Exploring Pseudo-Labels for Domain Adaptation in Egocentric Video

By

RHODRI GEORGE LILES GUERRIER

School of Computer Science
UNIVERSITY OF BRISTOL

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of MASTERS OF SCIENCE BY RESEARCH in the Faculty of Engineering.

FRIDAY 15 SEPTEMBER, 2022

Word count: 20,915

Abstract

The task of unsupervised domain adaptation offers an effective way to assess the generalisability of models. It incorporates the problems of unlabelled data and domain shift, which are common in real-world settings due data collected from different people, locations and equipment. Pseudo-labelling approaches to this problem attempt to annotate the unlabelled data using the knowledge from another labelled distribution, so it can be used to train a model which adapts well across both distributions. These methodologies show great promise as both complimentary and stand-alone techniques. By exploring this approach in two different ways for egocentric data, this work shows that noise reduction of pseudo-labels does not always lead to better adaptation and the methodology as a whole can struggle on larger and more complex datasets, with numerous classes.

A popular training regime which randomly samples clips from videos is confirmed in this work to cause variations in the feature representations and class predictions across the same video. More sophisticated pseudo-labelling methods are then proposed to avoid harmful segments of videos and to exploit more discriminative ones. Although some of these methods reduce pseudo-label noise much better than random sampling, they cause a degradation of the model during adaptation. Heavier focus on particular video regions causes a bias towards the largest class and reduces the diversity of samples seen by the model.

The efficacy of pseudo-labelling is further analysed in comparison to a more complex dataset. Some success is possible, but adaptation difficulty shows that the pseudo-label class distribution matters more when the dataset has a large set of classes. The work shows that when the pseudo-label class distribution does not match that of the test class distribution very well, the methodology can struggle considerably. This points to a difficult problem for pseudo-labelling strategies on the unsupervised domain adaptation task.

Dedication and acknowledgements

Thank you to my supervisor, Professor Dima Damen, as well as both Toby Perrett and Chiara Plizzari. At the beginning of the course I knew absolutely nothing about machine learning and research. Without their help, advice and direction, I would probably still be in a similar situation.

I would also like to make a special thanks to my girlfriend, Amber Ainscough, who put up with my constant catastrophising and late night frustration at GPU memory errors, whilst always supporting me.

Author's declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's Regulations and Code of Practice for Research Degree Programmes and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: .....RHODRI GUERRIER..... DATE: .....15/09/23.....

Table of Contents

List of Tables

List of Figures

Figure **Page**

CHAPTER 1

INTRODUCTION

Although human beings are extremely good at learning something new and then generalising it to different scenarios or tasks, machine learning models actually struggle with this quite a lot. One reason for this is likely to do with how each obtain knowledge. Whilst humans learn by interacting with the world over many years and encounter a huge variety of different scenarios, objects and people, machine learning models learn on narrow and specialised datasets. The latter regime is capable of producing extremely powerful models. However, it has also been known for a long time that they can become heavily biased on the datasets which they are trained upon (e.g. [51]). This means that when they encounter data from a different dataset (or more commonly referred to as a different distribution), they can perform much worse than expected. A lot of early machine learning research did not often refer to this phenomena though, due to the conventional method of assessing a model's performance. This involves simply splitting a dataset into training and testing splits, training the model with the former and then seeing how many samples from the latter are predicted correctly. However, with the rise of larger deep learning models [2, 12, 13] and more available data of differing distributions [4, 9, 10, 29, 44], this area of generalisability has become more important than ever.

This work then focuses solely on this ability to generalise by investigating the task of unsupervised domain adaptation. This is a framework which assesses how well a model can generalise across different distributions of data for the same task, when labelled data is not available for some portion of the data [61]. Specifically, this work focuses on video data rather than images. The problem does exist for images too, but video is considered a much more difficult problem due to that fact that whilst images vary in appearance mostly, videos tend to vary both in appearance and through time [4]. Furthermore, the work focuses on one category of solutions for this problem called pseudo-labelling, which has shown promise across a number of recent works

[11, 33, 61]. Both the capability to produce these pseudo-labels and utilise them for a method to bridge distribution differences are discussed in this work, as well as how they compare with different solutions in the field. This is measured in relation to two popular egocentric datasets [9, 10], as they offer a wide range of naturally captured human actions which are helpfully split to simulate the kind of distribution differences that cause degradation of conventional models.

