SUPPLEMENTARY MATERIAL
Rethinking Zero-shot Video Classification: End-to-end Training for Realistic Applications

Network	Train	classes	UCF 50	UCF 101	HMDB 25	HMDB 51
C3D	K400	361	33.7	25.7	17.0	13.3
R3D_18	K400	361	37.2	29.0	20.4	16.8
R(2+1)D_18	K400	361	38.7	30.6	22.0	18.1
C3D	K700	664	40.3	33.1	22	17.0
R3D_18	K700	664	41.2	34.2	23.6	19.0
R(2+1)D_18	K700	664	**43.0**	**35.0**	**25.8**	**20.6**
R(2+1)D_18	K400	400	50.1	44.5	27.2	22.5
R(2+1)D_18	K700	700	54.6	49.7	30.5	25.6

Table 1: Accuracy of different backbone architectures trained on the first (K400) and last (K700) version of Kinetics [19]. The models are evaluated on a single clip (16 frames).

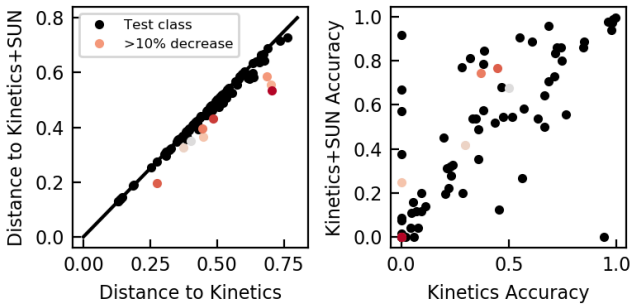

Figure 1: Each dot represents a UCF101 test class. Test class accuracy (right) and distance (left) to the train set (Kinetics664) for two models: one with random initialization and one pretrained on SUN (see Sec 3.4, Main Text). A colored dot indicates a test class that reduces its distance to the train set by more than 10% when SUN is included on training.

1. Backbone choice

Supplementary Table 1 compares the accuracy of three 3D convolutional backbones on two kinetics versions using our Training Protocol 1 (Sec. 4.2, Main Text). For this comparison we also tried using the full Kinetics 400/700 datasets, without removing overlapping test classes. The table shows that adding the 6% of the training classes most overlapping with the test set yields an unexpected >40% accuracy boost for UCF and 25% on HMDB. This proves that the zero-shot learning constraint is non-trivial.

2. SUN pretraining: easier task or better representation?

Section 3.4 (Main Text) shows that pretraining on a scenes dataset (SUN397) improves ZSL performance. In this section, we ask whether the boost is due to better model generalization or simply because the source domain becomes closer to the target domain.

Per each UCF101 test class, Sup. Fig. 1 shows the W2V distance to Kinetics train classes as well as (Kinetics + SUN) train classes. Test classes that got more than 10% closer to training data are marked in color. The right subplot, however, shows that the model trained on (Kinetics + SUN) boosts the accuracy of many classes – in particular, the accuracy of many classes that are *not* among the colored ones rose significantly. The model pretrained on SUN data increases performance on many classes which are not close to SUN data. We conclude that pretraining on SUN allows the model to generalize better over almost all test classes, not only the ones close to SUN data.

3. Training class diversity

We expand the analysis of Sec. 5.2 and Fig. 5, Main Text, by testing the influence of training class diversity on both UCF and HMDB. Sup. Fig. 2 correlates model performance with training class density. For this experiment, we selected 50 train classes with different density in the Word2Vec space, using the same clustering approach we used in Sec. 5.2. Per each diversity value, we select 50 classes and train a model multiple times to compute the standard deviation. Sup. Fig. 2 shows that test error de-

