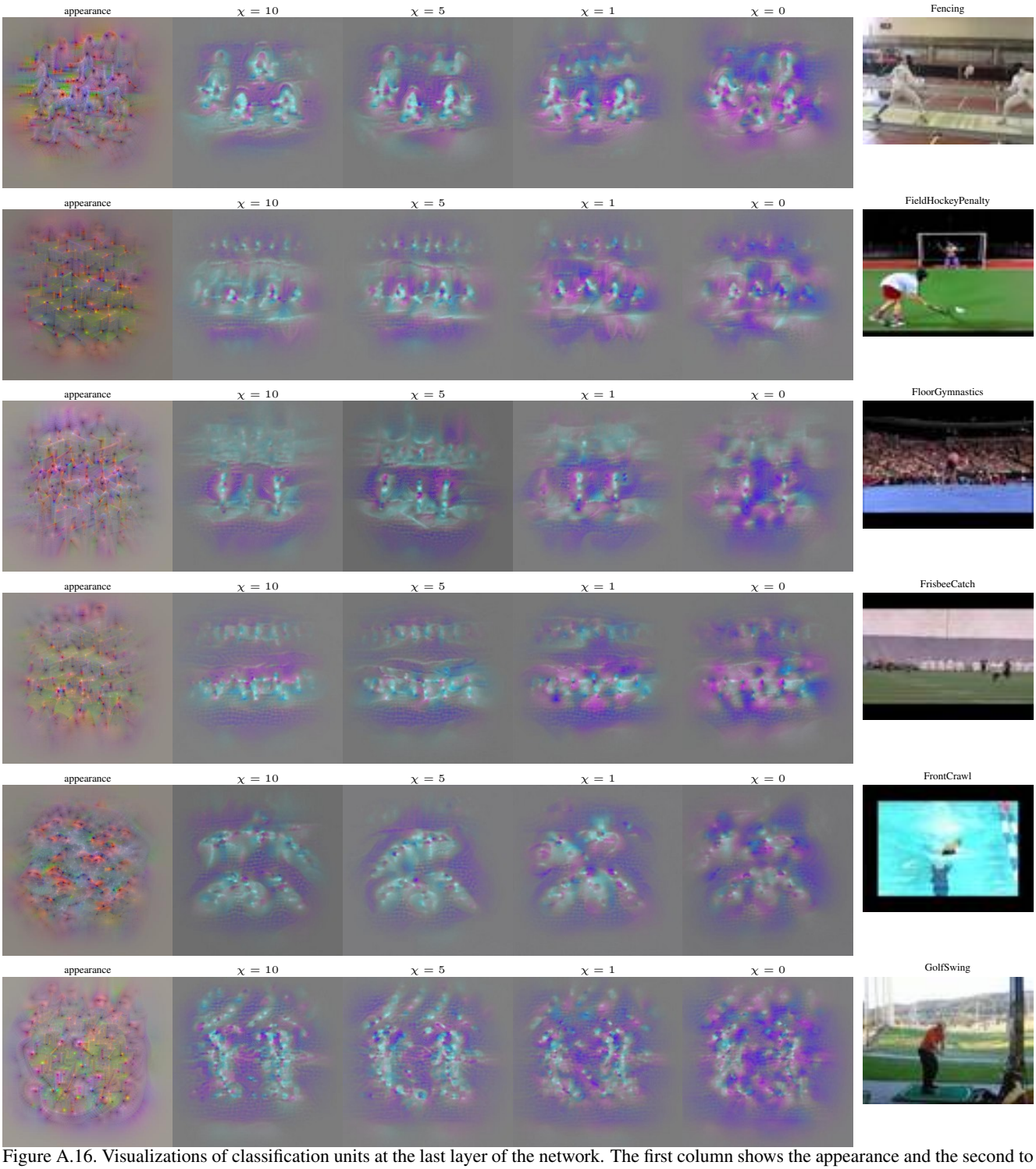


Figure A.16. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

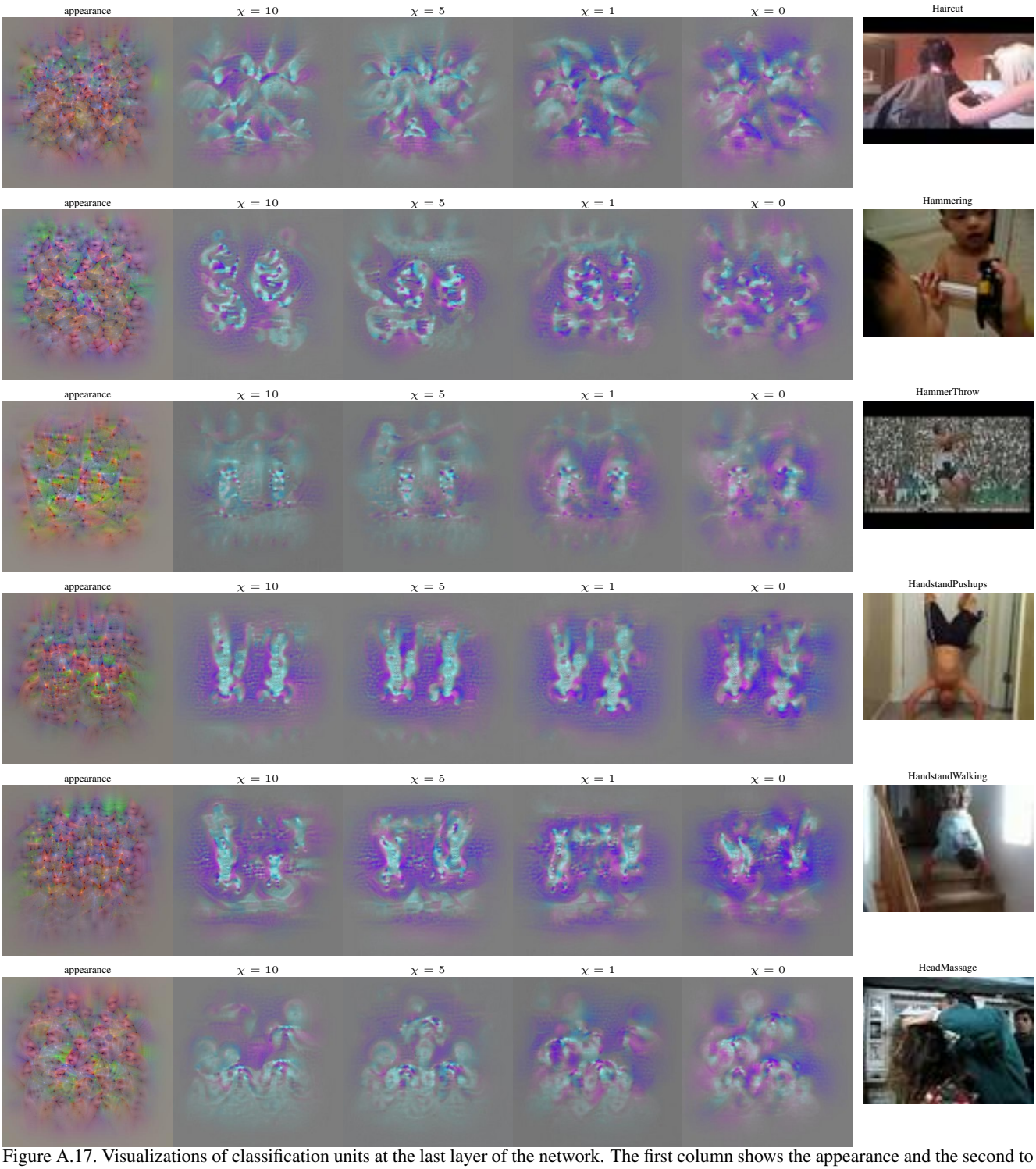


Figure A.17. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

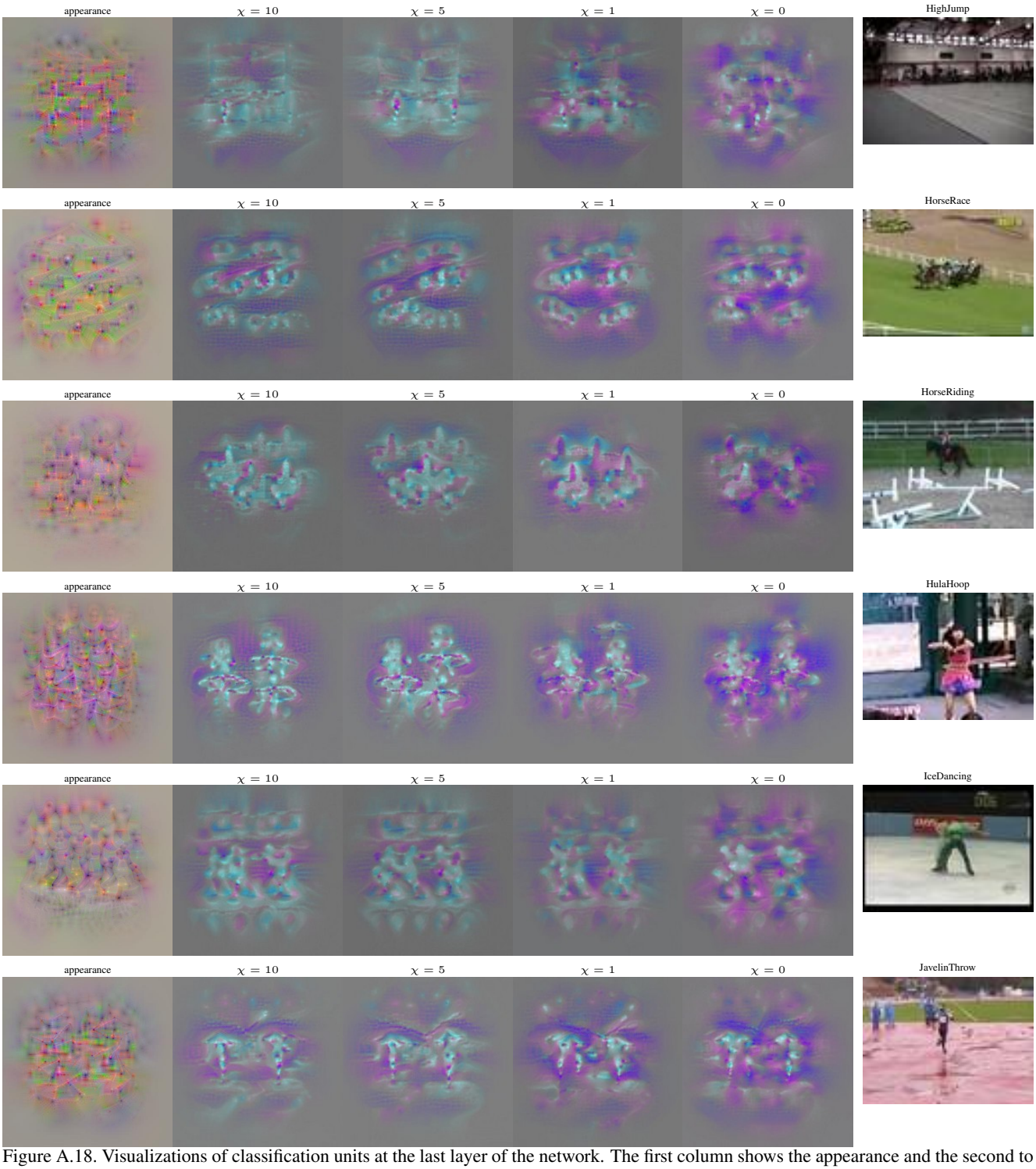


Figure A.18. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

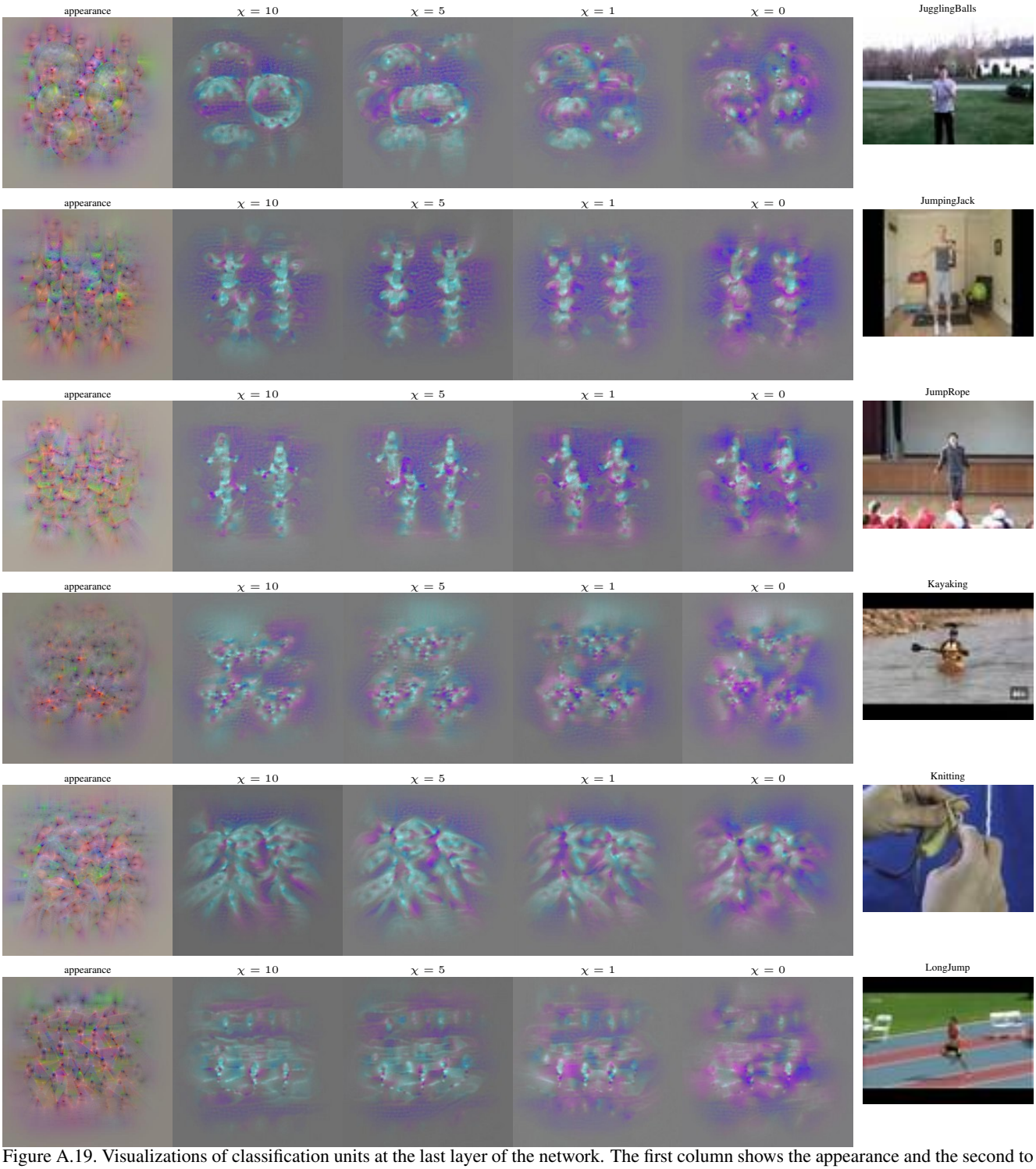


Figure A.19. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

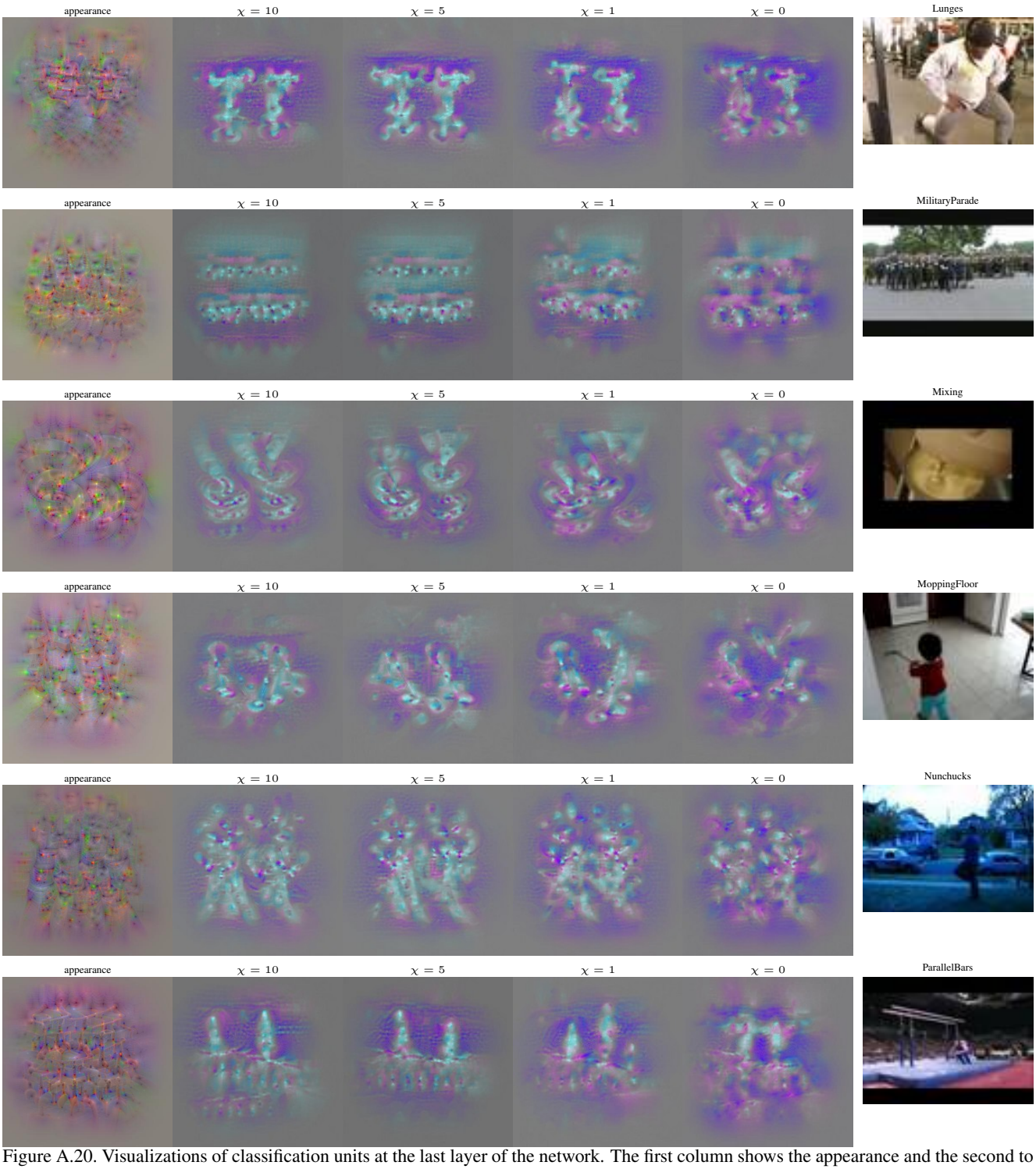


Figure A.20. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

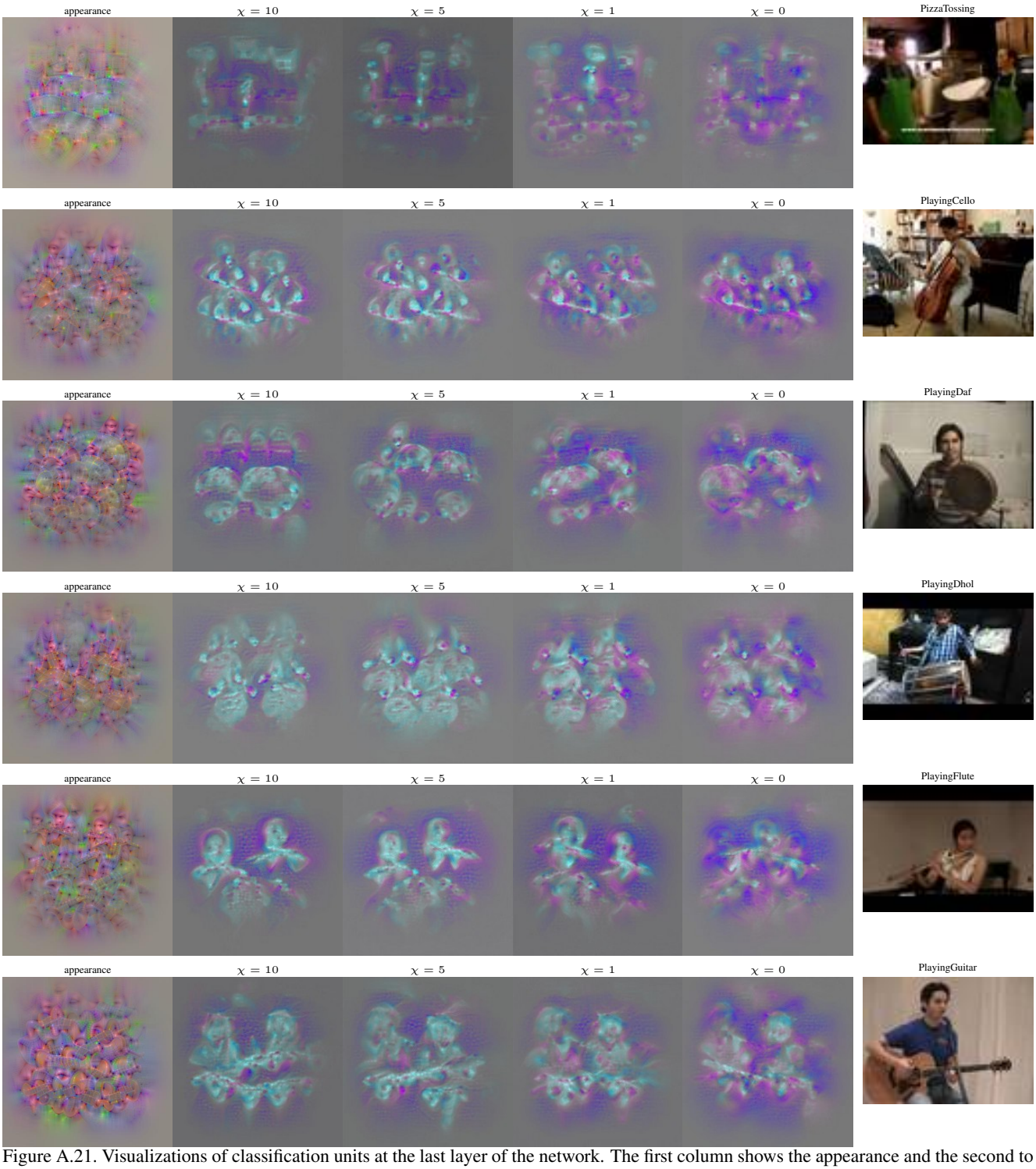


Figure A.21. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

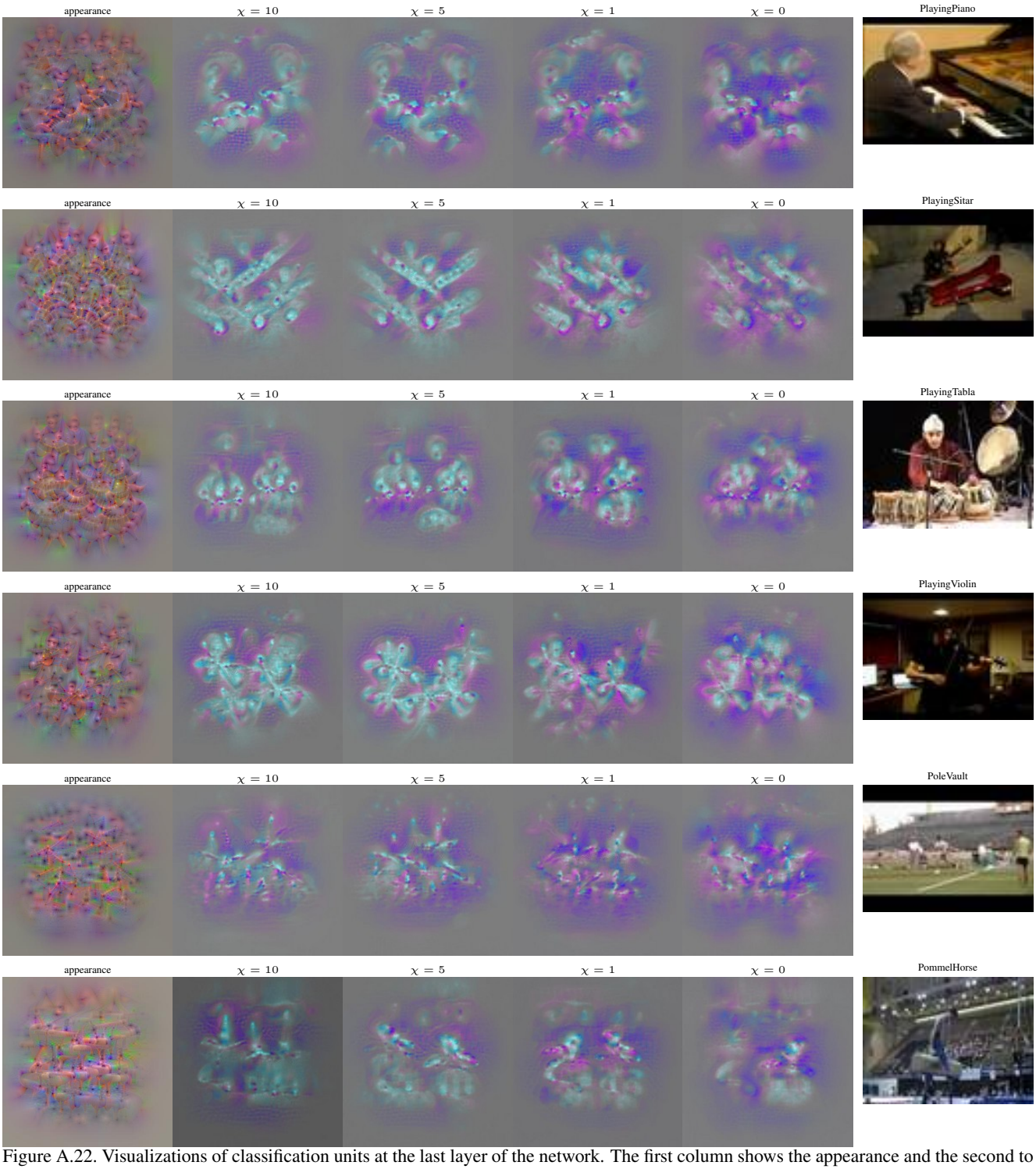


Figure A.22. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

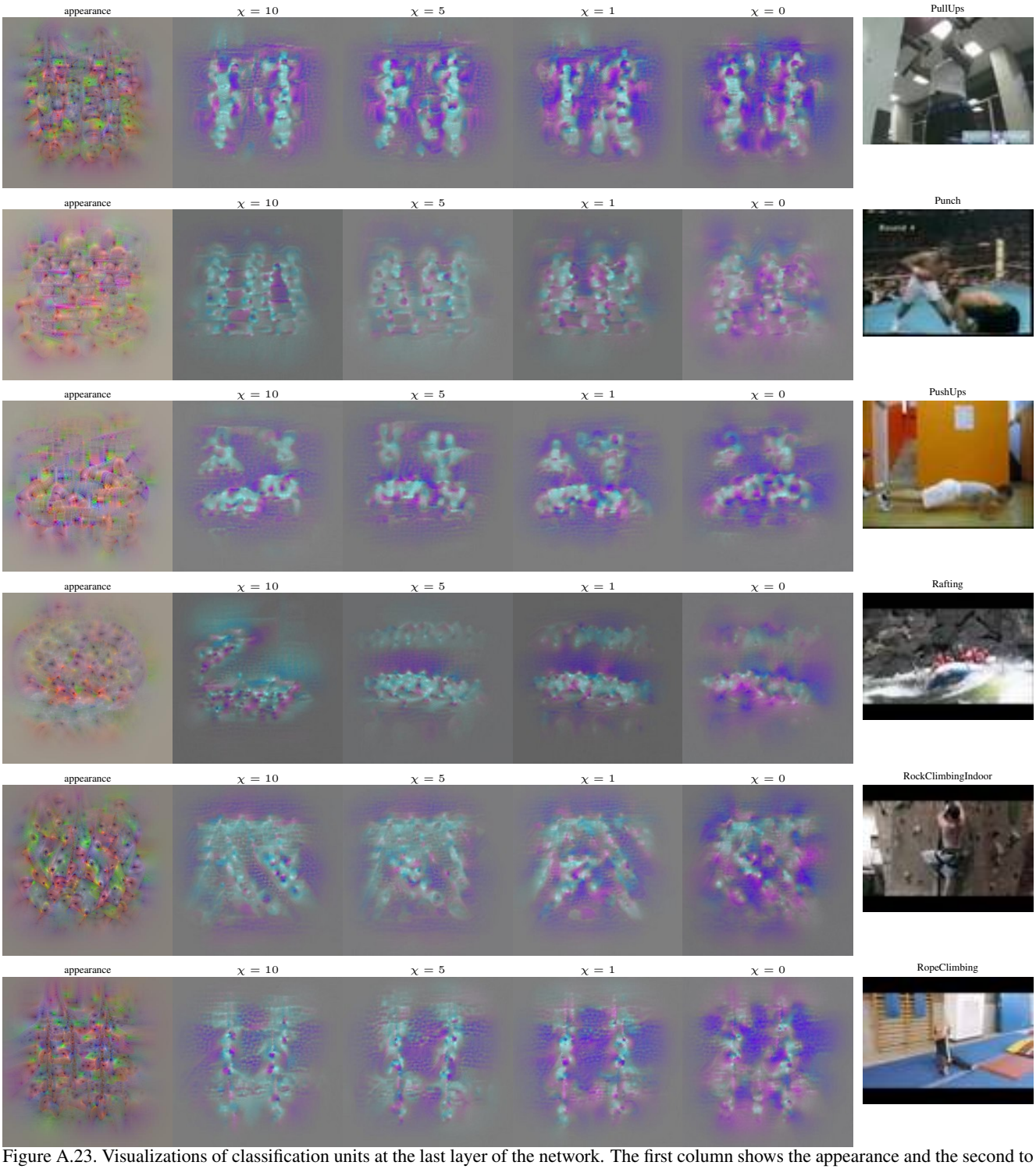


Figure A.23. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

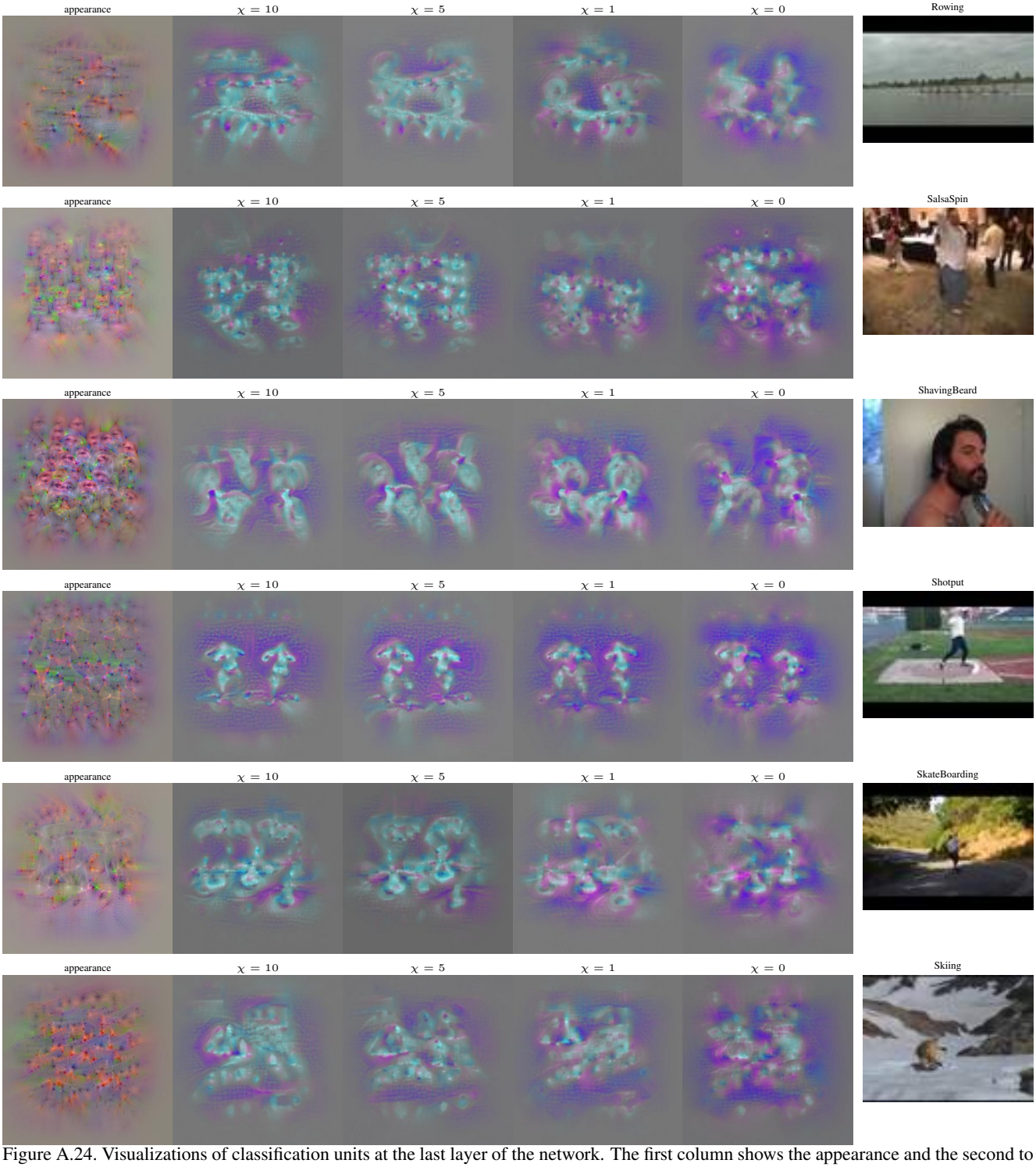


Figure A.24. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

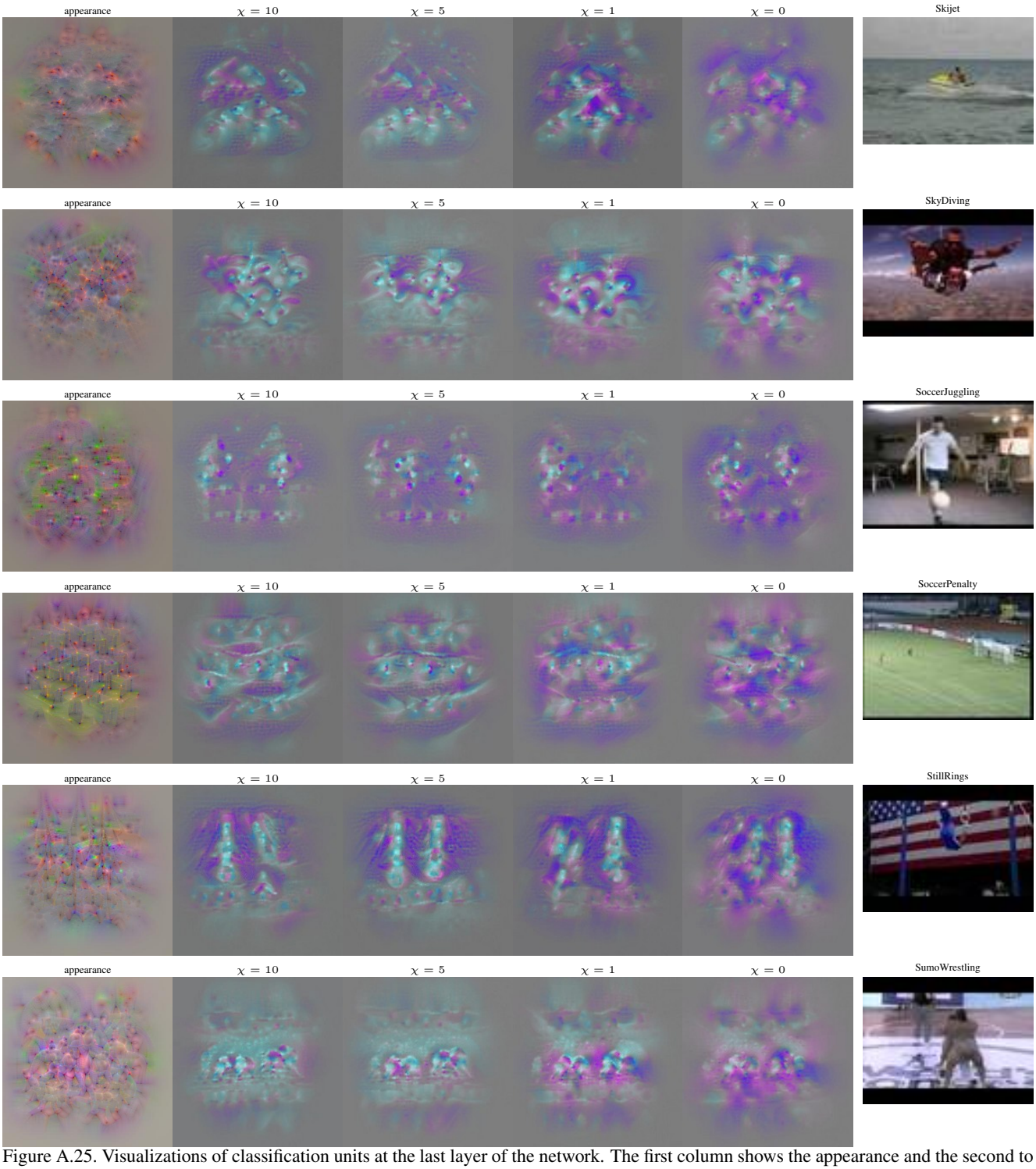


Figure A.25. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

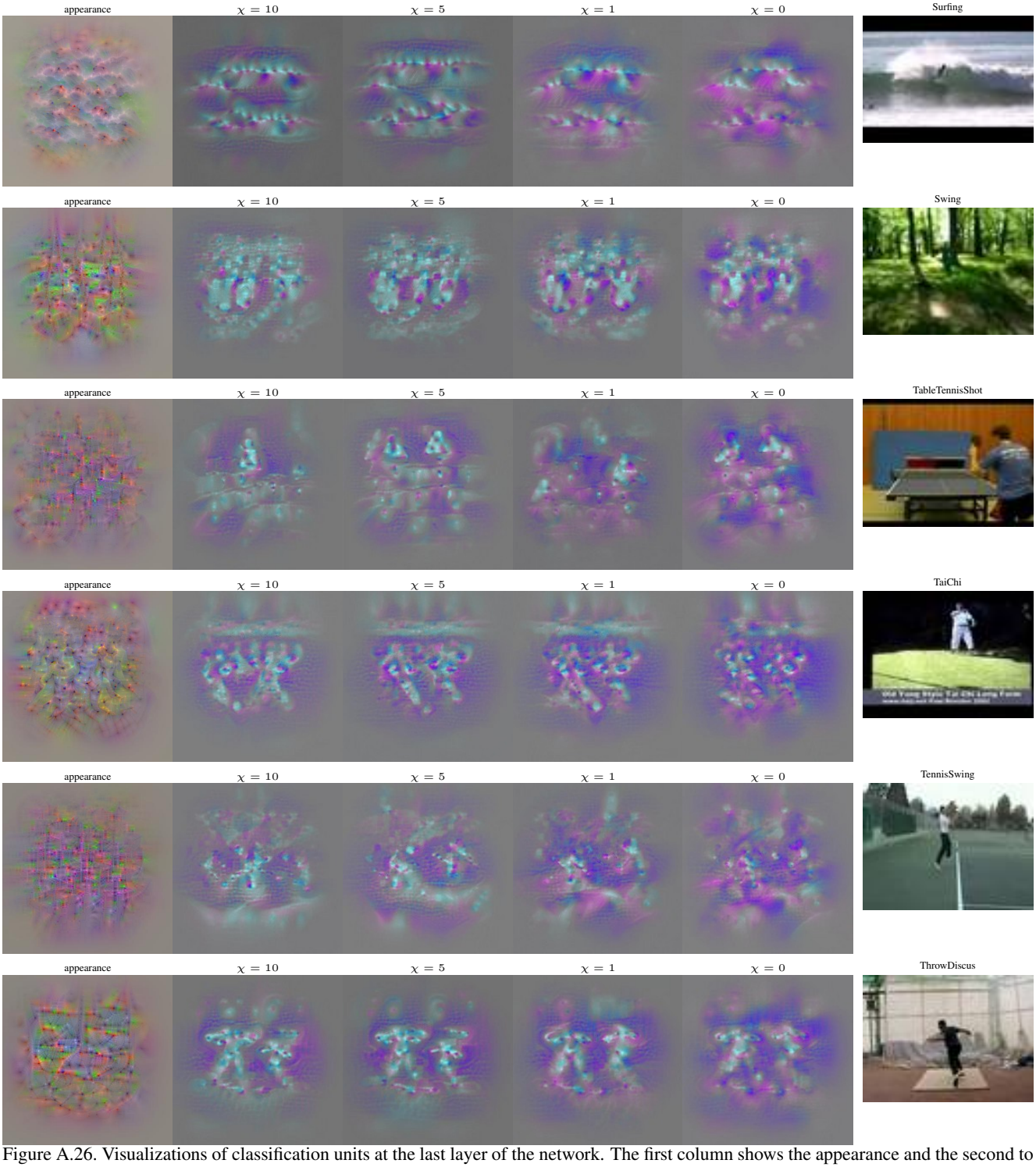


Figure A.26. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

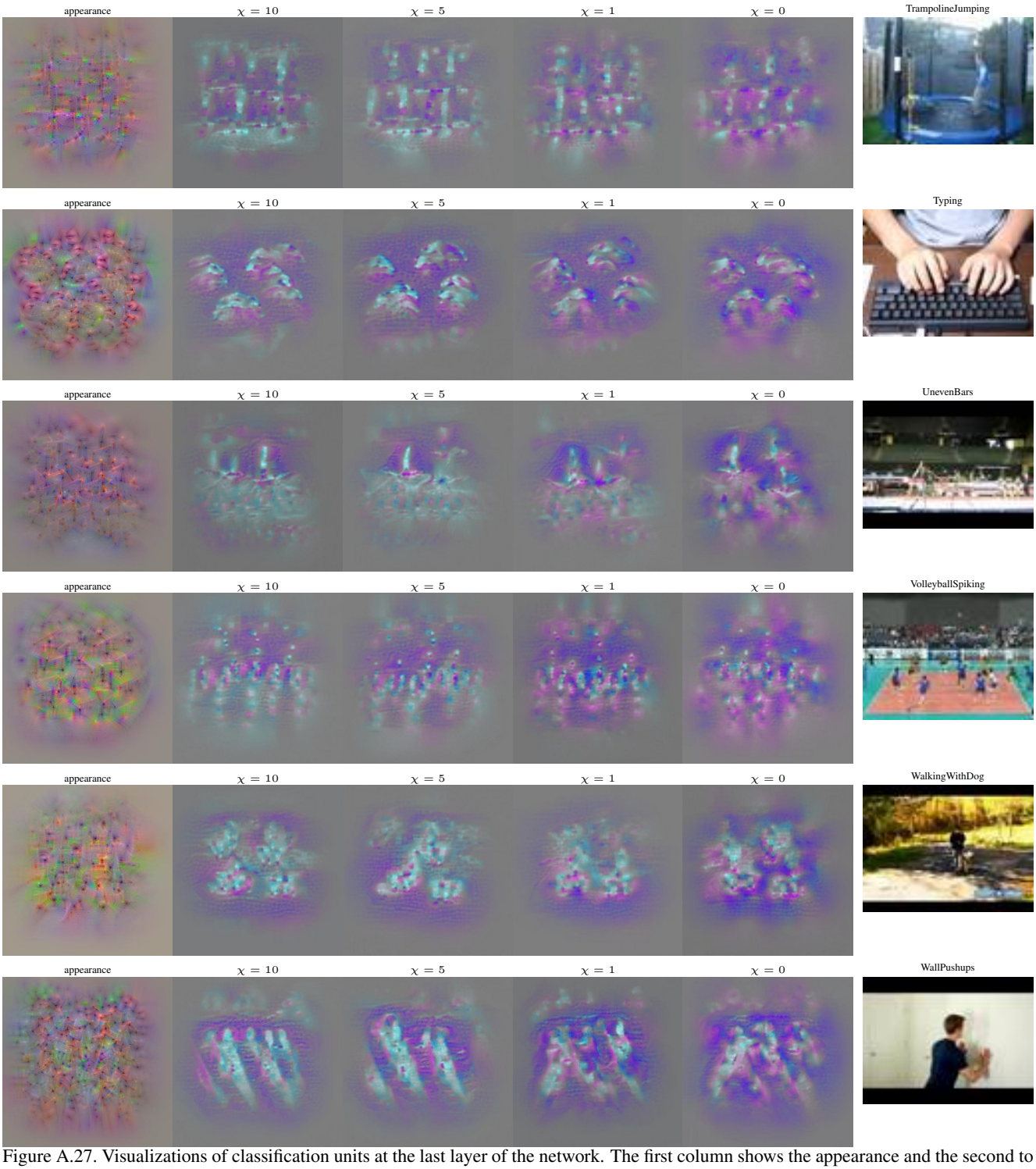


Figure A.27. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

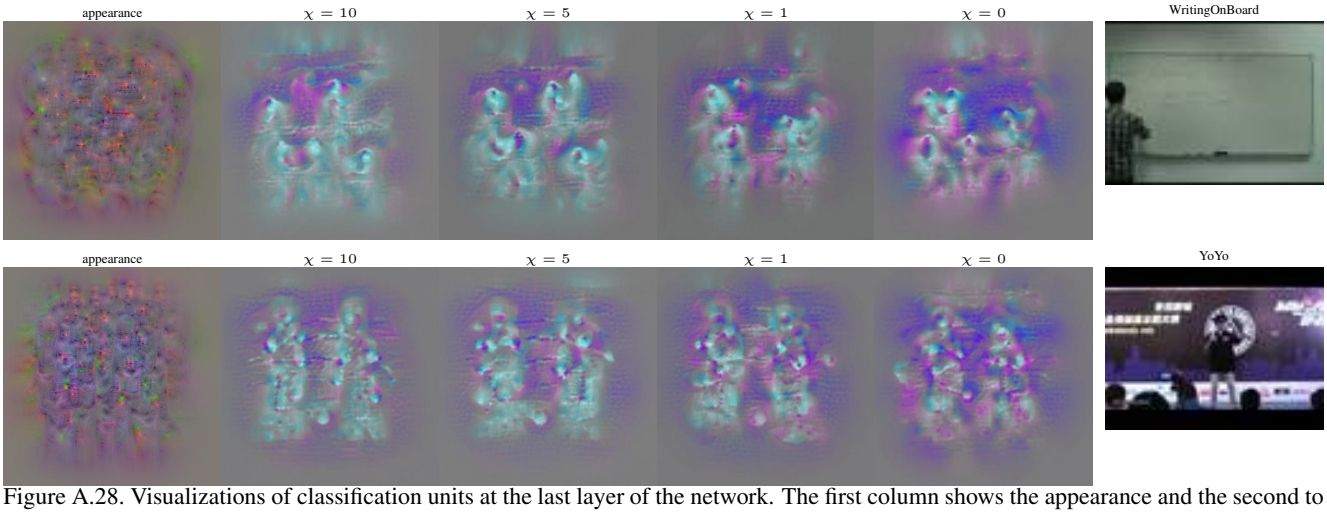


