Cooking Activity Detection

This exercise looks to detect cooking activities from a head-mounted camera providing a first-person (or egocentric) perspective of working in a kitchen.

Data collection

Due to time limitations, a small dataset was put together from cooking videos obtained from YouTube. While in [1] the labelling is done with a combination of narration, transcription and parsing, for simplicity, the video was cut into distinct segments associated with specific tasks. These segments are saved as individual videos and named according to their task, allowing the class to be inferred from the filenames when loading the data. Given the small amount of data and simplicity of the dataset, the data is split into just four classes: heating, chopping, stirring and an ‘unknown’ class. These were selected as they were the only activities that occurred consistently in the data, and the ‘unknown’ class allowed for inclusion of miscellaneous activities.

Transfer learning

The code allows transfer learning from any of the four networks used in [1], using the embedded features to train a simple LSTM model for activity classification. In the work here, the Temporal Relation Networks (TRN) model [2], which itself is an extension of the Temporal Segment Networks model [3], is used. The TRN model has the advantage of using a the Temporal Relation pooling strategy, which better encodes temporal dependency in the data, thus producing better performance for activity recognition tasks [2]. The

The model

TRN processes input sequences to produce sequences of deep encodings. These deep encodings are then aggregated to produce a consensus-based prediction. In this case, the consensus-based processing is replaced by an LSTM trained on the transfer-learned features. The choice of an LSTM is twofold:

1. While the transfer learning provides significant compression of the 224x224=50176 input dimensions – compressing these to 2048 dimensions – the dimensionality is still very high for the small amount of training data (N=390). This is compounded by the fact that the embedding comprises sequences of 8 segments in length. Using an LSTM allows the dimensionality to remain at 2048, rather than a concatenation-based approach which would result in a vastly disproportionate 2048x8=16384 dimensions.
2. Given that activity detection is highly reliant on temporal context [2, 4], it is beneficial to process sequences effectively; LSTMs are specifically designed to deal with the encoding sequential information through their various gating mechanisms [5].

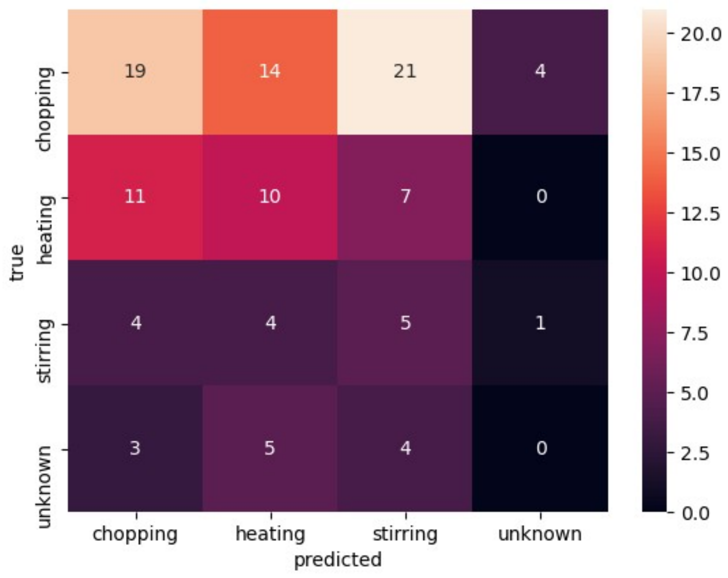
Due to the small amount of data, the LSTM comprises a single layer of 100 neurons. This was chosen as it provides reasonable degree of variational freedom while not substantially increasing the model complexity (as doing so would further add to convergence issues given the small amount of data/comparatively high dimensionality).

Initial experiments demonstrated that the LSTM struggled to converge. To improve performance, dimensionality reduction was incorporated in the form of PCA. Using PCA, the top 200 principal components are extracted from the deep embedding (capturing ~30% of the variance). Fewer than 200 features resulted in under-fitting due to loss of information, while the model struggled to converge with greater than 200 features due to high dimensionality and low data.