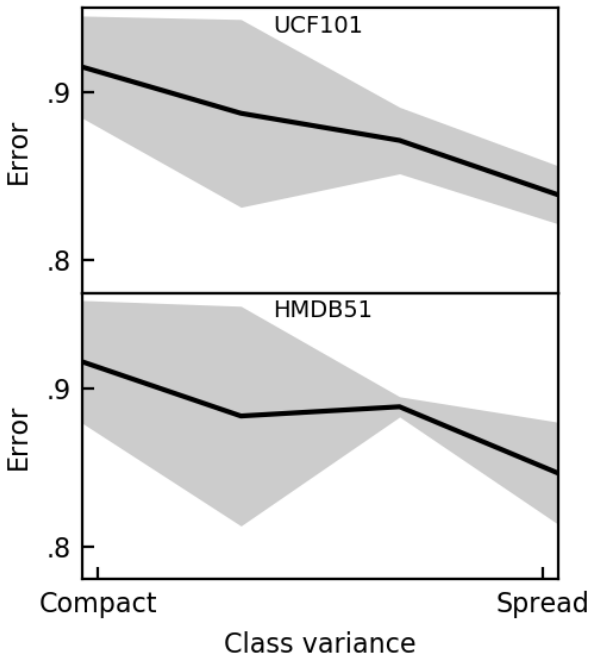

Figure 2: Performance of the e2e model trained on 50 Kinetic664 classes and tested on UCF and HMDB. The 50 classes are chosen based on diversity in their W2V embedding (see Fig. 5, Main Text, for details). The more semantically diverse the training classes, the lower the error.

creases as training classes become more diverse. At the same time, the standard deviation decreases, indicating that for compact classes, the performance highly depends on where in the class space we sample the classes, which is something we only know once the test set is available.

This outcome is not obvious, since we might expect the task to become harder when class variance increases (given the same number of training datapoints). However, we do not observe decrease in performance. Therefore, we can conclude that the model can only benefit from a high variety within the train class distribution. This new insight can be useful during training dataset collection.

4. Analyze the model capability action per action

What does better or worse accuracy indicate for specific classes? We break down the change in performance between models for each UCF101 test class.

4.1. Direct comparison by sorting classes

In Sec. 5, Main Text, we evaluated the model using error aggregated over all the test classes. It is also interesting to know whether the network is getting better at recognizing specific classes, or improves across the board?

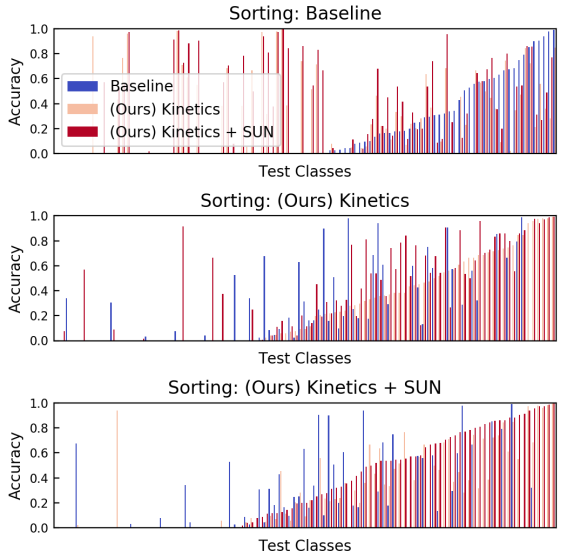

Figure 3: Accuracy on each UCF101 test class, for three models. Each subplot uses different model's accuracies to sort the classes, otherwise the numbers are the same.

Sup. Figure 3 shows the accuracy on each UCF test class for three models: baseline, e2e trained on Kinetics, and e2e pretrained on SUN397 and then trained on Kinetics. We sorted the classes from hardest to easiest for each model. Sup. Fig. 4 shows the same information, zoomed in on worst and best actions only. The two plots show that some of the actions which are difficult for the baseline model are correctly classified by our e2e models. On the other hand, the inverse situation is rare. In addition, the actions which are correctly classified by the baseline are also easily identified by our models.

In addition, the results of e2e trained on Kinetics and e2e pretrained on SUN and trained on Kinetics are highly correlated, but the second achieves overall better performances. This suggests that SUN provides *complementary* information to Kinetics which are useful for the target task. On the other hand, the baseline is less correlated with the e2e results, suggesting that the fixed visual features have a lot to learn and *should be fine-tuned*.