Figure A.28. Visualizations of classification units at the last layer of the network. The first column shows the appearance and the second to fifth column the motion input generated by maximizing the prediction layer output for the respective classes, while using different degrees of temporal variation regularization (χ). The last column shows 10 sample frames from the first video of the corresponding class in the test set. Best viewed electronically.

B. Visualizing multiple architectures and datasets

For sake of space, we focus all our experimental studies of the main paper on a VGG-16 two-stream fusion model [3] that is trained on UCF-101. Our visualization technique, however, is generally applicable to any spatiotemporal architecture. In this section, we visualize various other architectures: Spatiotemporal Residual Networks [2] using ResNet50 [5] streams, Temporal Segment Networks [13] using BN-Inception [6] or Inception_v3 [11] streams, trained on multiple datasets: UCF101 [9], HMDB51[7] and Kinetics [1]. All results in this block (B) are shown for constant spatiotemporal regularization during reconstruction.

It is noteworthy that deeper models (*e.g.* ResNet & Inception_v3) are inherently harder to optimize, since we do not use batch normalization (BN) for activation maximization; essentially, we absorb the BN layers by projecting them into the preceding conv-layer weights. Moreover, for extremely deep nets, the jitter-based regularization becomes more important for filters that have large strides on the input. Deeper layers typically have large cumulative filter strides when backprojected to the input, which causes a subsampling effect when optimizing that can be ameliorated by the spatial jittering during optimization.