Results

As a simple illustration of the model's performance, a video, 'labeled.mp4', has been generated which labels each frame according to the predicted task. As can be seen on viewing, the performance is quite poor. The poor performance is likely down to several key factors:

1. With dimensionality D=200 (after dimensionality reduction) and training set size N=390, the network is likely to either vastly over-fit to the training data. This is confirmed by the below metrics.
2. While the LSTM is fairly small, having only a single hidden layer of 100 units, it's still a relatively complex model that likely requires significantly more data to train effectively.



Confusion matrix of activity classifications on test set

Dataset	**Accuracy**	**F1 score (weighted)**
Training	0.8597	0.8593
Test	0.3036	0.3136

Table of accuracy and F1 score results on training and test data

The confusion matrix generated from the test set demonstrates that the majority of misclassifications fall into the 'chopping' and 'heating' classes. Observing the data demonstrates that these classes are associated with very distinct frames: the 'chopping' always occurs around a chopping board in a better illuminated area and distinct beige/wood colour palette, whereas the 'heating' data comprises frames dominated by a black/metallic colour palette. It's likely that due to limitations in the data, these broad features override activity-specific features of the subject's movement, thus causing misclassifications due to failing to capture the distribution of the activities themselves. As a result of this, stirring is frequently misclassified – and as it tends to occur in the same area as ‘chopping’, this is the class that stirring tends to fall into. This is likely down to a combination of limited data and poor quality dimensionality reduction, as the principal components used only captures ~30% of the variance in the data.

Discussion

Transfer learning is typically a very powerful approach for harnessing the power of sophisticated, large-data network architectures in more limited-data settings. Nonetheless, the same statistical principles apply in this case as with any other ML application, and unfortunately the combination of high dimensionality and small data inhibited model convergence.

The fact that the top 300 principle components only described ~30% of the variance is an indicator that the embedded features are well de-correlated and don’t contain significant redundancy. Thus, if small data were a requirement of the transfer learning application, then transfer learning from a model with a smaller deep encoding would be beneficial. This would help by compressing the features into a lower dimensional representation, which in turn would facilitate training with a smaller dataset. One method for achieving this may be to adapt one of the TSN/TRN/TSM models to add an autoencoder with a sufficiently small bottleneck – this autoencoder could either be trained to autoencode the input features, or as a sub-network autoencoding the embedded features.

A bi-directional LSTM may well improve performance in the case of more training data, as these exhibit enhanced performance in sequential processing tasks [6], and have specifically been shown to improve performance in a human activity recognition context [4]. However, when a bi-directional LSTM was tried in this context, performance suffered significantly – likely due to the increase in model complexity in combination with limited data.

In a scenario in which more data could be collected, retraining of the segment consensus layer may be more effective than substituting this layer for an LSTM. Doing so may also help to preserve some of the interpretability benefits of the TRN. More time would also allow for better training protocols to be implemented, such as early stopping, and for hyper-parameters to be optimised, e.g. via Bayesian optimisation.

More time would have allowed for investigation of the performance of the TSM model, which may have provided more descriptive embeddings, as this is a further improvement on the TRN method.

References

[1] Damen, Dima, et al. "Rescaling egocentric vision." arXiv preprint arXiv:2006.13256 (2020).

[2] Zhou, Bolei, et al. "Temporal relational reasoning in videos." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[3] Wang, Limin, et al. "Temporal segment networks: Towards good practices for deep action recognition." European conference on computer vision. Springer, Cham, 2016.

[4] Alawneh, Luay, et al. "A Comparison of Unidirectional and Bidirectional LSTM Networks for Human Activity Recognition." 2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops). IEEE, 2020.

[5] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

[6] Graves, Alex, and Jürgen Schmidhuber. "Framewise phoneme classification with bidirectional LSTM and other neural network architectures." Neural networks 18.5-6 (2005): 602-610.