4.2. Confusion matrices

Sup. Figures 6 - 8 show confusion matrices of the three models we evaluated on UCF101. Sup. Fig. 5 shows the three CM directly compared with each other. In particular, we show the L2 distance computed pair-wise between

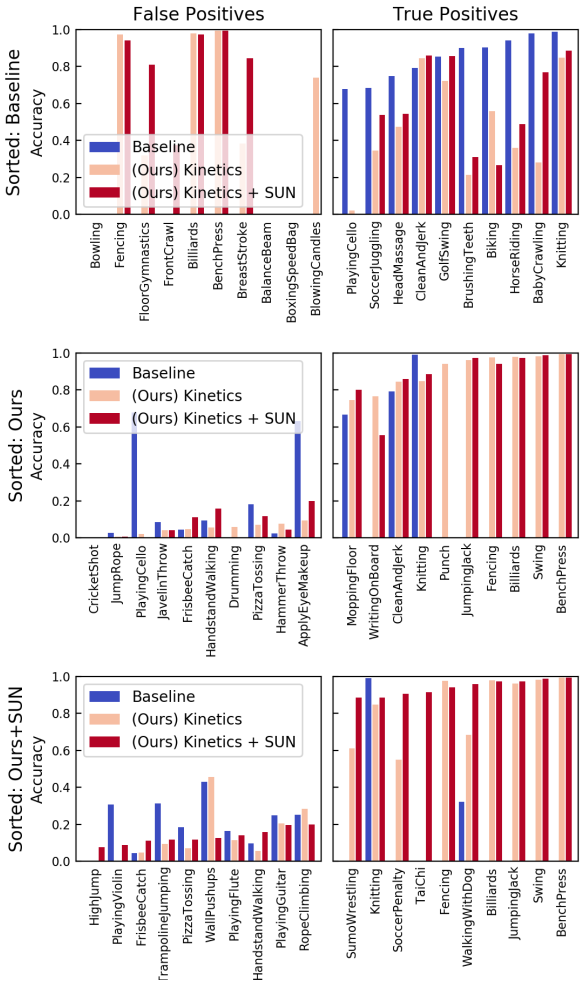

Figure 4: Accuracy on best and worst 10 classes for each model.

the CMs. This shows biases present in the Baseline model, which were removed by e2e training. Some interesting biases we discovered:

Playing: All models confuse the classes starting with the word "Playing". This issue probably comes from the way we embed the class name into the semantic embedding – simply averaging the words. Future work might focus on tackling this problem by using a different semantic encoder. This bias is less pronounced in e2e models.

JumpingRope: The baseline model wrongly classifies many actions as *JumpingRope*.

HandStandWalking: Our model trained on only Kinetics has a bias towards *HandStandWalking* class. This is attenuated by pre-training on SUN.