B.1. Visualization of Inception networks

In this section we show results for models based on a batch-normalized GoogLeNet Inception architecture [6]. An interactive web-based illustration of the GoogLeNet (ImageNet) architecture, showing the layer names with respective number of parameters (ch) and output resolution (width×height) for a 227×227 sized input, can be found here. The models are taken from the Temporal Segment Networks approach [13], which pretrains the motion stream on TVL1 optical flow [14], IDT-flow [12] (termed warped flow in [13]) and difference images. This extra data yields highest accuracy of the motion stream in UCF101 and HMDB51. From Fig. B.1 to Fig. B.24, we show alternatively, for HMDB51 and for UCF101, what the convolutional filters of the motion and appearance stream are capturing. More concretely, Fig. B.1, shows what the first layers of the HMDB51 networks capture while Fig. B.2 shows what the corresponding UCF101 networks capture *etc*. When going from low to higher layers, we observe two things: First that the filters become more task specific (HMDB51 *vs*. UCF101) which is as expected; for example, the motion filter f006 at a higher conv-layer shown in Fig. B.23(HMDB51) and in Fig. B.24(UCF101) is activated by an optical flow pattern that could relate to the "pull-up" action in HMDB51, whereas the filter f006 for the motion stream trained on UCF101 could relate to the "IceDancing" class. Second, we see that the inputs for the appearance and motion streams are spatially similar at early layers and become dissimilar later. This suggests that the prior of the (ImageNet) pretraining on the spatial shape of the filters is dominant in the lower layers of the networks.

For these class level units shown in Fig. B.25-B.27 we see clearly recognizable pattern structure. More generally, when going from the first layer (Fig. B.1) to the last layer (Fig. B.25) we see that the features become more and more task-specific and abstract as we progress through the network, with units at early layers showing not much recognizable structure in the visualizations, while units at the classification layer are showing clear structure that matches to their corresponding class.

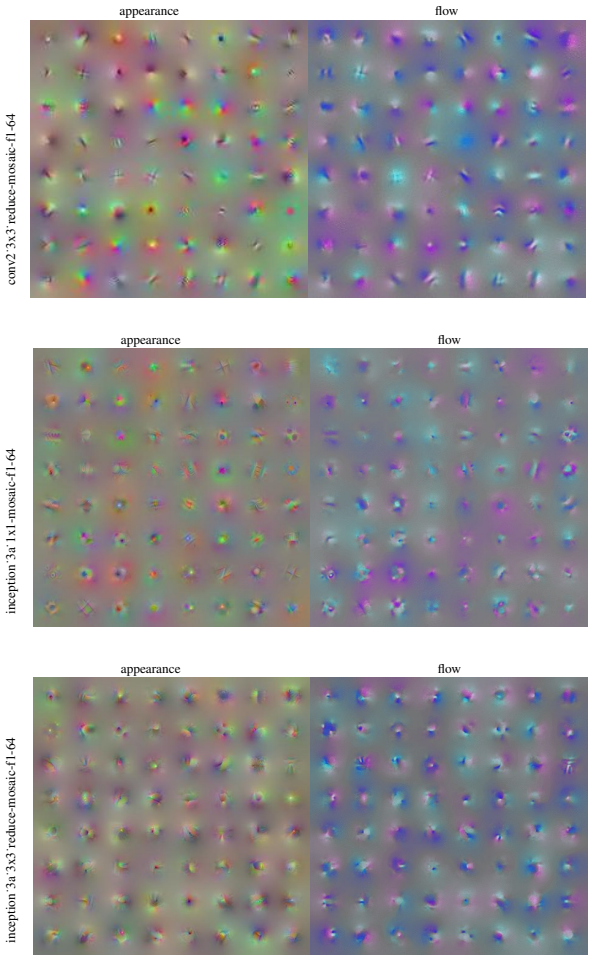


Figure B.1. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

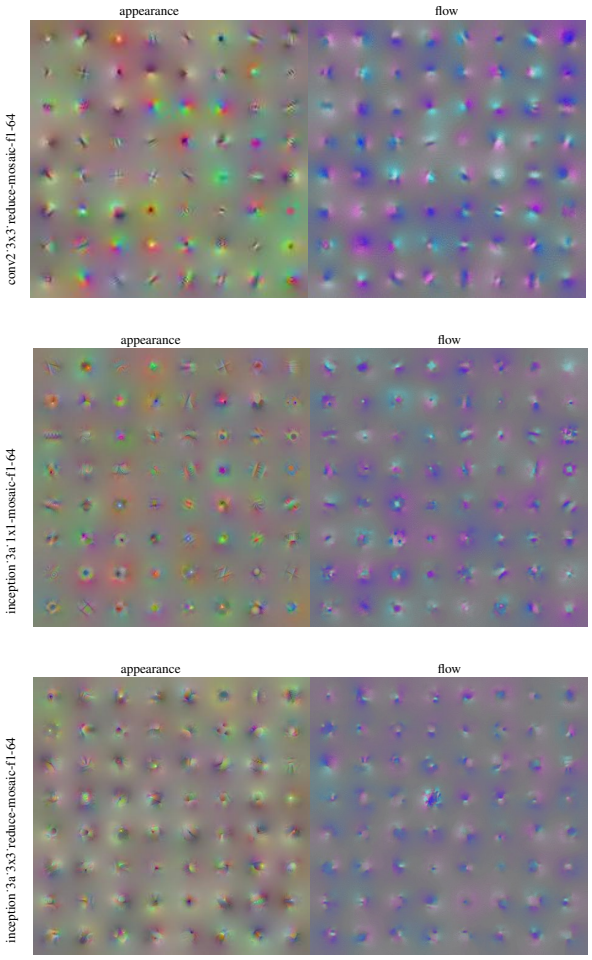


Figure B.2. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

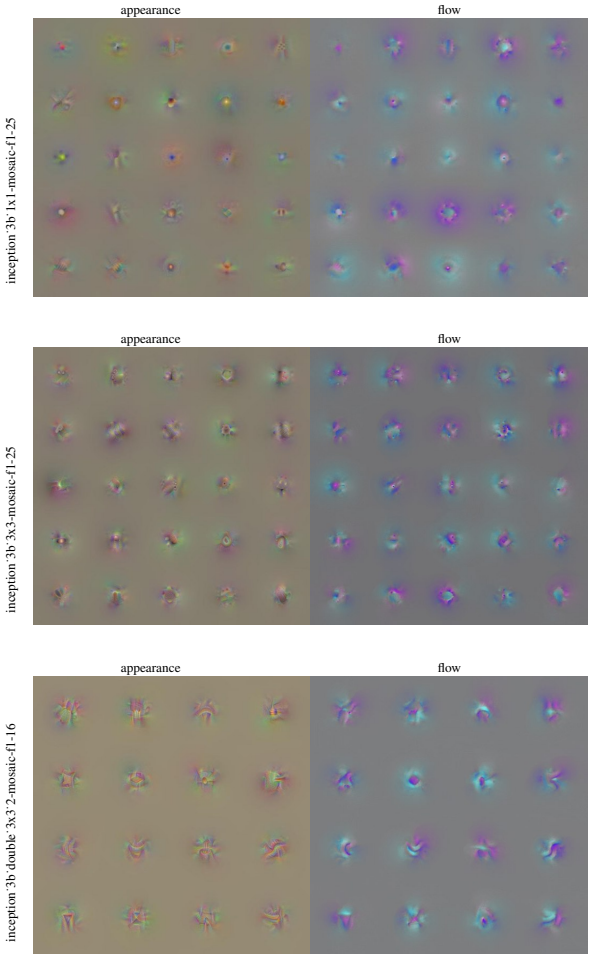


Figure B.3. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

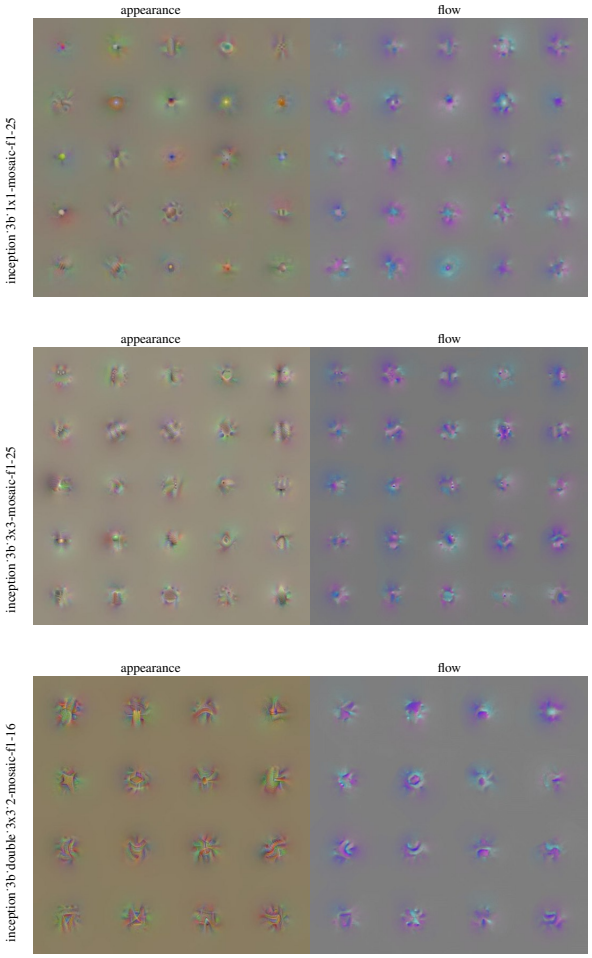


Figure B.4. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

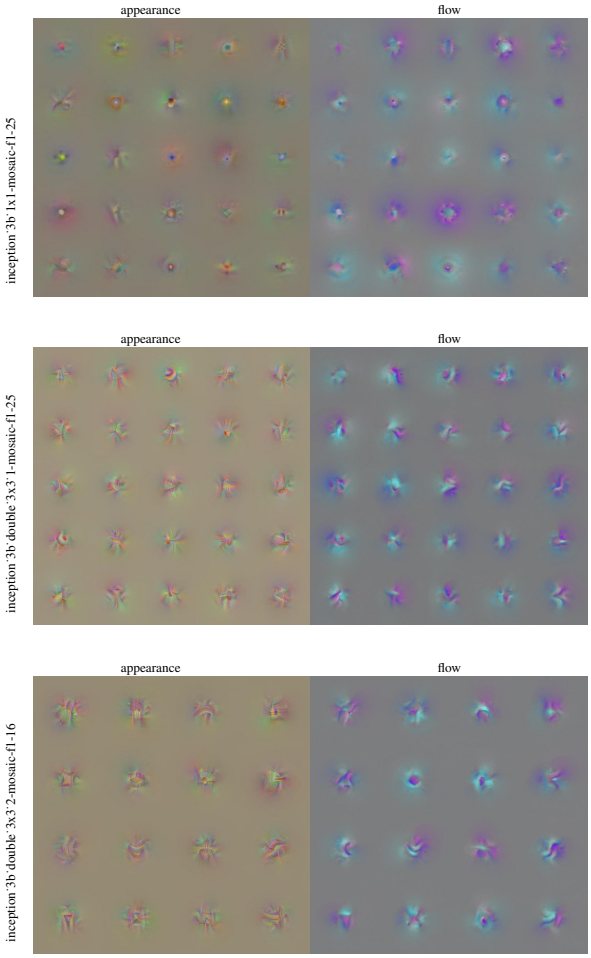


Figure B.5. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

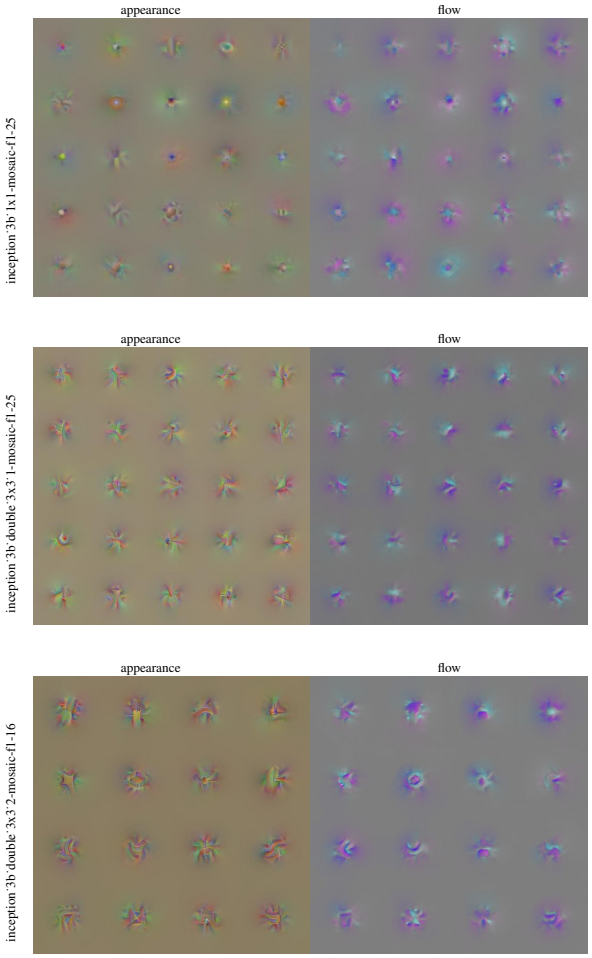


Figure B.6. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

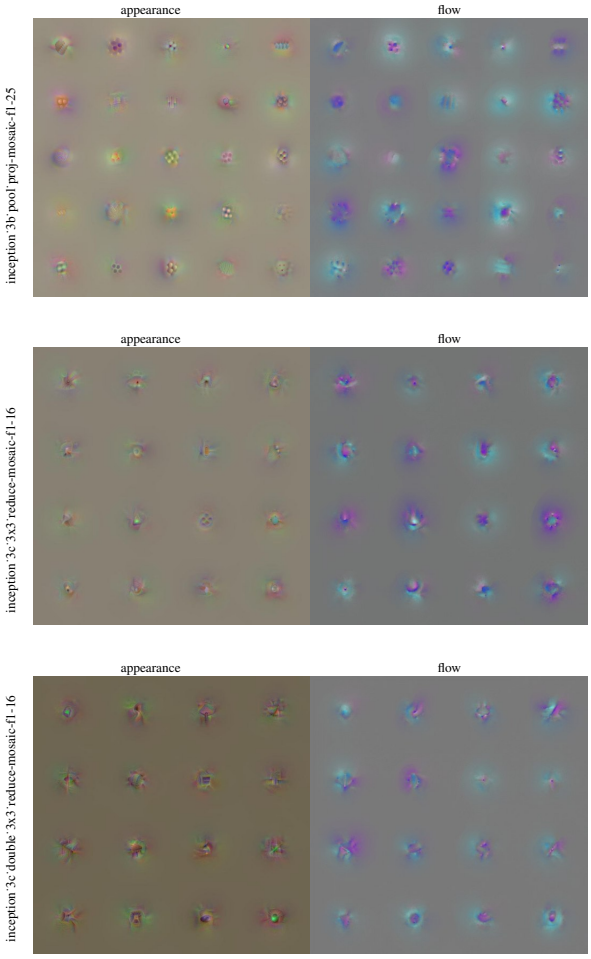


Figure B.7. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

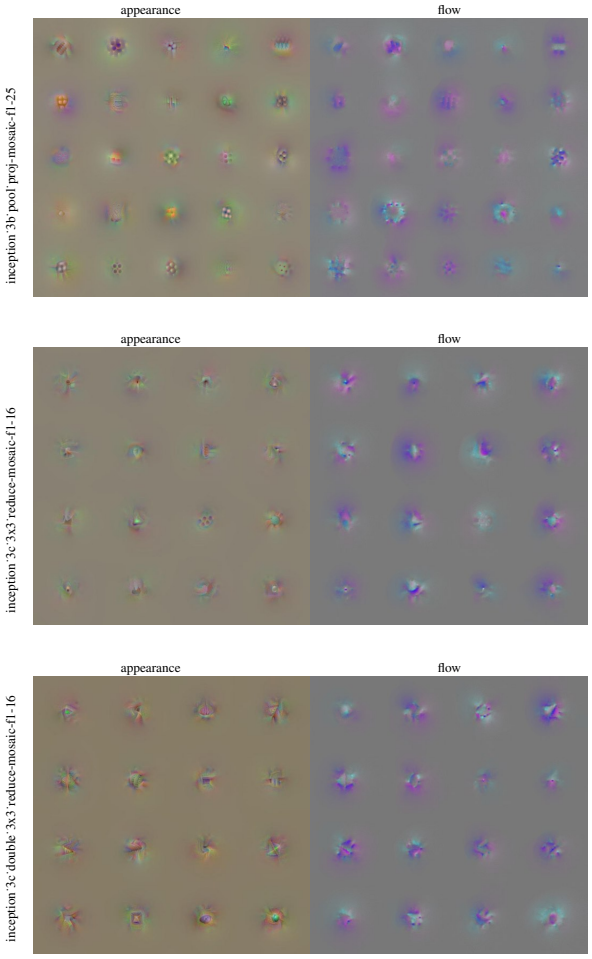


Figure B.8. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

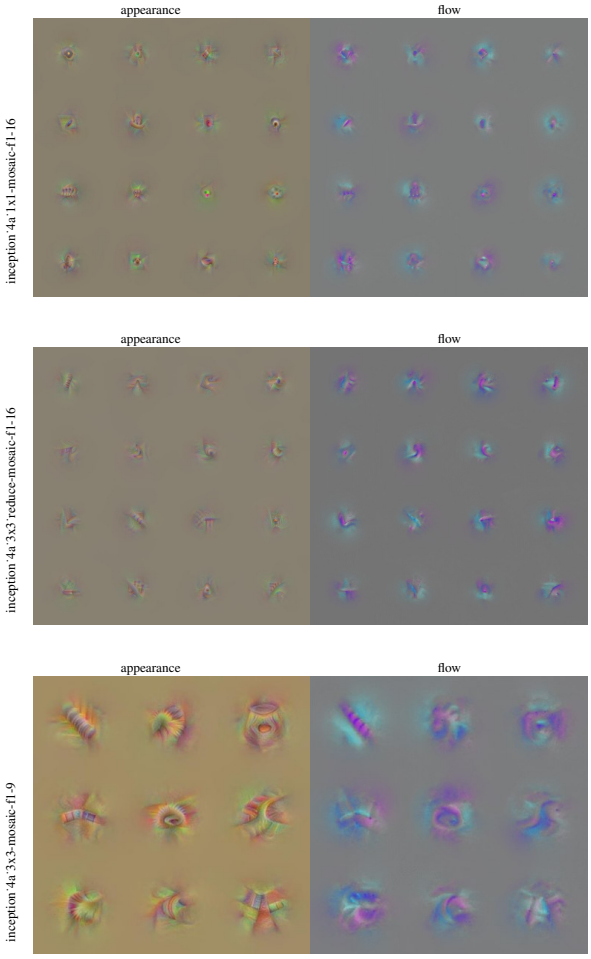


Figure B.9. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

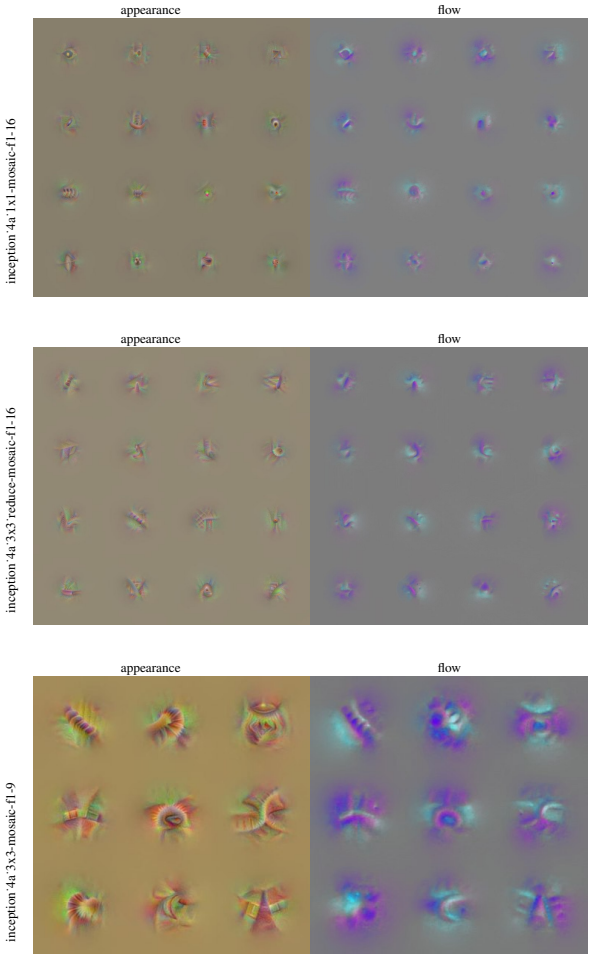


Figure B.10. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

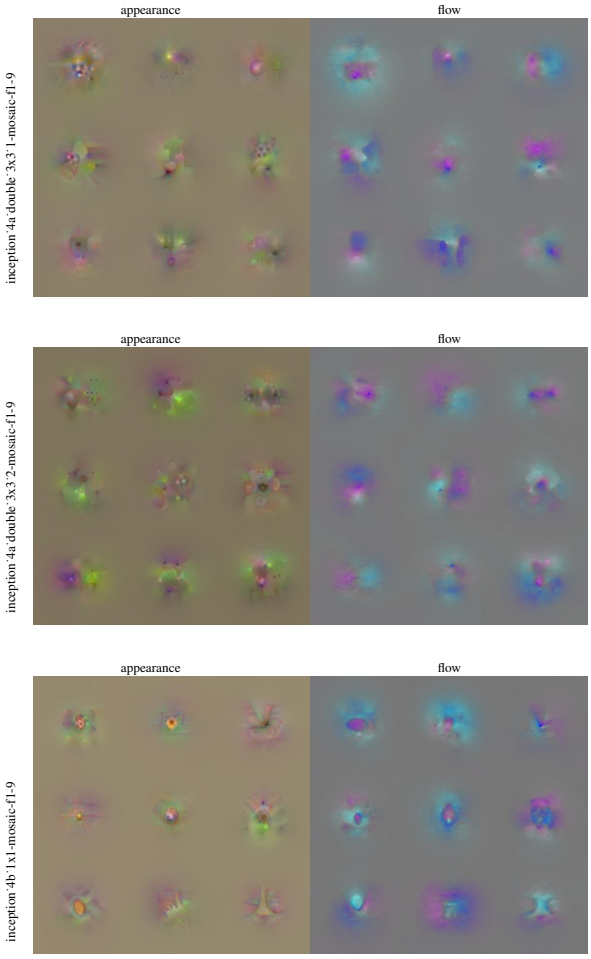


Figure B.11. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

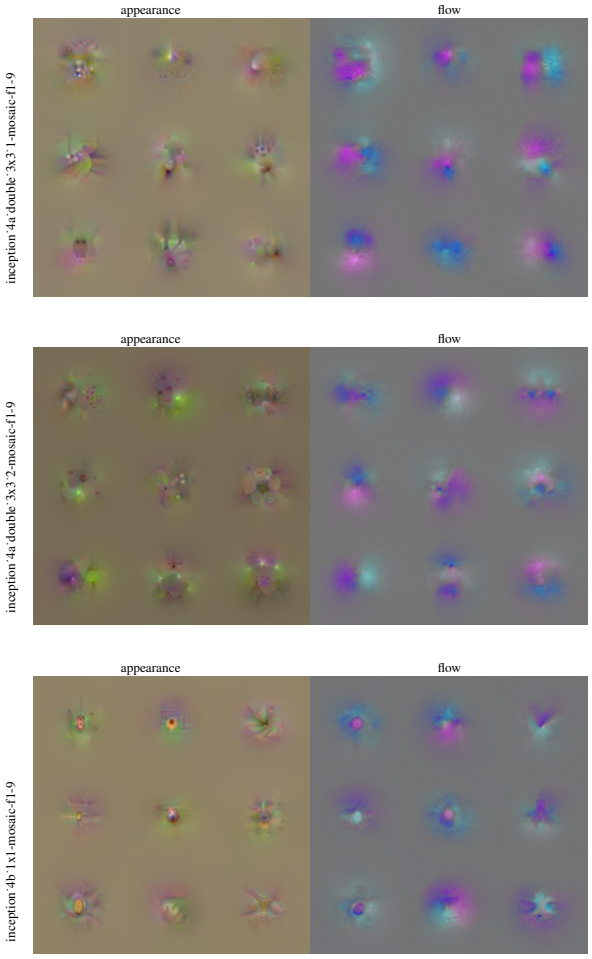


Figure B.12. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

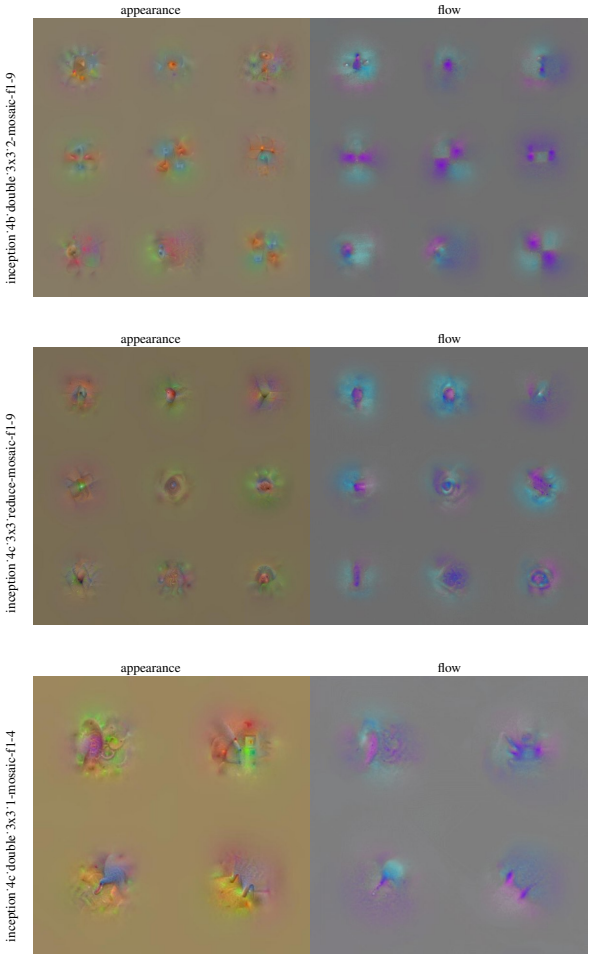


Figure B.13. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

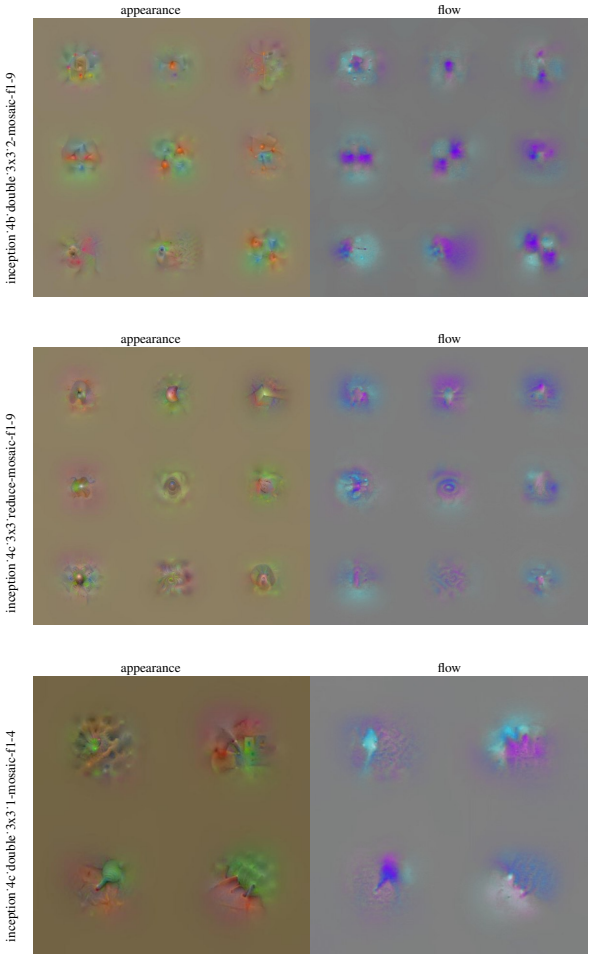


Figure B.14. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

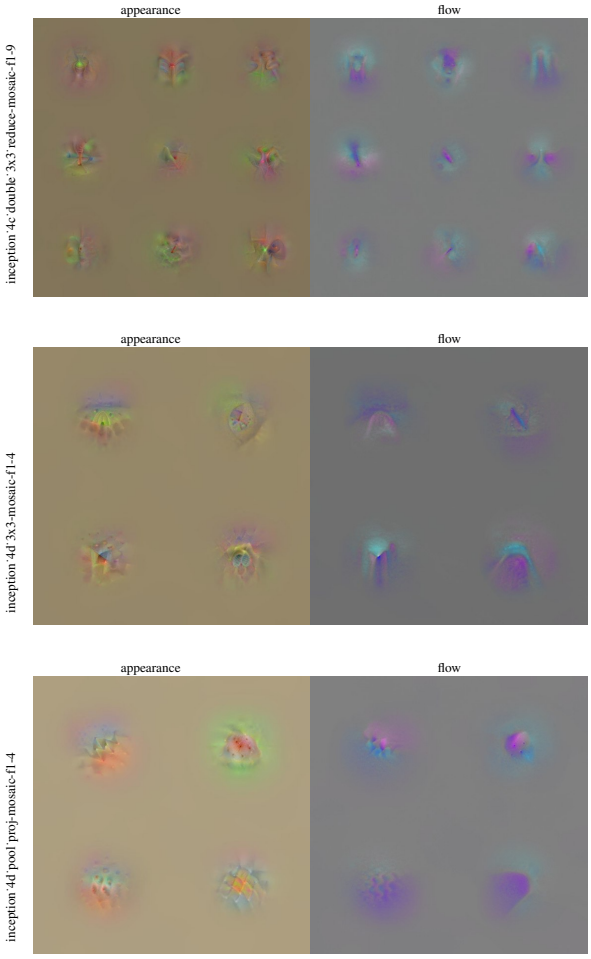


Figure B.15. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

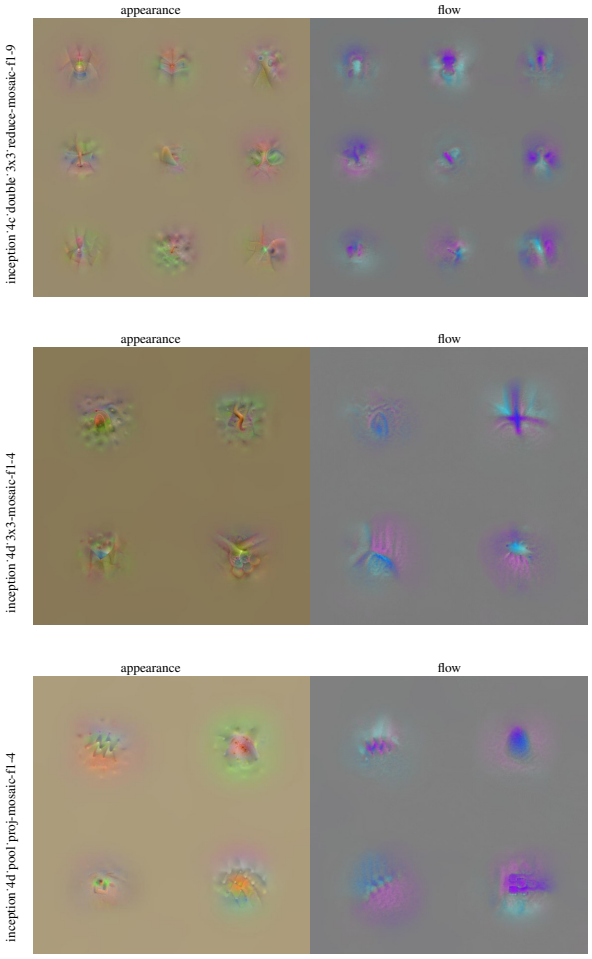


Figure B.16. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

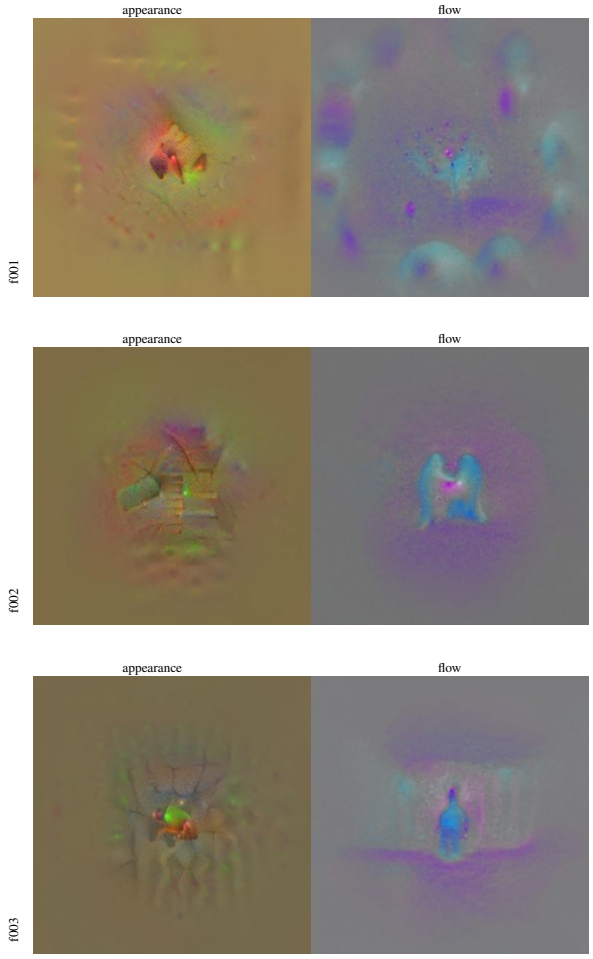


Figure B.17. Visualizations of convolutional layer inception˙5b˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

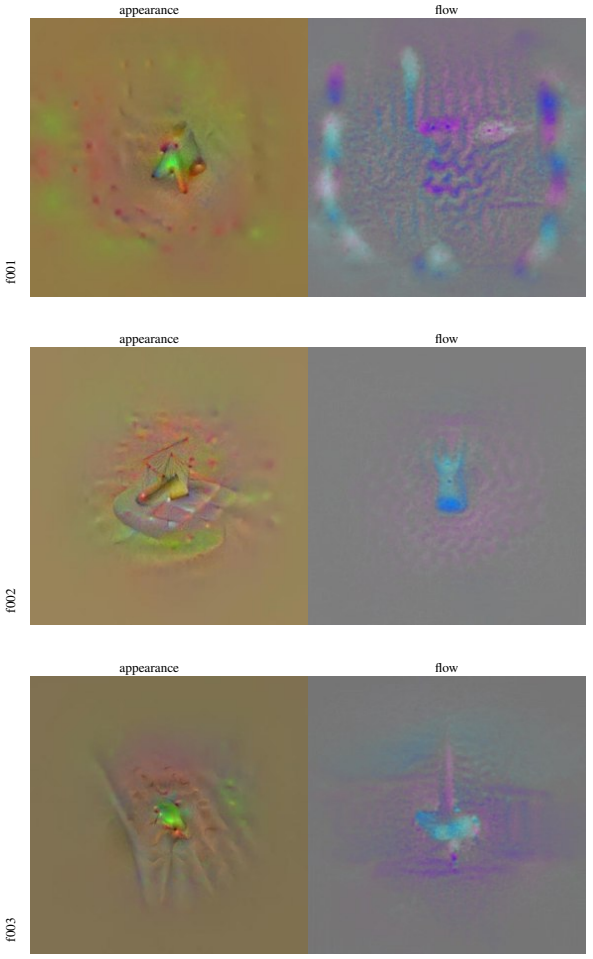


Figure B.18. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

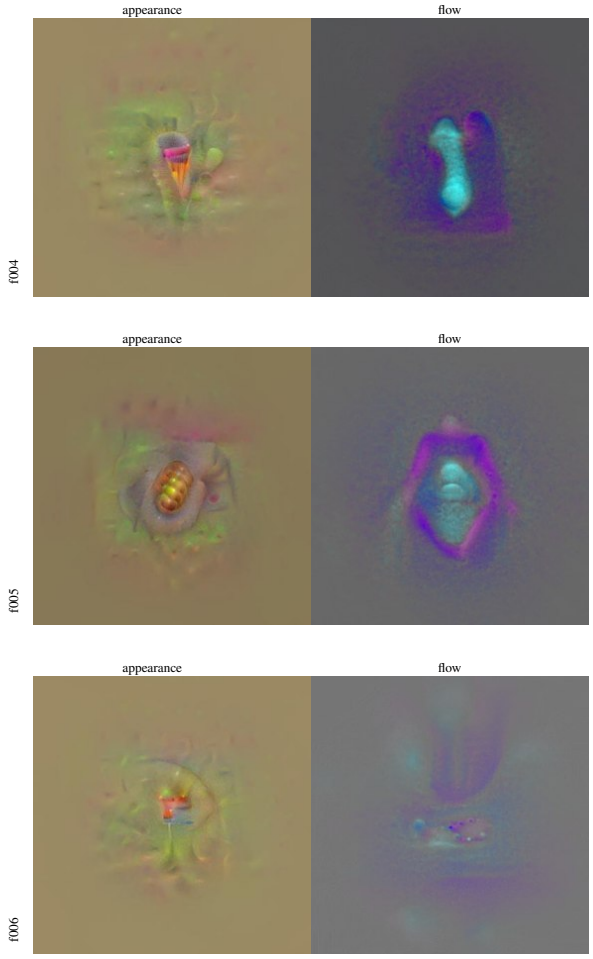


Figure B.19. Visualizations of convolutional layer inception˙5b˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13] trained on HMDB51 [7]. The number of filters shown at each layer is inversely proportional to its receptive field.

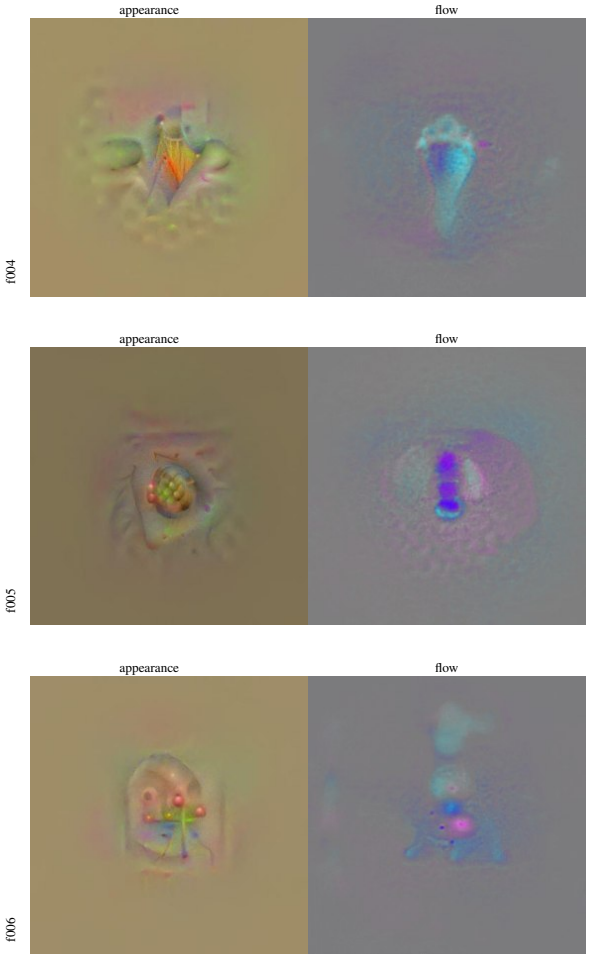


Figure B.20. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

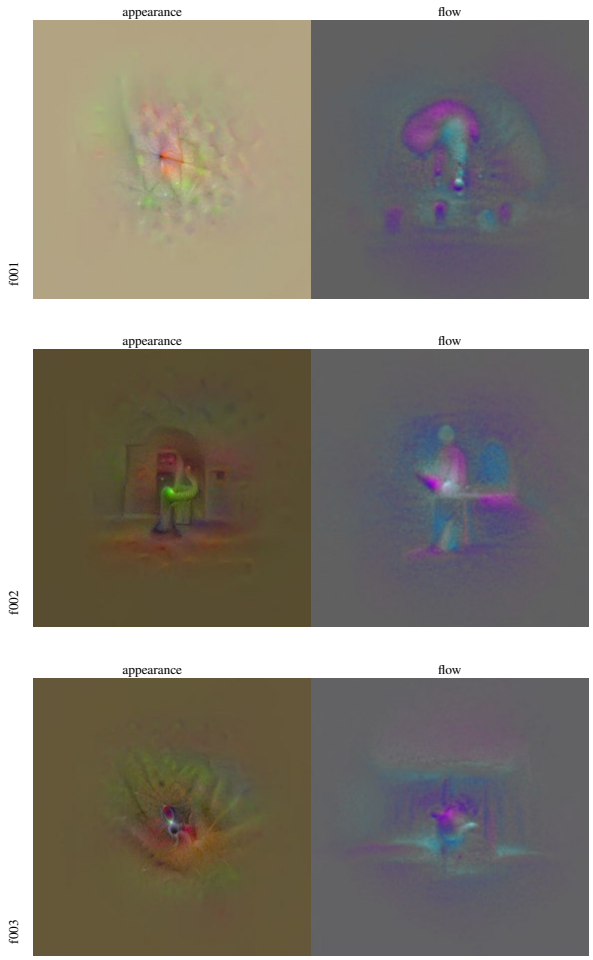


Figure B.21. Visualizations of convolutional layer inception˙5b˙double˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

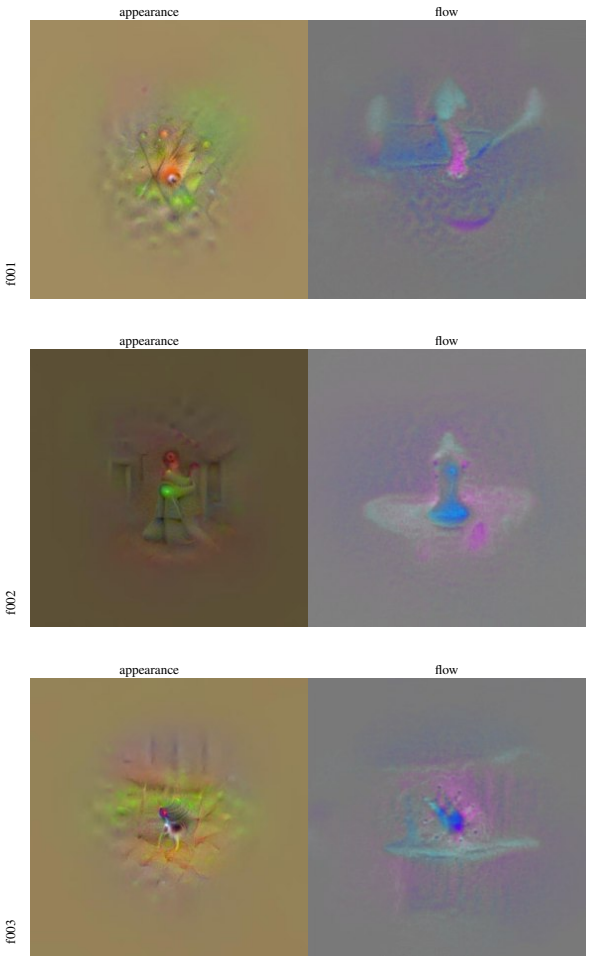


Figure B.22. Visualizations of convolutional layers of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

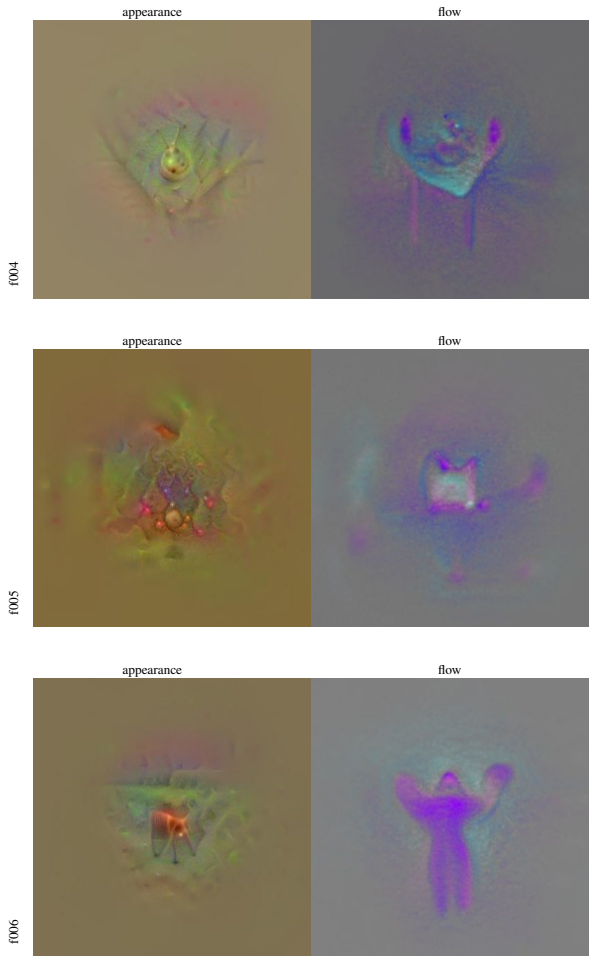


Figure B.23. Visualizations of convolutional layer inception˙5b˙double˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. The number of filters shown at each layer is inversely proportional to its receptive field.

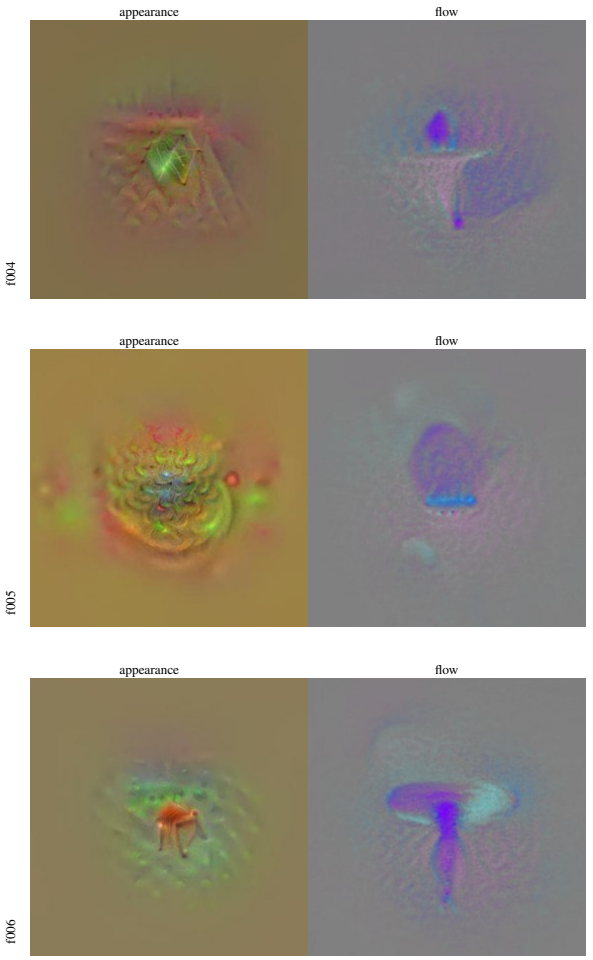


Figure B.24. Visualizations of convolutional layer inception˙5b˙double˙3x3˙reduce of the BN-Inception [6] Two-Stream network [13] trained on UCF101 [9]. The number of filters shown at each layer is inversely proportional to its receptive field.