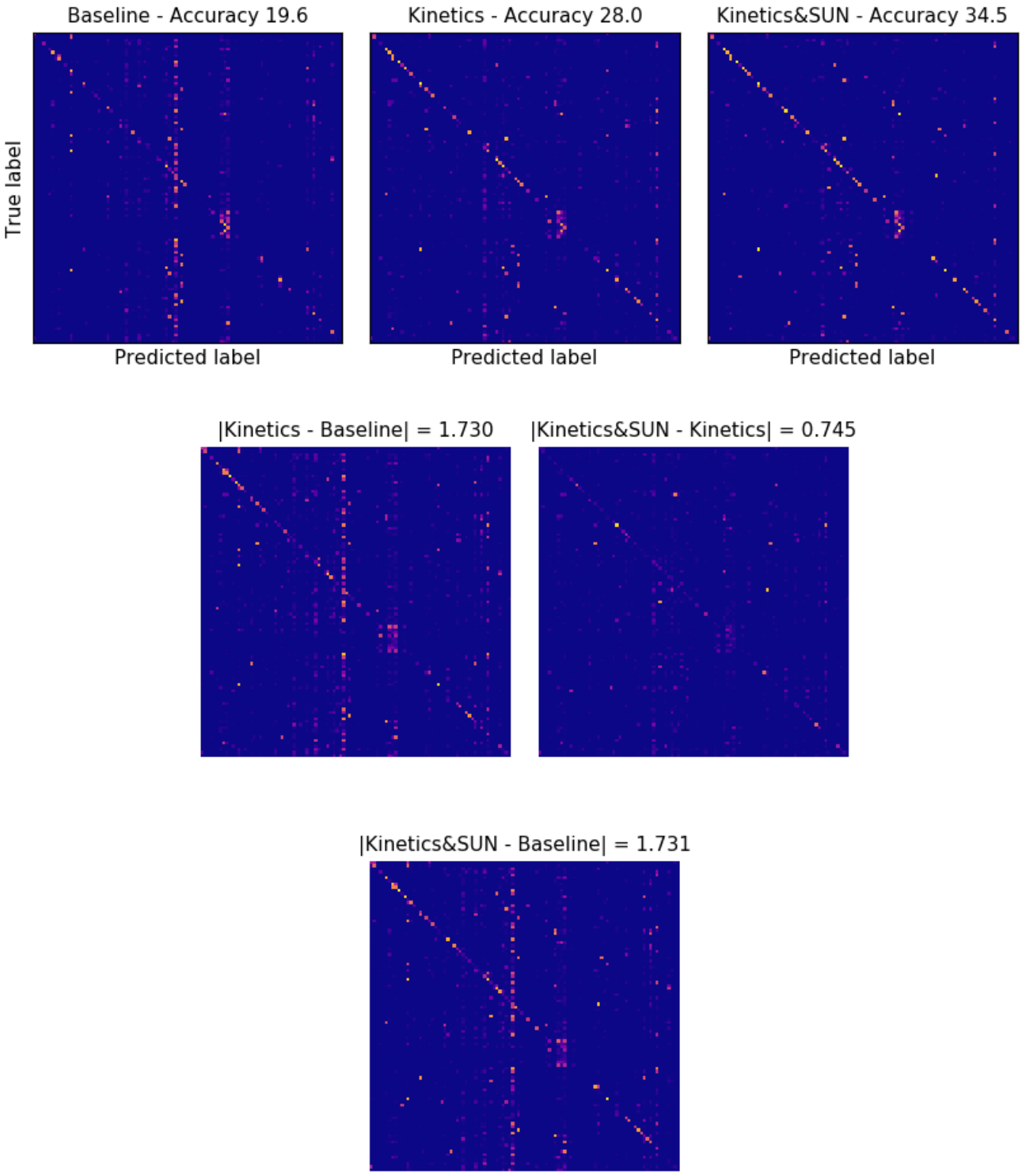

Figure 5: **Top**: UCF101 confusion matrices. **Middle and Bottom**: Pairwise L2 distances between the CMs, with average score indicated in the title.