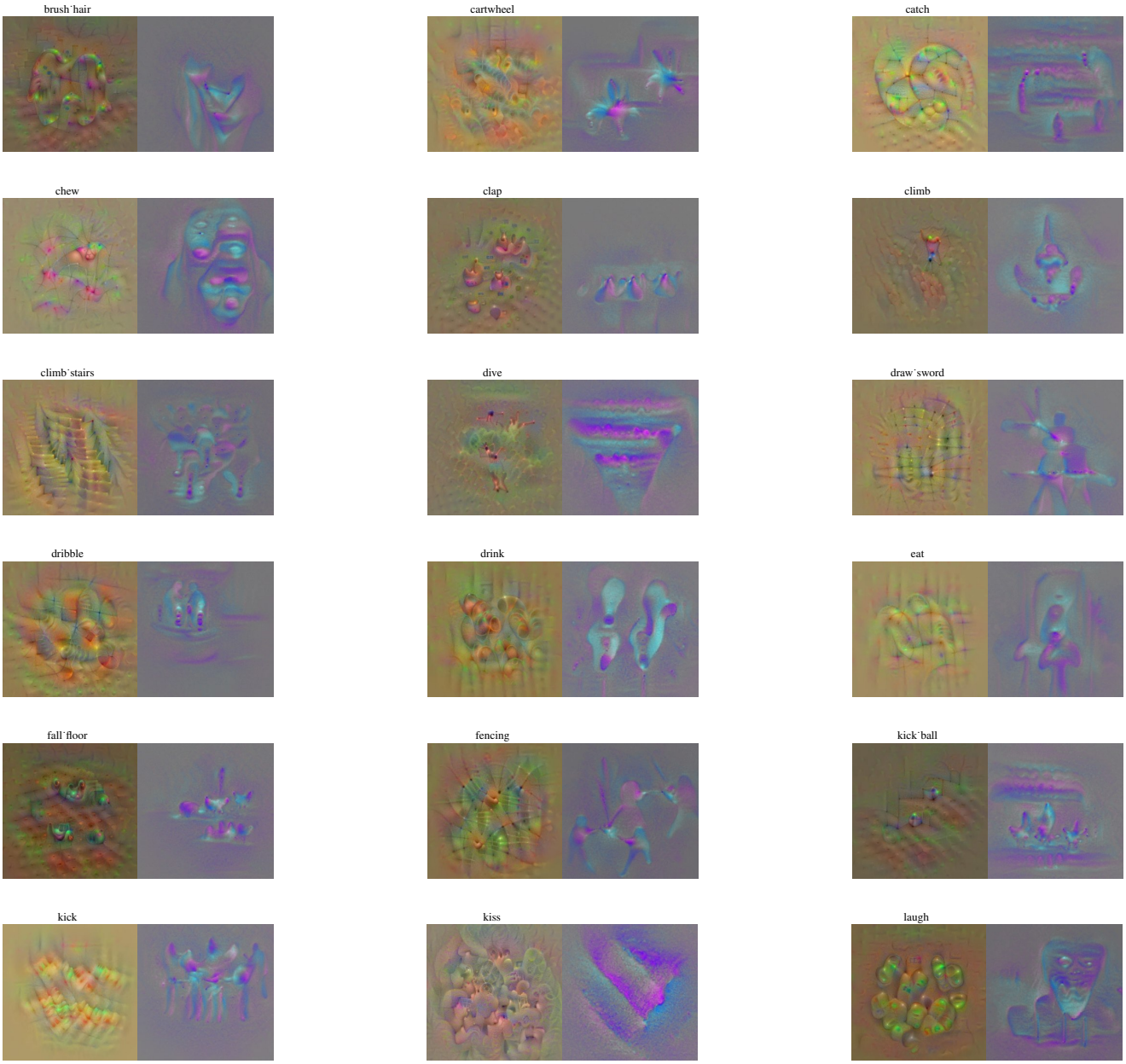


Figure B.25. Visualizations of classification units at the last layer of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above.

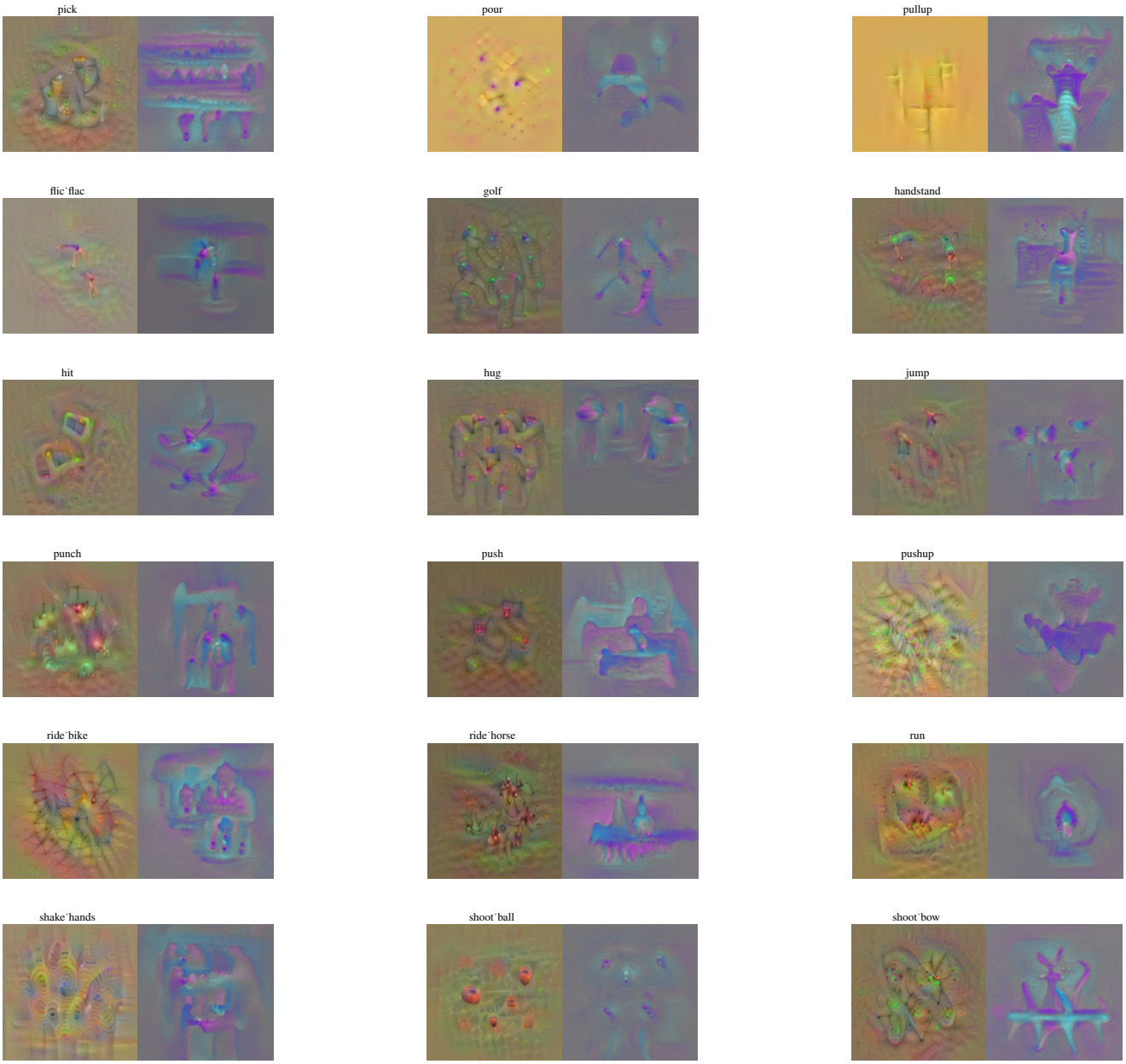


Figure B.26. Visualizations of classification units at the last layer of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above.

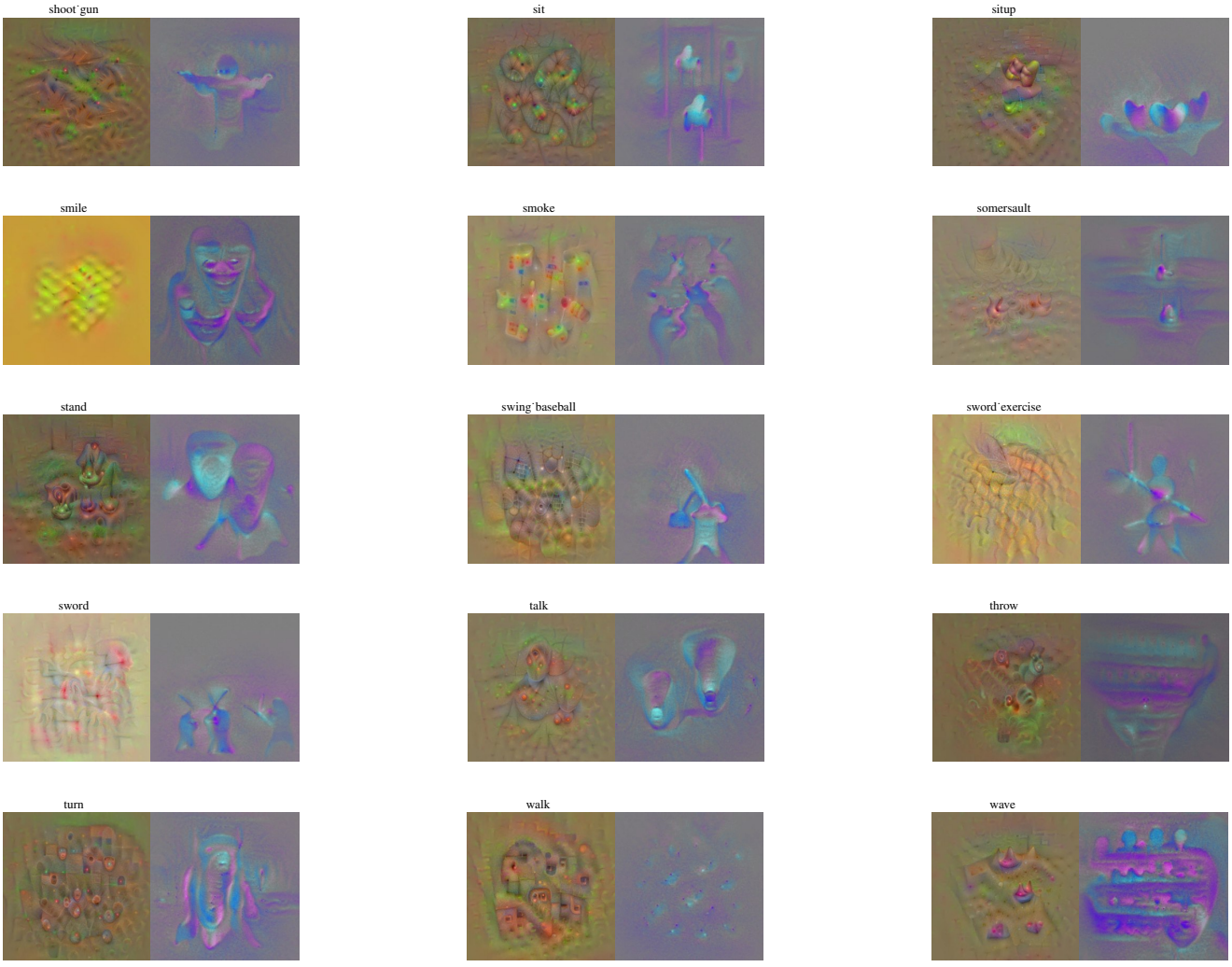


Figure B.27. Visualizations of classification units at the last layer of the BN-Inception [6] Two-Stream network [13] trained on HMDB51[7]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above.

B.2. Visualization of Spatiotemporal Residual Networks

We now show results of our approach applied for visualization of the filters in a Spatiotemporal Residual Network [2] trained on UCF101. This architecture uses two ResNet50 [5] streams. An interactive web-based schematic of the ResNet50 (ImageNet) architecture, showing the layer names with respective number of parameters (ch) and output resolution (width×height) for a 224×224 sized input, can be found here.

It is noteworthy that the visualized Spatiotemporal Residual Network [2] architecture consists of a fusion stream that operates on appearance and motion information, as well as a pure motion stream that projects into the fusion stream at five layers (*i.e.* res2a, res3a, res4a, res5a) throughout the network hierarchy (the detailed architecture and connections between the two ResNet50 streams is available in [2]). Therefore, it is interesting to study the filters in the fusion stream and compare them to the parallel motion filters in the other stream. We show results for multiple layers throughout the network hierarchy from Fig. B.28 to Fig. **??**. Please click on the fusion filters (first column) to see the motion and appearance input played as a video; the motion stream filters should automatically play on repeat. When looking at the filters, we see interesting abstract patterns appearing that show strong spatiotemporal correlation for the fusion filters, while the motion stream (right column), which projects as a residual connection into the parallel fusion stream, is maximized by slightly different motion patterns. This qualitatively shows that the fusion stream learns spatially correlated patterns of flow and appearance. Also interesting is the fact that the lower fusion stream layers, shown in Fig. B.28, exhibit only very weak motion information (noisy reconstructions); this can be seen as evidence for the need of the parallel motion stream to build up these abstract, strong motion features later on in the hierarchy of the network.

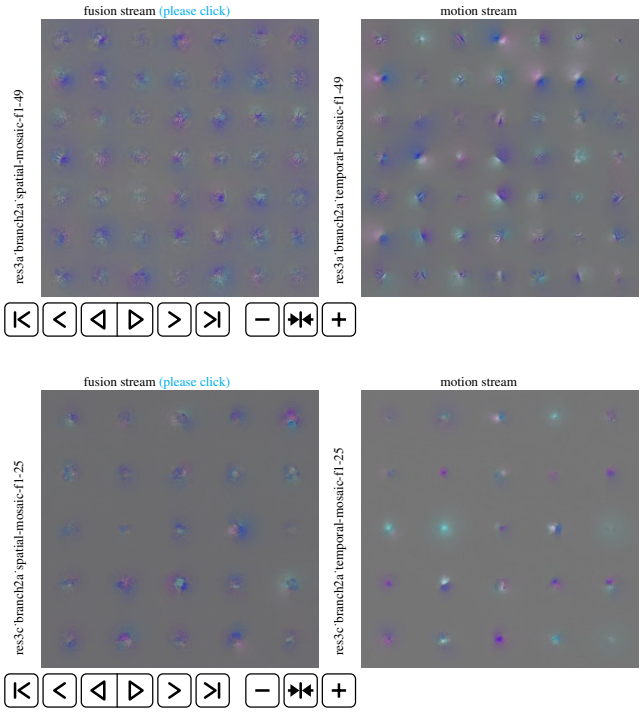


Figure B.28. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column spatiotemporal filters of the fusion stream with motion and appearance shown sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

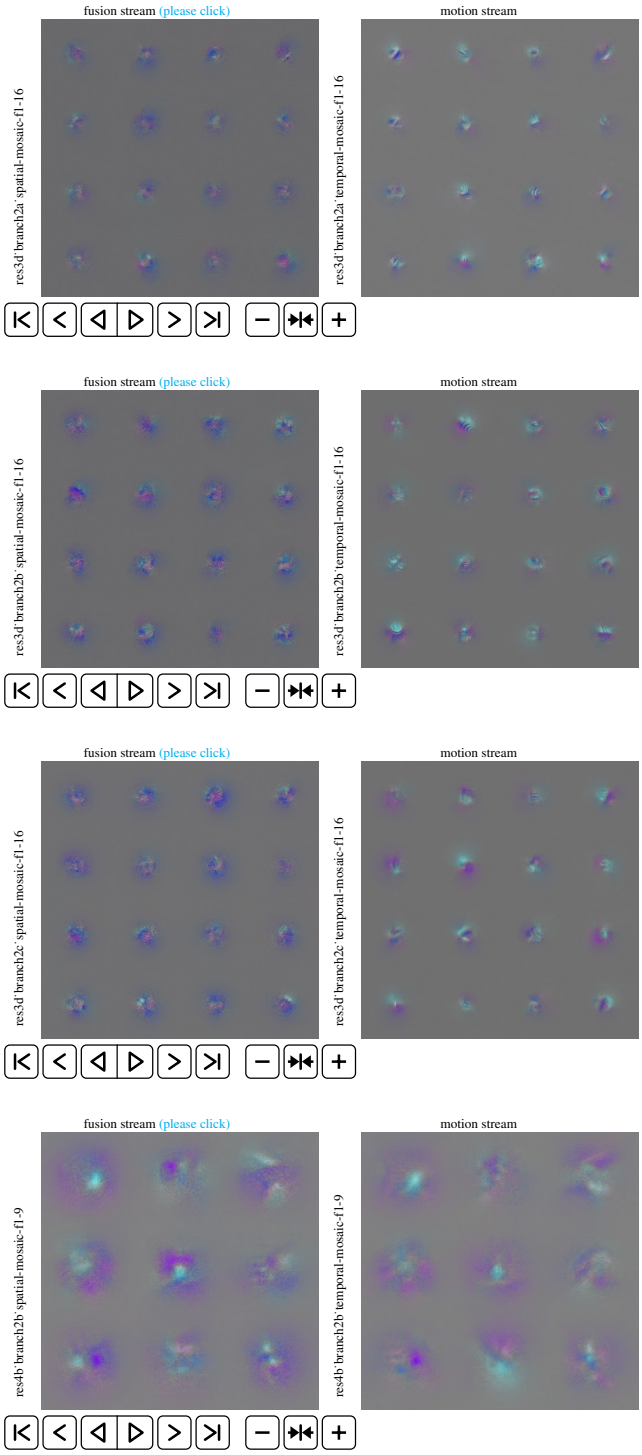


Figure B.29. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

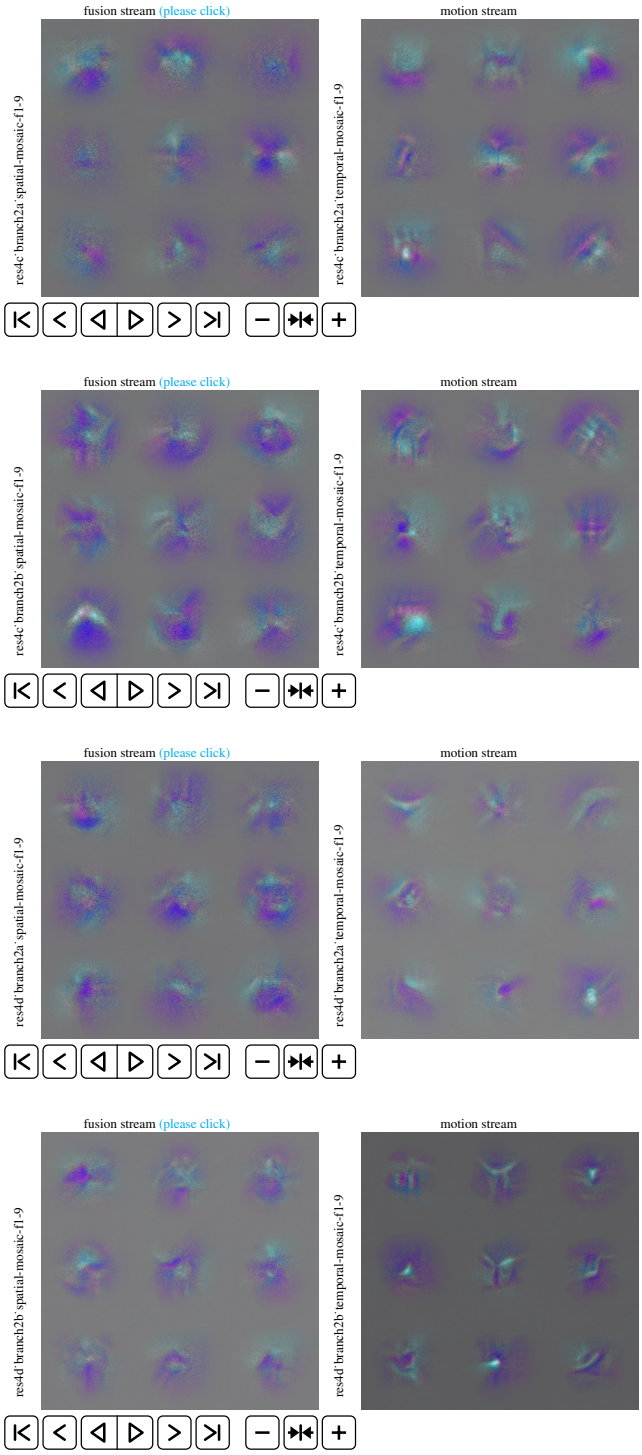


Figure B.30. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

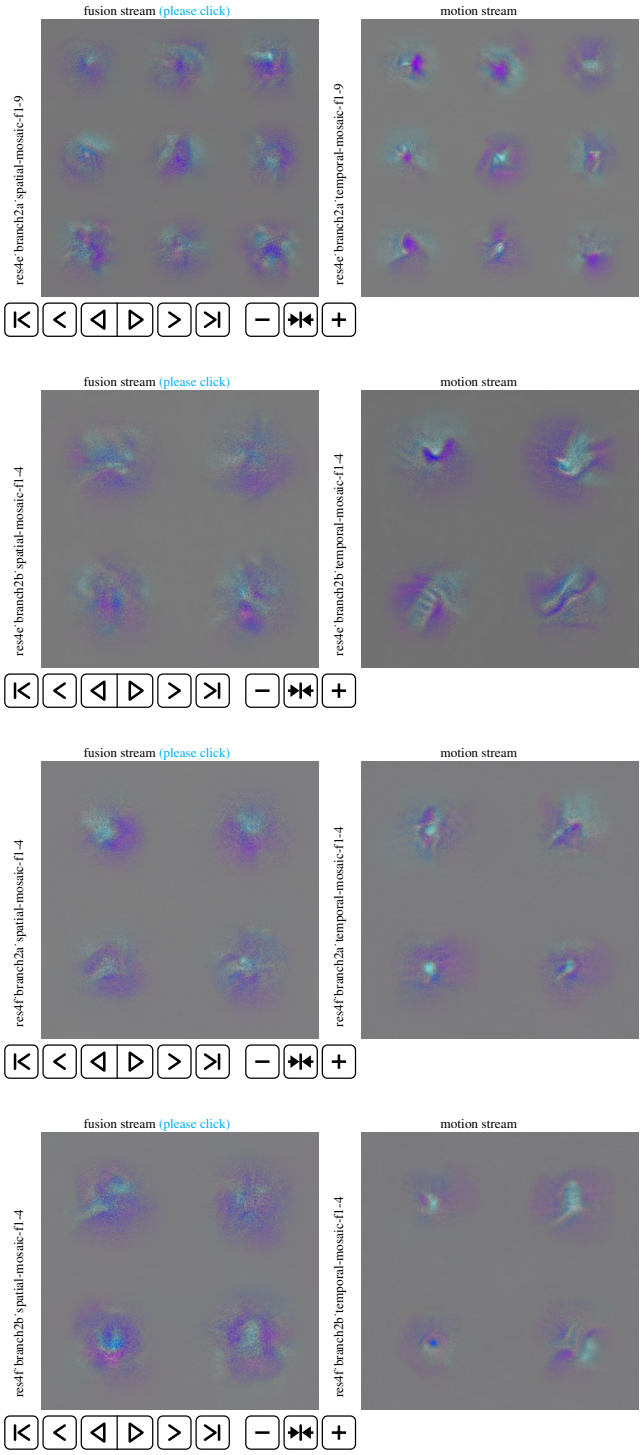


Figure B.31. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

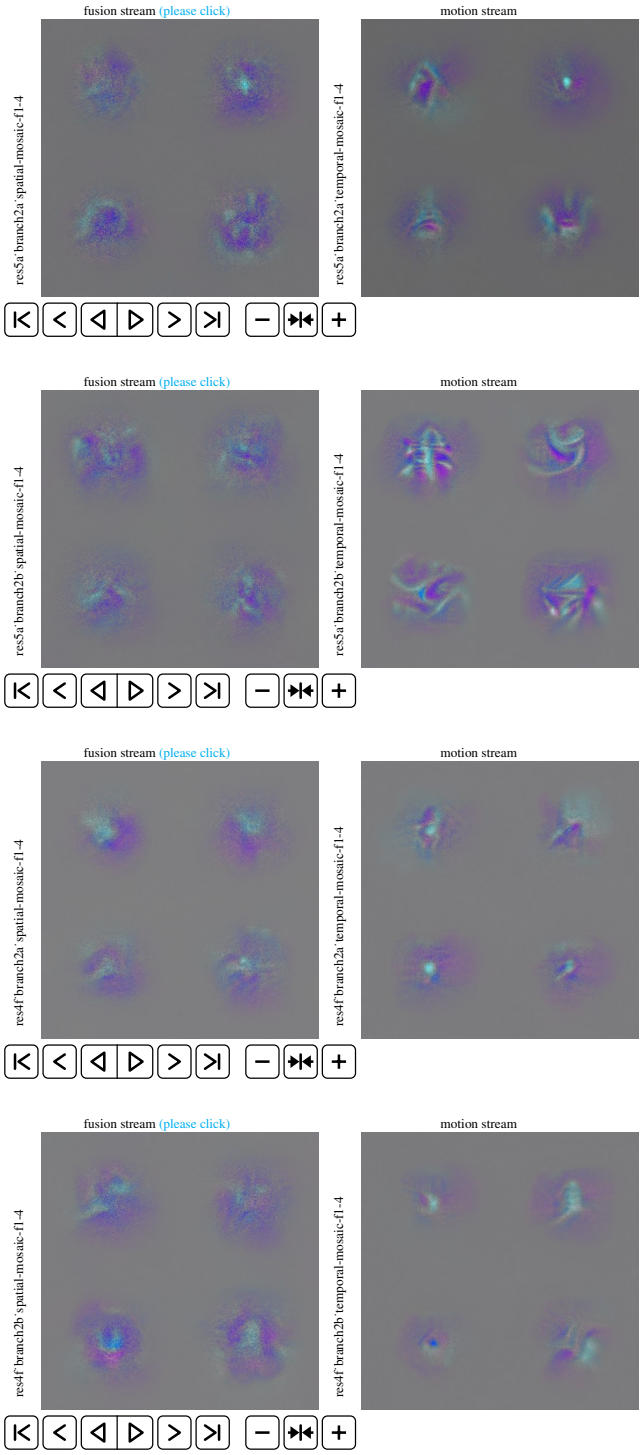
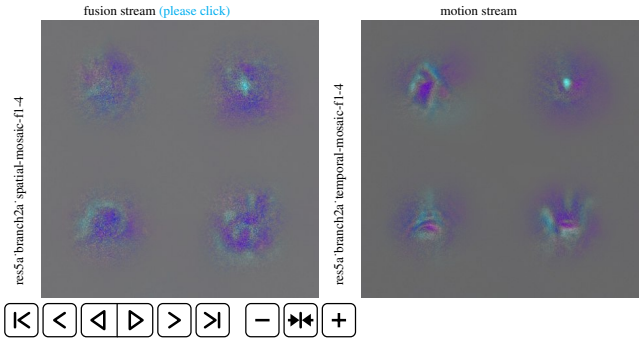
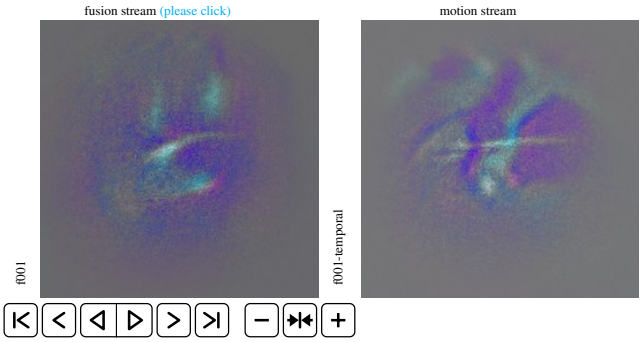
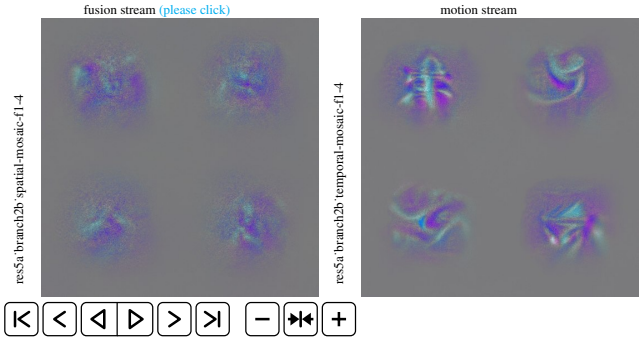
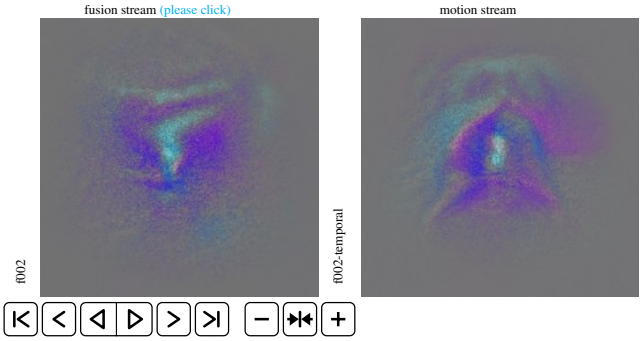


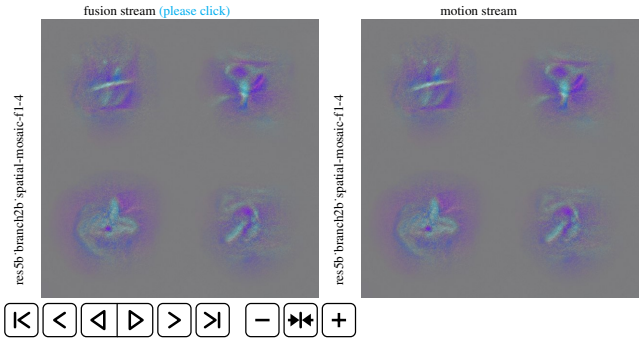
Figure B.32. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

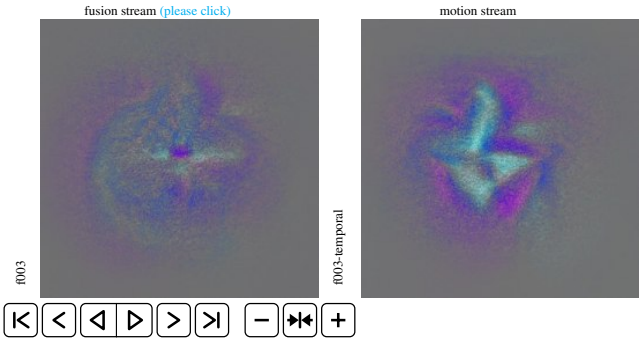


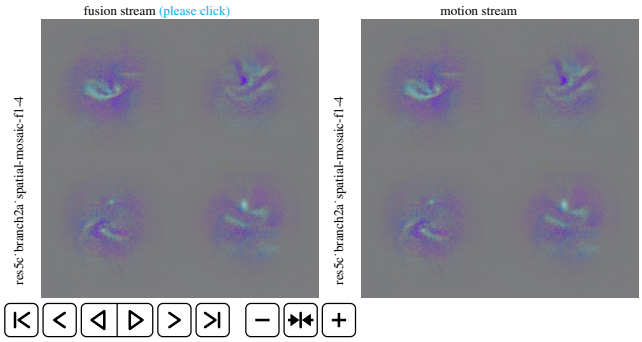












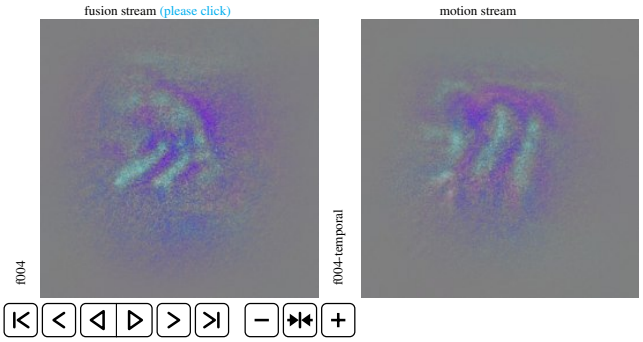
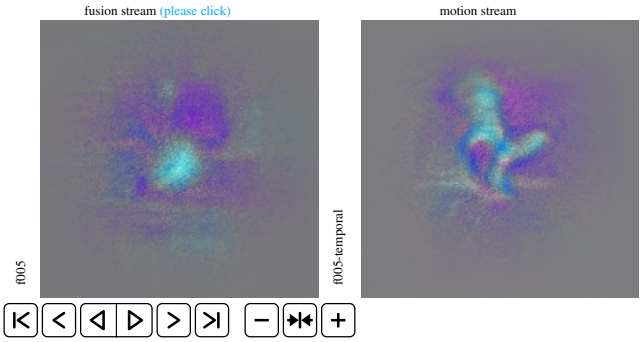
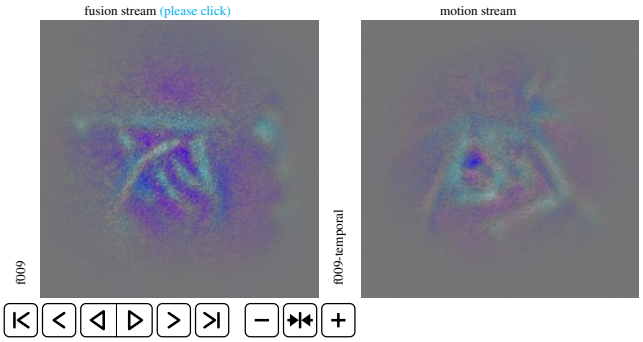
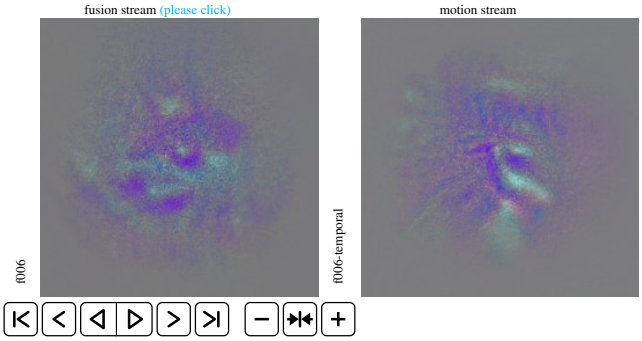


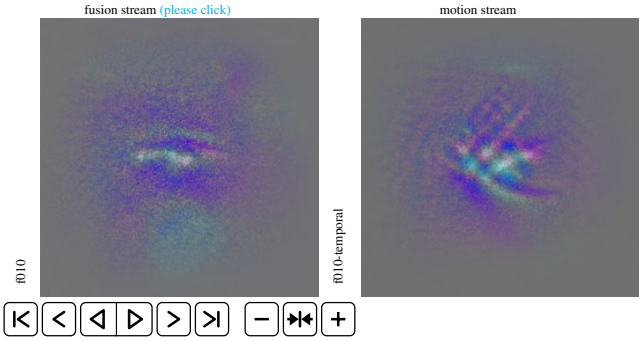
Figure B.33. Visualizations of convolutional layers of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback), whereas the right column shows the motion stream that projects into the fusion stream.

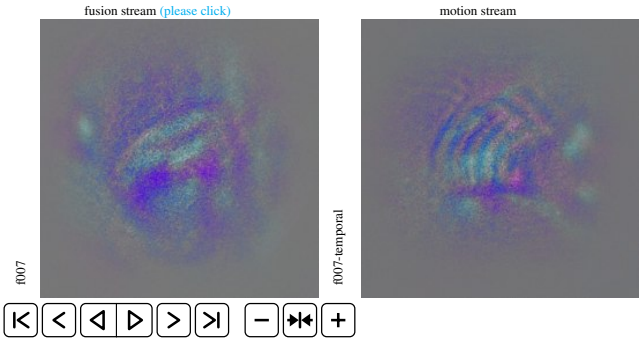
Figure B.34. Visualizations of convolutional filters at layer res5b˙branch2b of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback); the right column shows the motion stream that projects into the fusion stream.

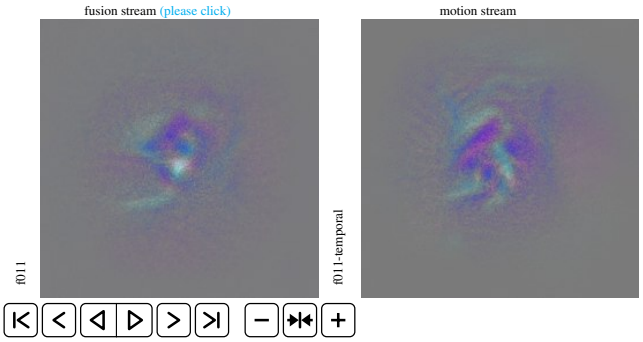


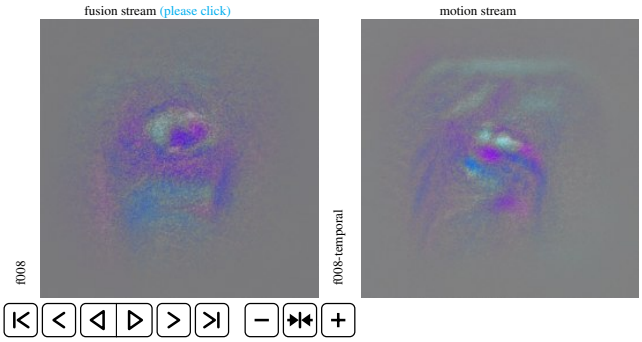












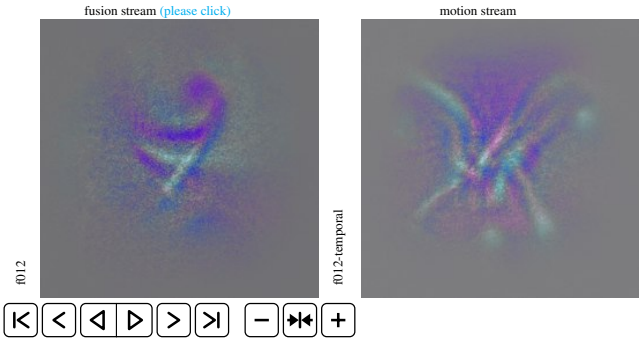


Figure B.35. Visualizations of convolutional filters at layer res5b˙branch2b of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback); the right column shows the motion stream that projects into the fusion stream.

Figure B.36. Visualizations of convolutional filters at layer res5b˙branch2b of Spatiotemporal Residual Networks [2] using ResNet50 [5] streams trained on UCF101 [9]. The left column shows spatiotemporal filters of the fusion stream with motion and appearance playing sequentially (click for playback); the right column shows the motion stream that projects into the fusion stream.

B.3. Visualization of Inception_v3 networks

We finally explore an Inception_v3 Two-Stream model that was trained on Kinetics [1]. An interactive web-based schematic of the Inception_v3 (ImageNet) architecture, showing the layer names with respective number of parameters (ch) and output resolution (width×height) for a 299×299 sized input, can be found here.

Fig. B.37 to Fig. B.41 show the convolutional filters at the intermediate layers of this network. Interestingly, similar patterns appear in the appearance and the optical flow path, even though these streams were not connected during training (*i.e*. no fusion between the streams). We think that the similarities in the spatial structures of the filters for appearance and motion, despite being trained separately from different input modalities, are due to similar initialization (*i.e*. via ImageNet). Finally, we also observe that temporal variation increases when going deeper in the network hierarchy. In Fig. B.42 to Fig. B.51 we show the class prediction units of these two streams. Please click on the optical flow plot to see the motion input playing as a video. Notably, Kinetics includes many actions that are hard to predict just from optical flow information[1].

[1] https://deepmind.com/research/open-source/open-source-datasets/kinetics/

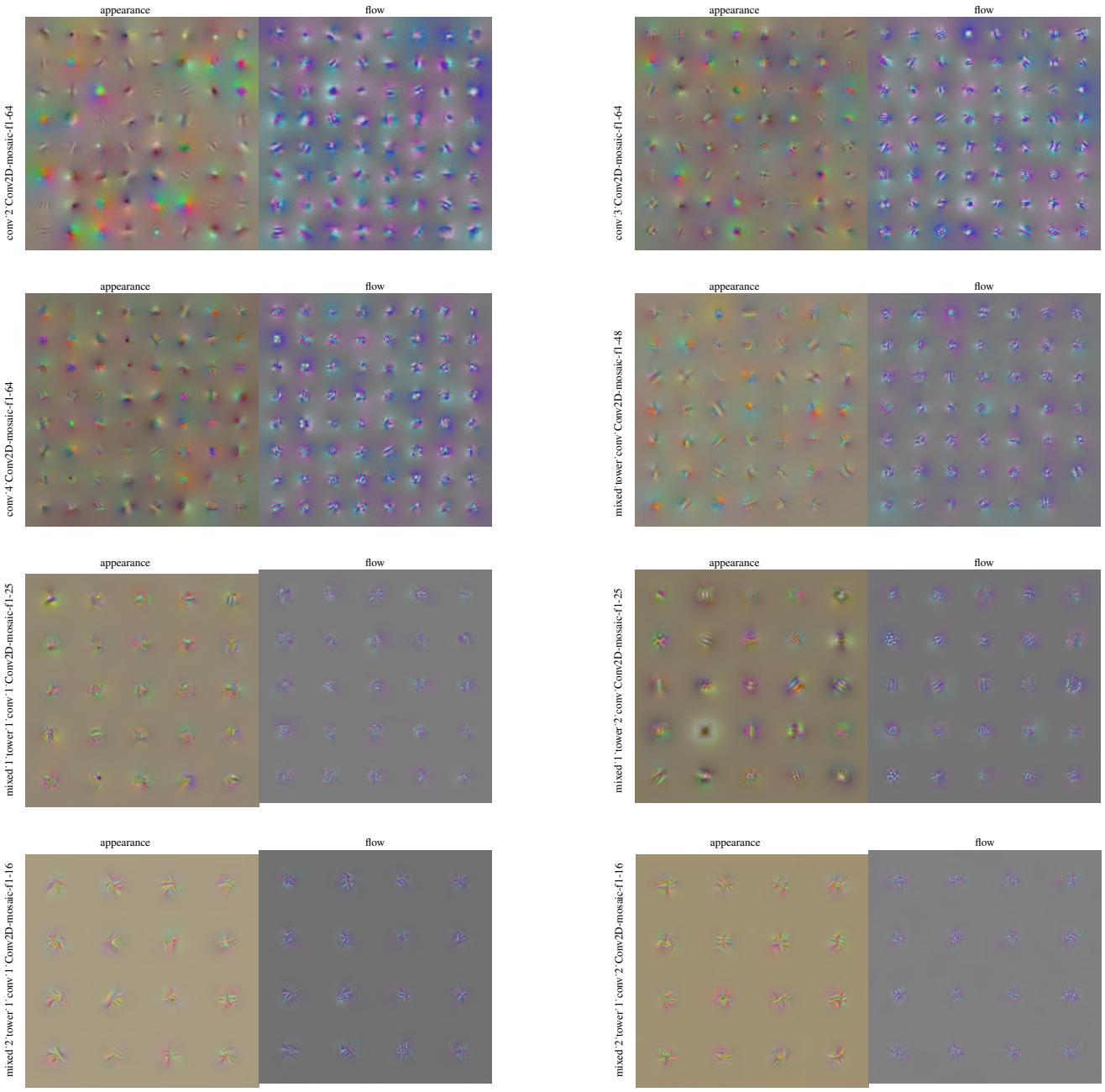


Figure B.37. Visualizations of convolutional layers of the Inception_v3 Two-Stream network [11] trained on Kinetics [1]. The number of filters shown at each layer is inversely proportional to its receptive field.

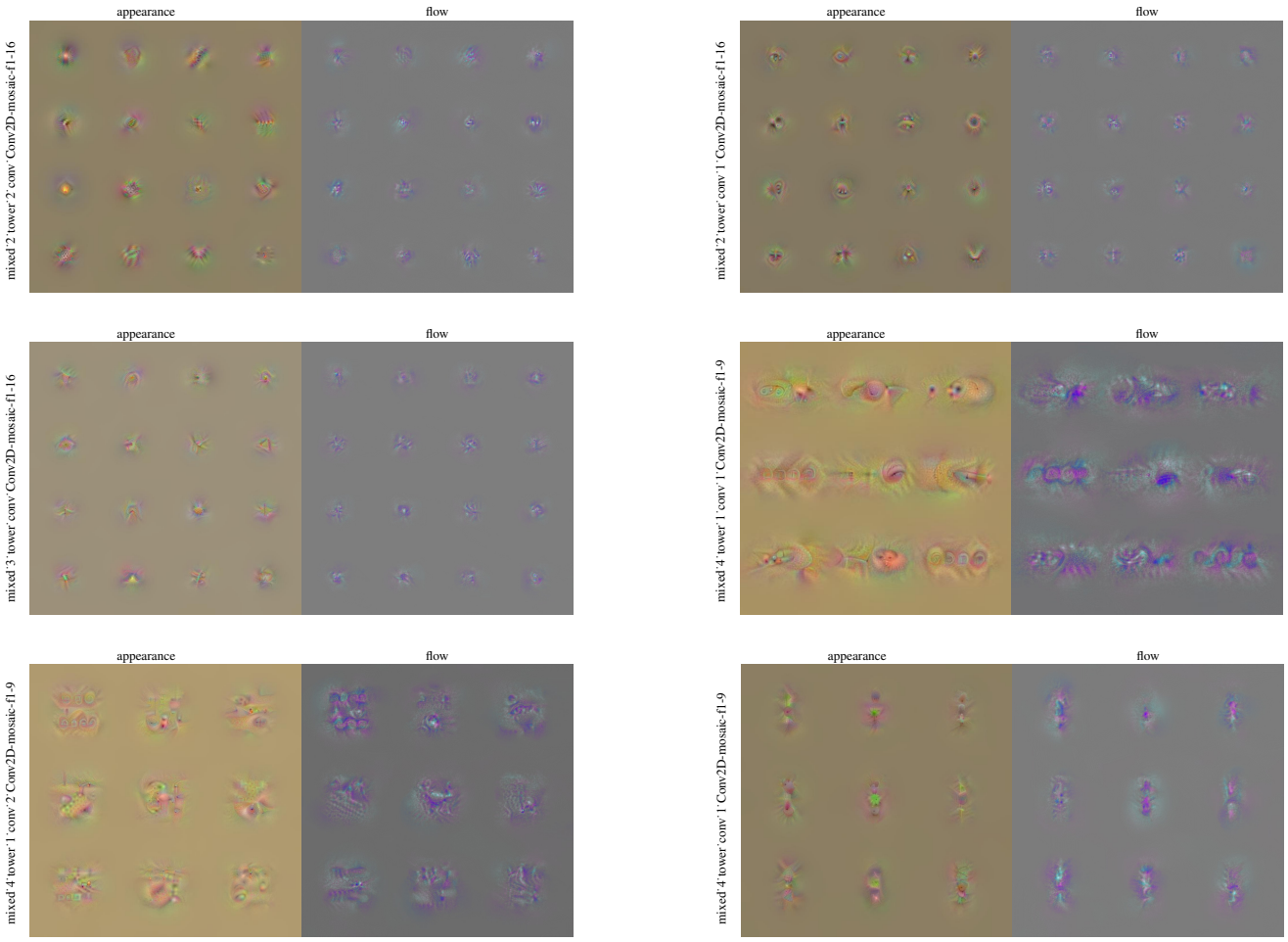


Figure B.38. Visualizations of convolutional layers of the Inception_v3 Two-Stream network [11] trained on Kinetics [1]. The number of filters shown at each layer is inversely proportional to its receptive field.