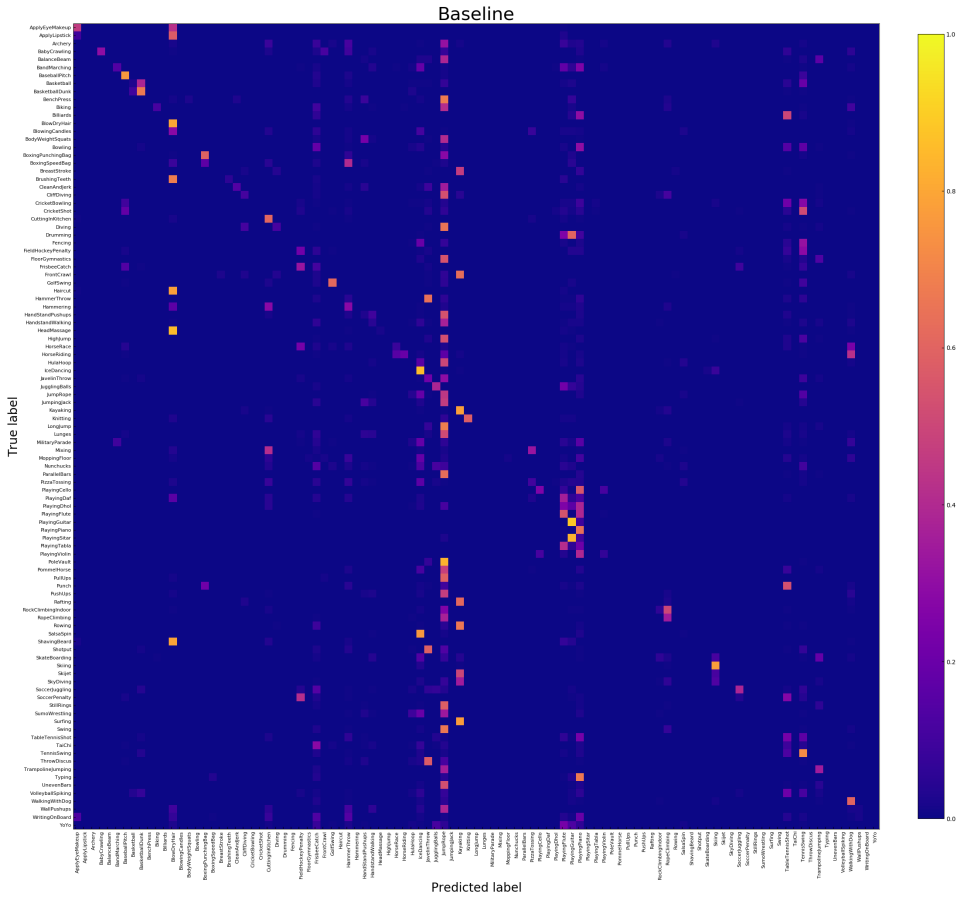

Figure 6: Confusion matrix on UCF101 using our baseline model (see Sec. 3.2 in the main paper). (Figure better seen zoomed in on the digital version)