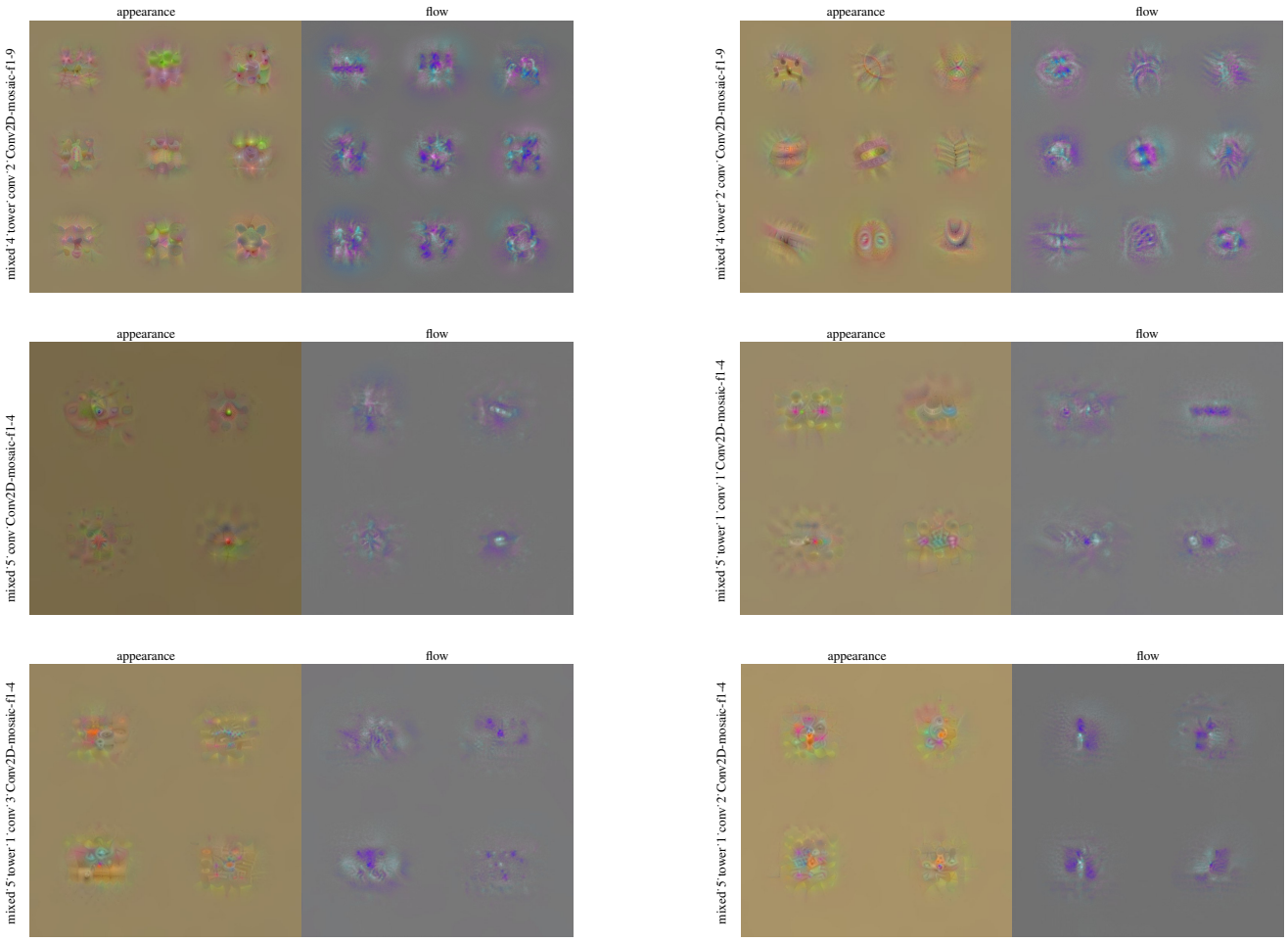


Figure B.39. Visualizations of convolutional layers of the Inception_v3 Two-Stream network [11] trained on Kinetics [1]. The number of filters shown at each layer is inversely proportional to its receptive field.

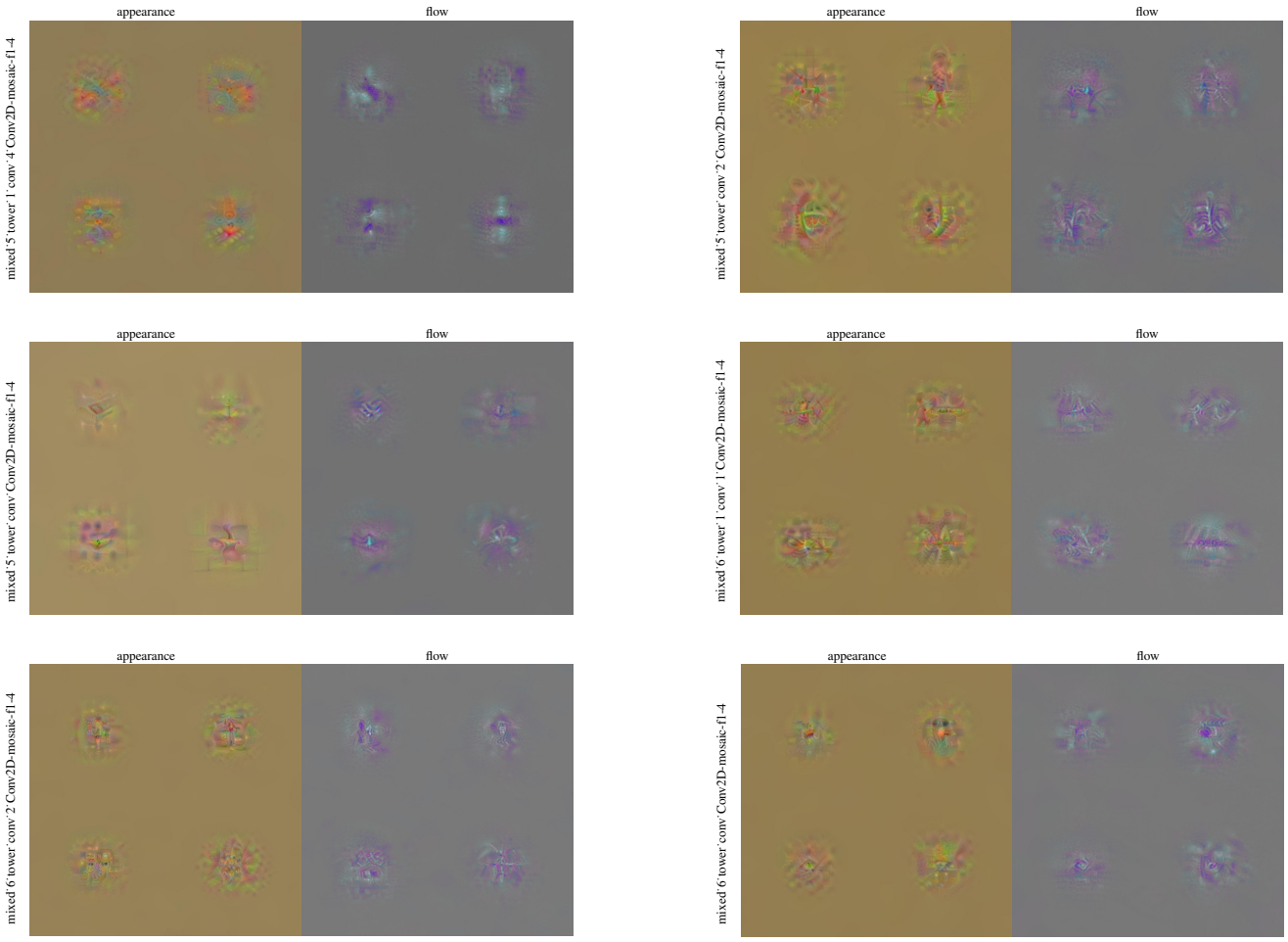


Figure B.40. Visualizations of convolutional layers of the Inception_v3 Two-Stream network [11] trained on Kinetics [1]. The number of filters shown at each layer is inversely proportional to its receptive field.

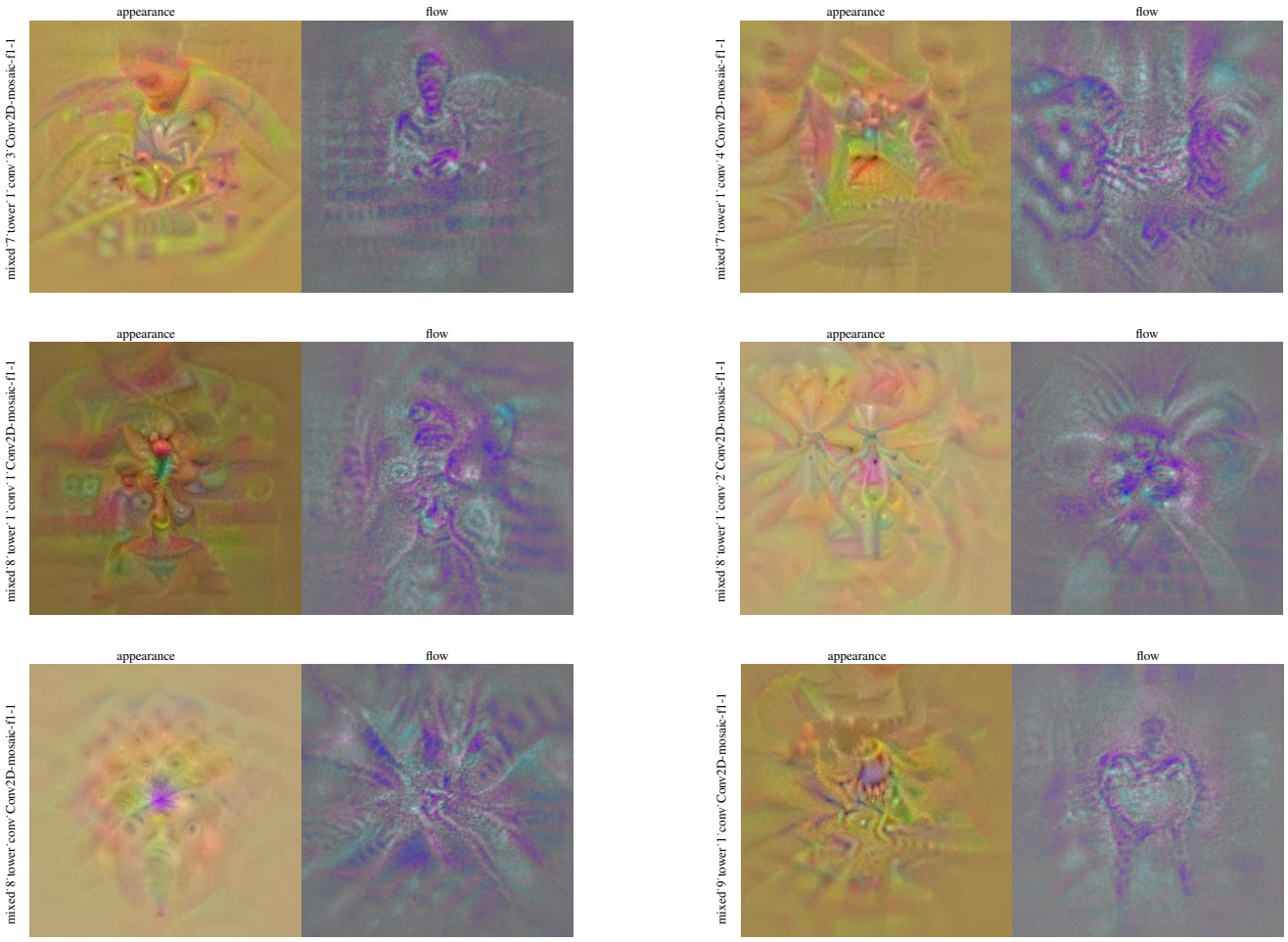


Figure B.41. Visualizations of convolutional layers of the Inception_v3 Two-Stream network [11] trained on Kinetics [1]. The number of filters shown at each layer is inversely proportional to the receptive field of that layer.