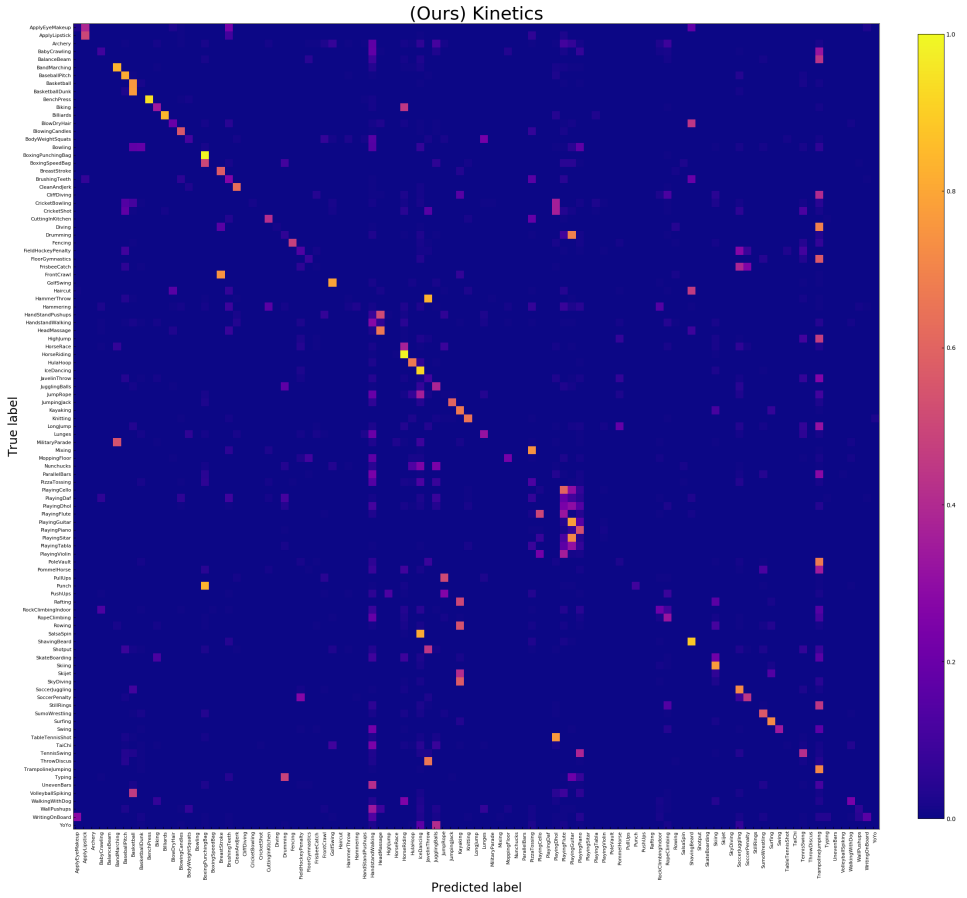

Figure 7: Confusion matrix on UCF101 using our e2e model trained on Kinetics. (Figure better seen zoomed in on the digital version)

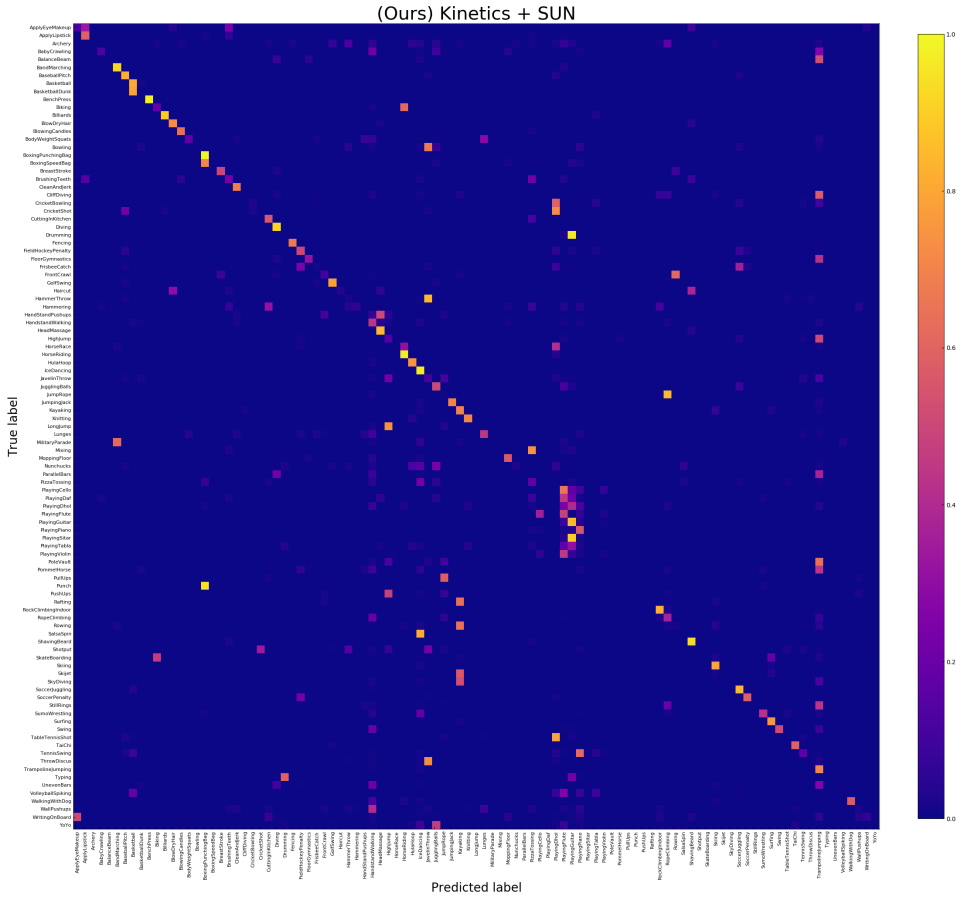

Figure 8: Confusion matrix on UCF101 using our e2e model pretrained on SUN397 (see Sec. 3.4 in the main paper) and fine-tuned on Kinetics. (Figure better seen zoomed in on the digital version)