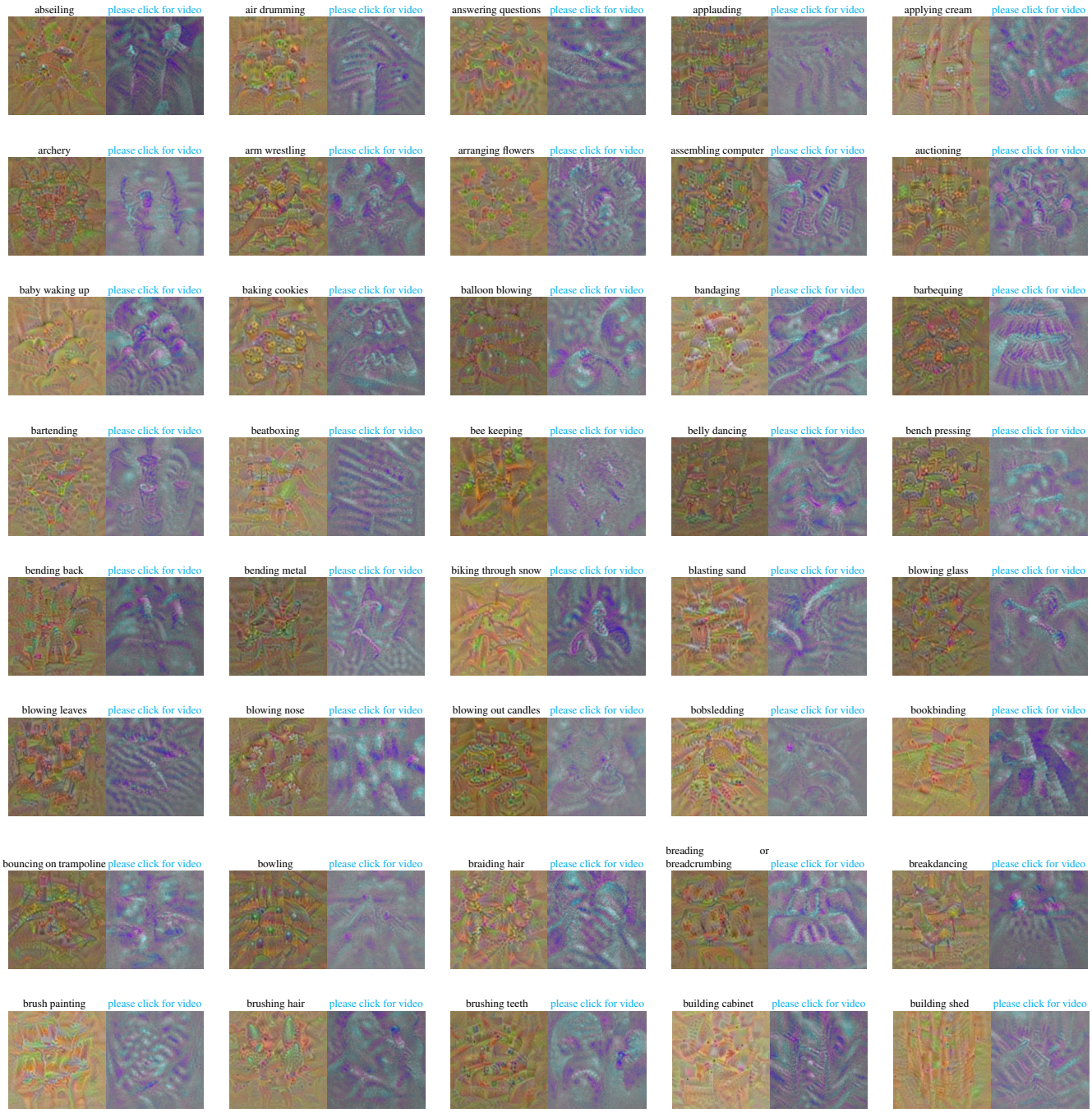


Figure B.42. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

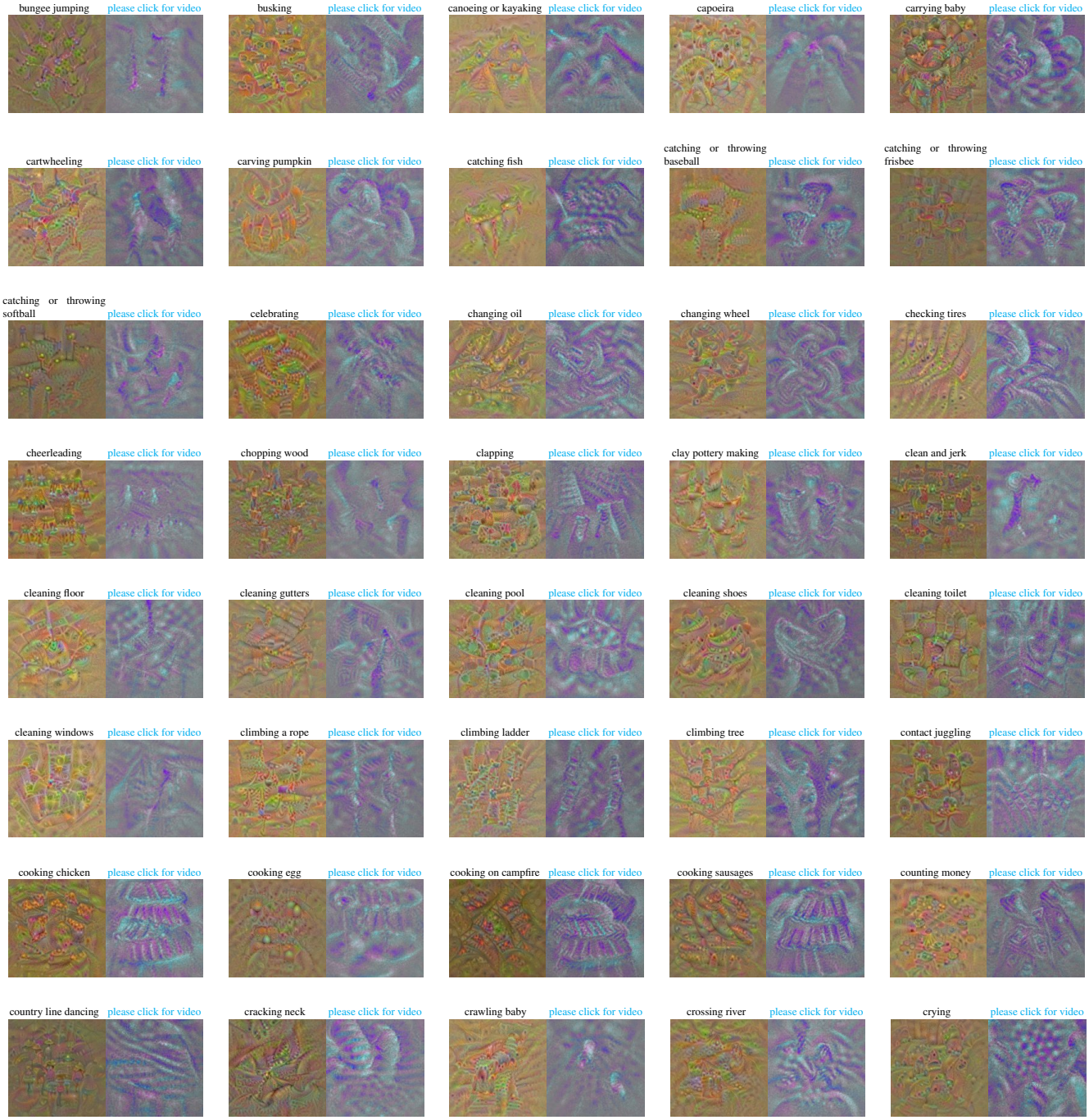


Figure B.43. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

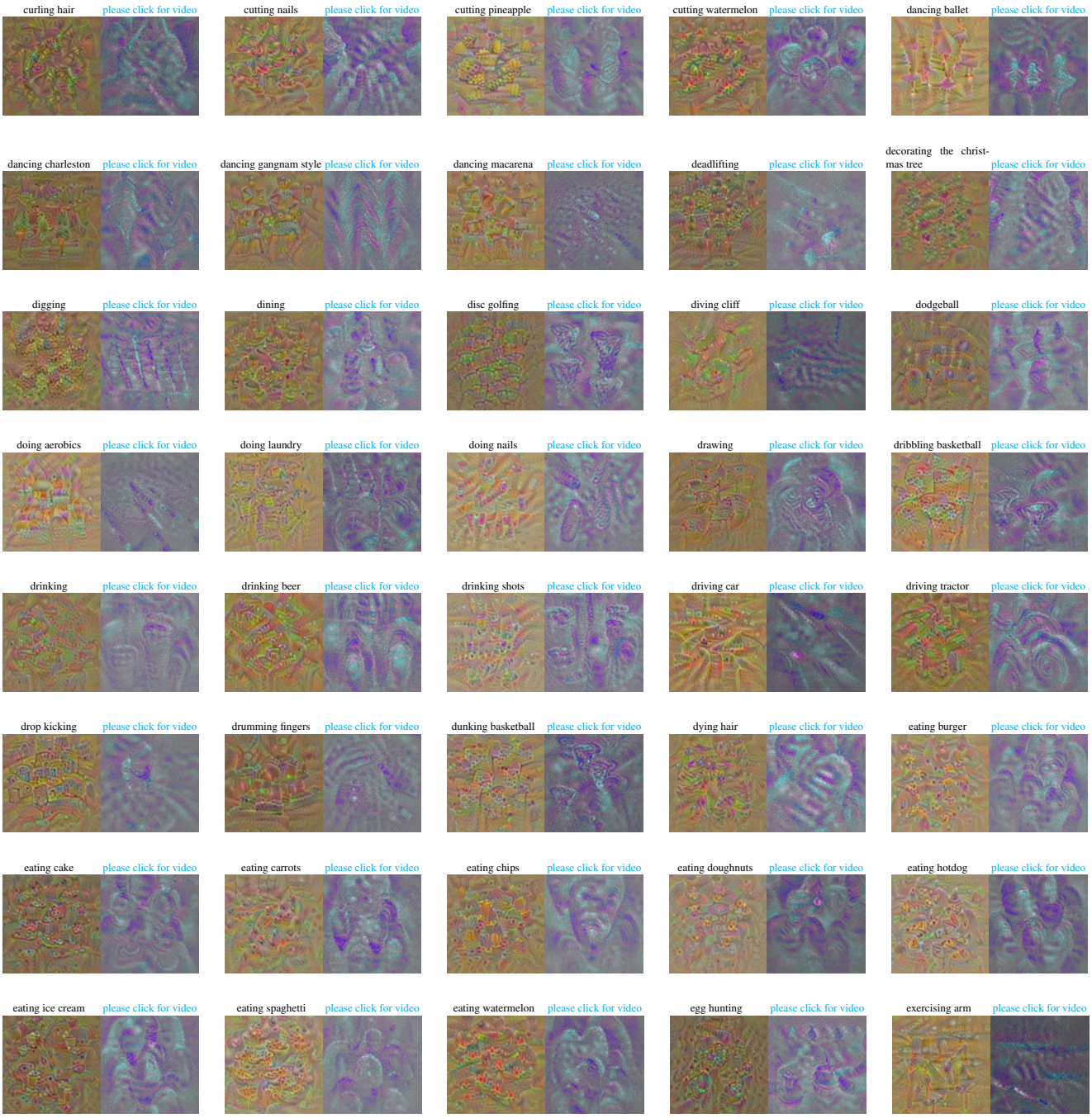


Figure B.44. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

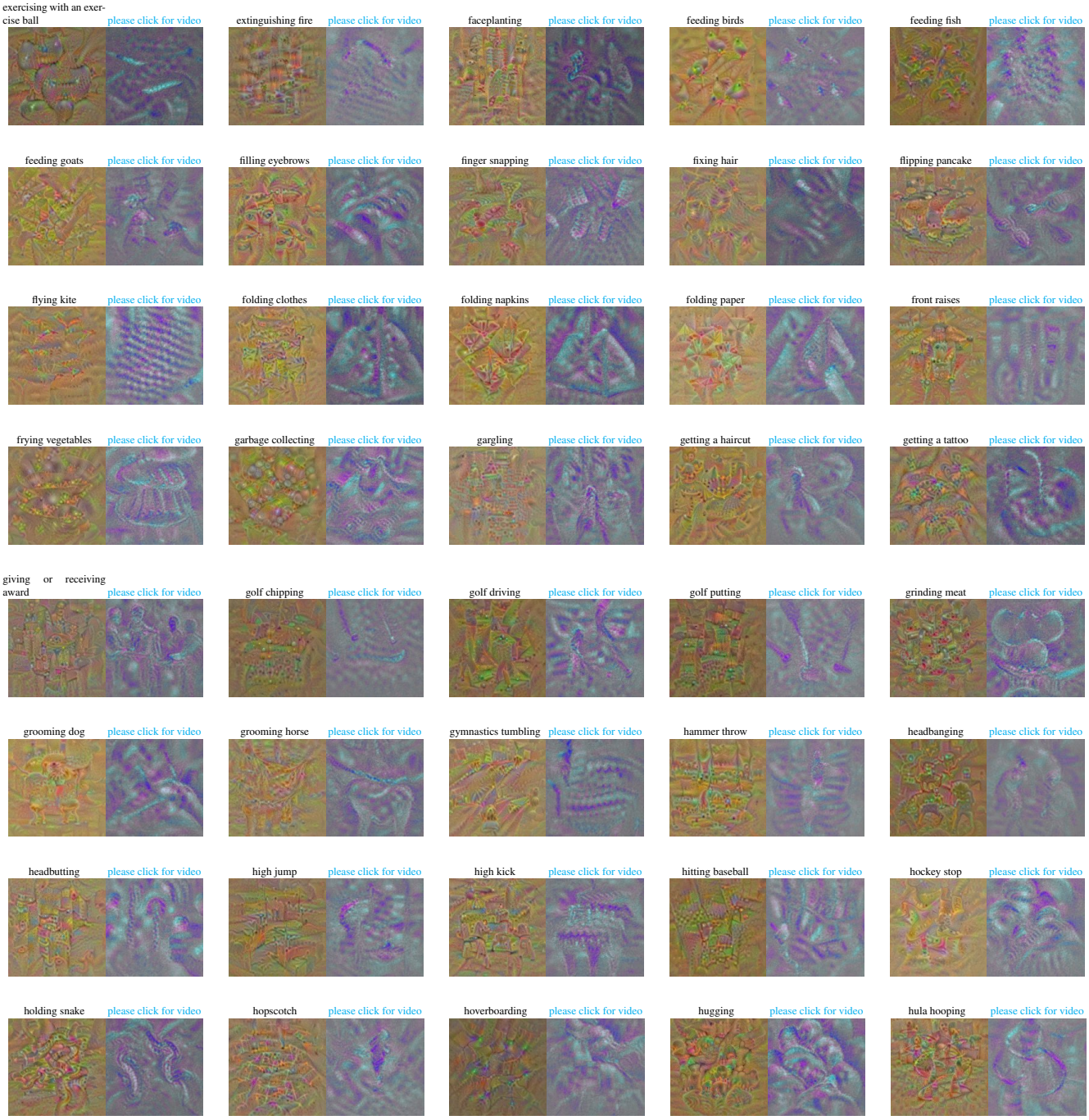


Figure B.45. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

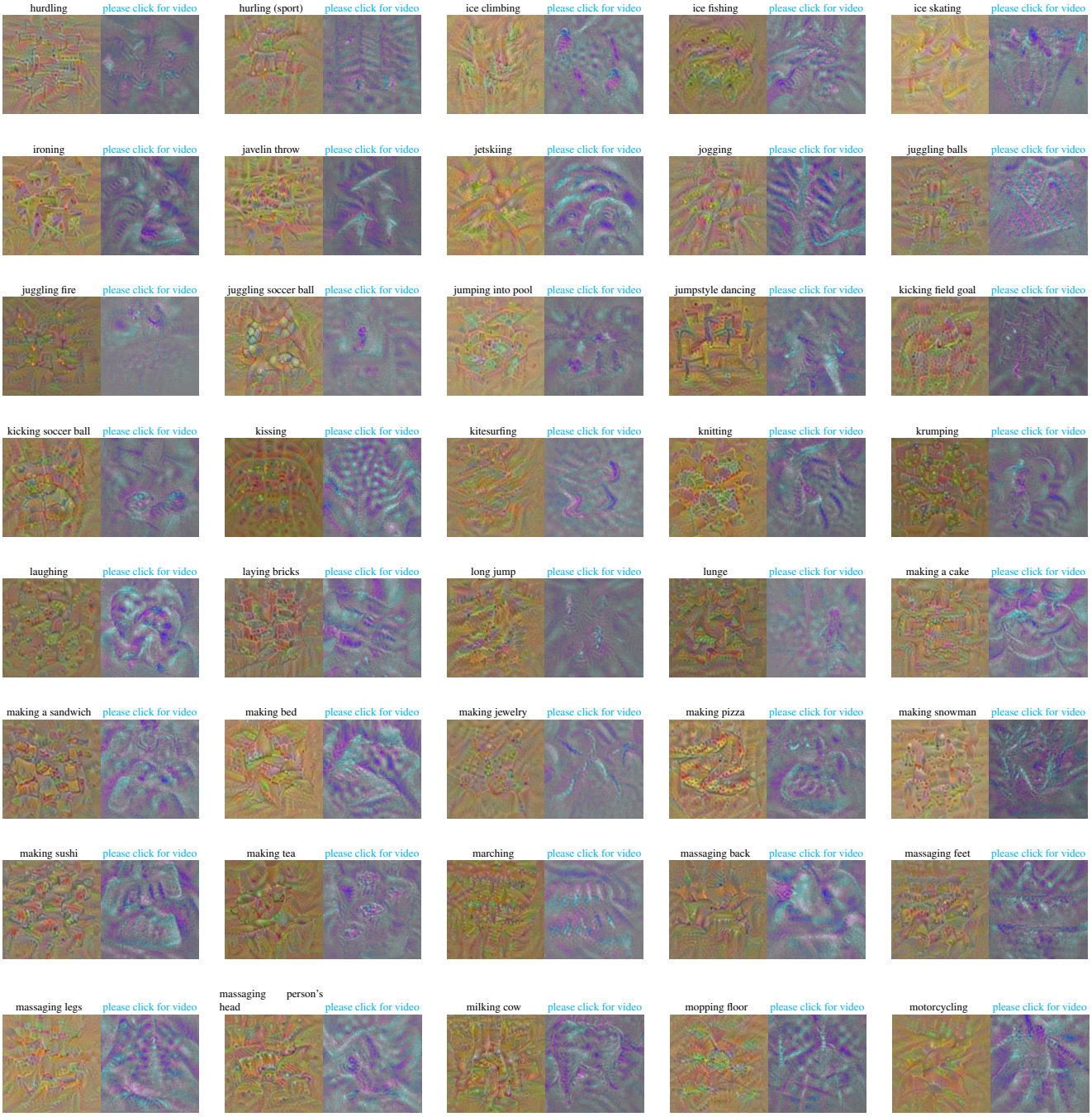


Figure B.46. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

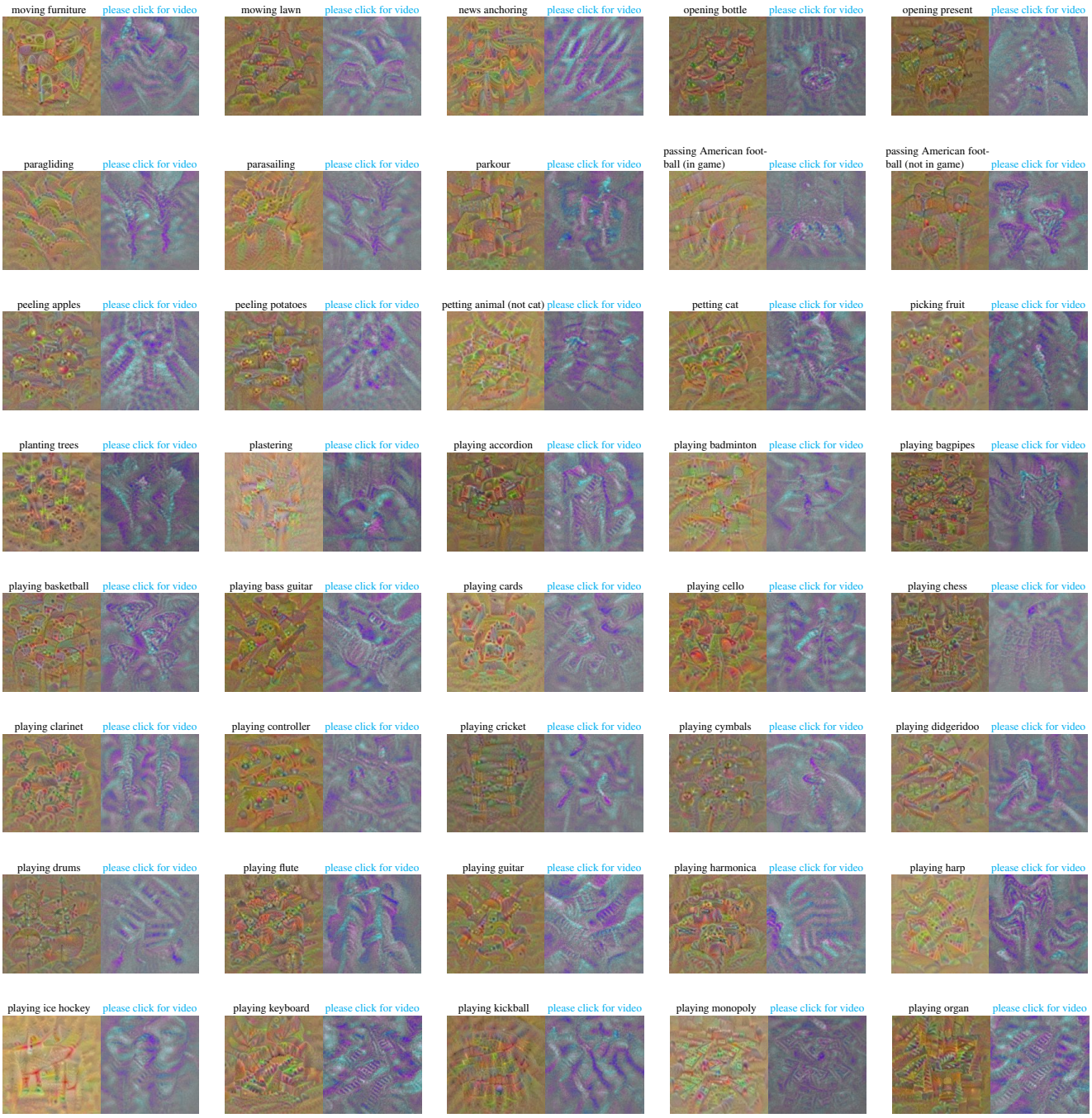


Figure B.47. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

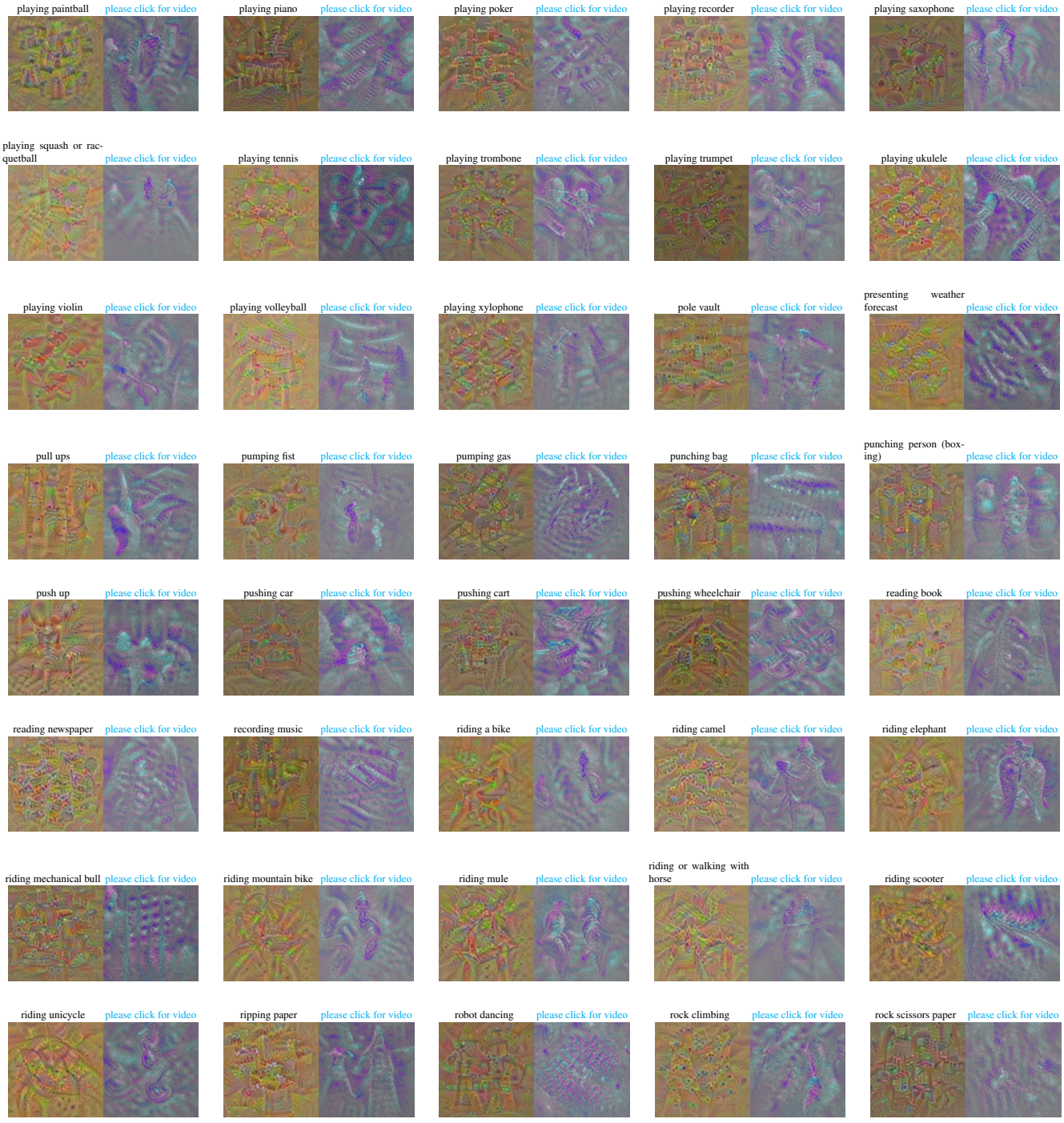


Figure B.48. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

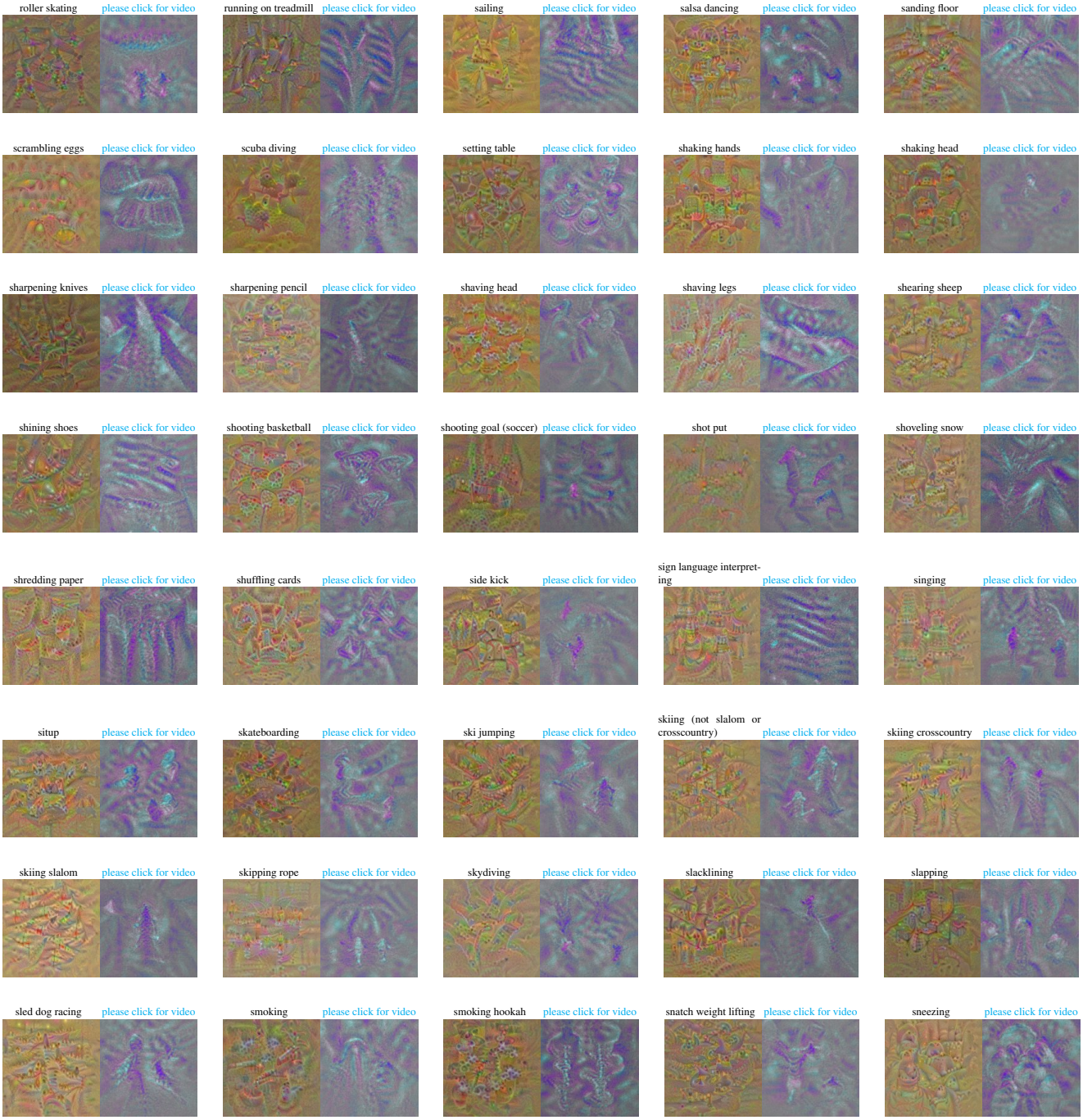


Figure B.49. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

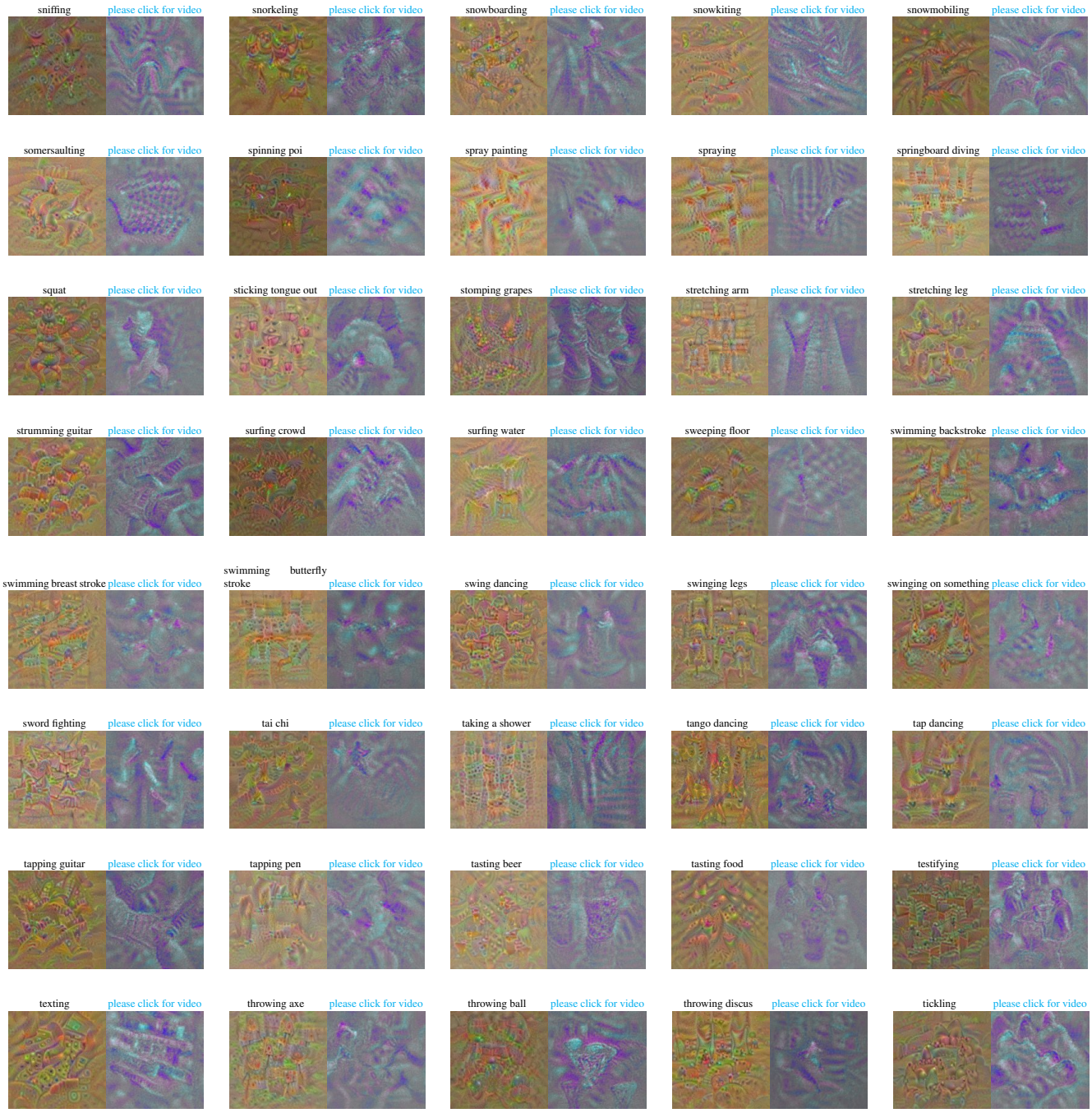


Figure B.50. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

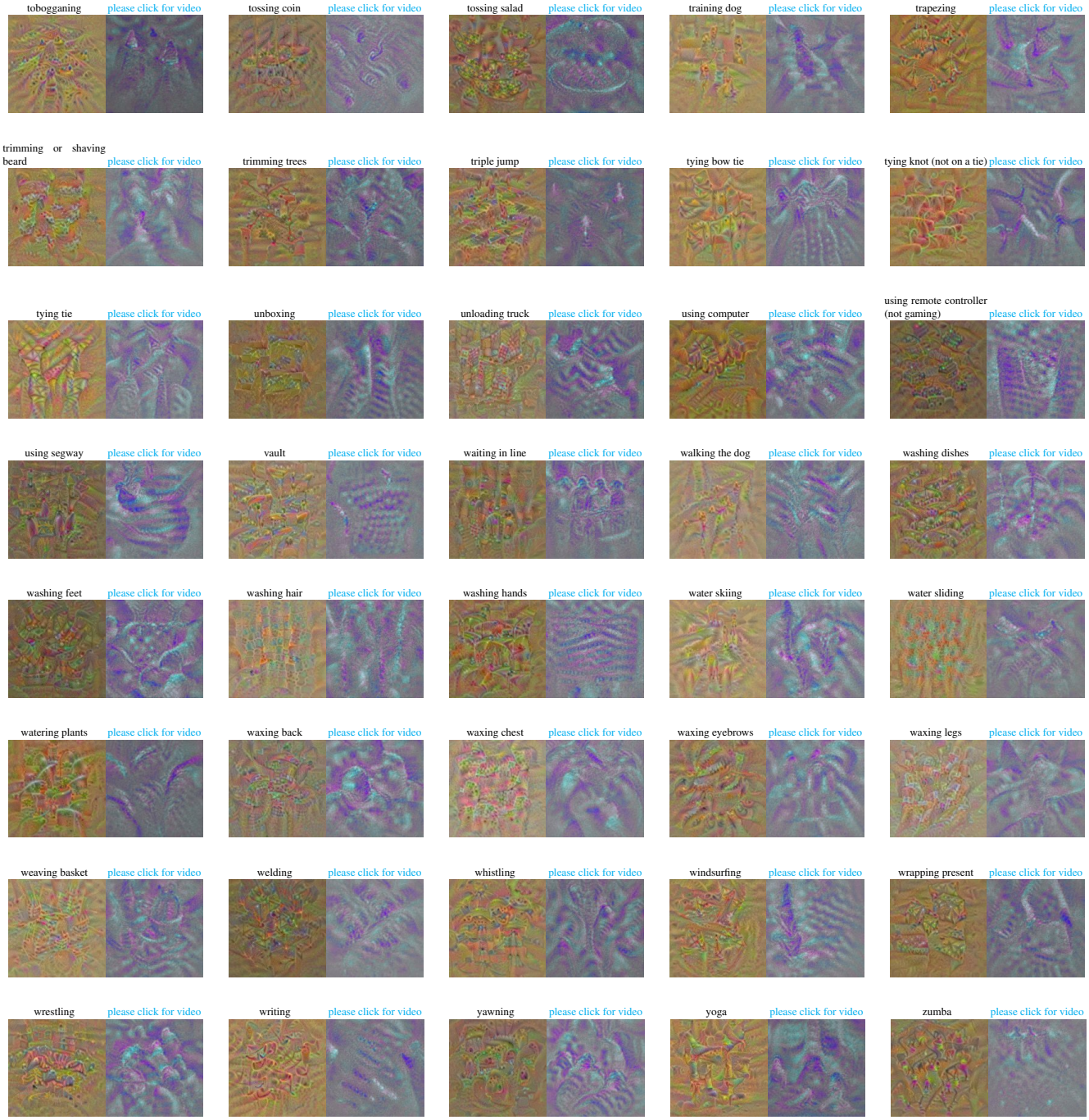


Figure B.51. Visualizations of classification units at the last layer of the Inception_v3 two-stream network [11] trained on Kinetics [1]. We show pairs of appearance and motion input generated by maximizing the prediction layer output for the respective classes listed above. Please click on the optical flow component for video playback.

References

[1] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proc. CVPR*, 2017.

[2] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016.

[3] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proc. CVPR*, 2016.

[4] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

[6] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. ICML*, 2015.

[7] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: a large video database for human motion recognition. In *Proc. ICCV*, 2011.

[8] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[9] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions calsses from videos in the wild. Technical Report CRCV-TR-12-01, 2012.

[10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, 2015.

[11] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv:1512.00567*, 2015.

[12] H. Wang and C. Schmid. Action recognition with improved trajectories. In *Proc. ICCV*, 2013.

[13] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016.

[14] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. In *Proc. DAGM*, pages 214–223, 2007.