1.1 Aims of Work

The aims of this work are as follows:

1. Investigate previous unsupervised domain adaptation approaches and understand the use of pseudo-labels.
2. Investigate how network predictions vary across different sections of variable length videos for a network trained on randomly sampled segments of videos.
3. Design and analyse new pseudo-labelling methods that exploit the full temporal range of video inputs on EPIC-Kithens-55.
4. Determine if pseudo-label noise is the primary factor for adaptation performance.
5. Determine how effective pseudo-labelling is on the larger, and more complex, EPIC-Kitchens-100 dataset by submitting to the annual UDA challenge hosted by the creators.

1.2 Structure of Thesis

The thesis starts with some background information in chapter 2. Video understanding architectures, motivation and definition for unsupervised domain adaptation and recent successful methods are all discussed in order to reflect on the state of the research field. Chapter 3 provides a discussion on the use of pseudo-labelling on the EPIC-Kitchens-55 dataset. The dataset is first described and a motivation for pseudo-labelling is given by exploring the variations in predictions across sections of video. The rest of the chapter explores new pseudo-labelling methods which utilise these variations, how they compare with the approach from [11] and whether increasing initial pseudo-label accuracy translates to an increase in adaptation. The effectiveness of pseudo-labelling is further analysed on a larger egocentric dataset, EPIC-Kitchens-100, in chapter 4, as part of the unsupervised domain adaptation challenge. The transformer architecture is coupled with pre-extracted features to determine how well two pseudo-labelling methods can adapt to the expanded dataset. A range of ablation studies are also reported on for source-only performance, ensembling and the effect of the pseudo-label class distribution on adaptation performance. Lastly, chapter 5 provides a conclusion to the work described in this thesis.

CHAPTER 2

TECHNICAL BACKGROUND

This chapter describes some of the relevant background literature to give a wider context for the research field related to the overall work. To begin with, some of the most common input data modalities are described as most modern video classification networks rely on multi-modal inputs to improve performance. A few of the most used video understanding network architectures are then described, as these are often used as backbones or to extract features for unsupervised domain adaptation methods. Next, a motivation and definition for unsupervised domain adaptation is given, along with three broad categories of common solutions used to tackle it. Lastly, some of the most recent and successful unsupervised video domain adaptation methods are explained.

2.1 Modalities

The following sections showcase how different modalities can complement one another in video classification and boost performance. However, it is important to first cover the most common modalities used as they are not always intuitively understood.

Firstly, the raw RGB frames are often the first choice for input into video models as they are relatively understandable and widely recognised. The input consists of a 4-dimensional tensor with shape $TxHxWx3$ where at each time step T there is an image of height H, width W and 3 channels for the red, green and blue colour scales.

Secondly, the RGB frames are often used to derive new inputs, called Optical Flow frames, which capture the direction and magnitude of motions within the video. This achieved by first measuring the displacement of each pixel between pairs of frames and then mapping the resultant horizontal and vertical components of the displacement vectors to separate frames. The most common approach for calculating this is described in [46] whereby the two directional frames

are used as separate channels in the input to the model, much like how RGB frames use three channels. The magnitude of the movement is then often translated into a grayscale value on the resulting flow frames so that the model receives both information on direction and velocity of motion.

Lastly, audio information is sometimes included in video classification models in order to combat the difficult of often occlusions and distortions in visual inputs. The most common approach to include this information is to simply convert the raw audio into a spectrogram so it can be used as a visual input to common architectures.

2.2 Video Understanding Architectures

2.2.1 I3D

The *Two-Stream Inflated 3D ConvNet* (or *I3D*) architecture is introduced by [2]. At the time, the release of the much larger and diverse Kinetics dataset [21] allowed the authors to identify two key concepts for building more successful architectures for modelling videos. Firstly, that deep image classification networks can be trivially expanded in order to produce spatio-temporal feature extractors. This involves taking an architecture, such as the Inception model from [19], and simply adding an extra dimension to the filter and pooling kernels. This means that established architectures can be converted for videos with ease. But more importantly the authors find that the pre-trained weights of the image-based architectures actually act as a valuable initialisation for the newly inflated networks. Meaning that the knowledge learned on large image datasets can still be helpful for implementing video-based models. Secondly, that although the RGB frames, coupled with the inflated image architectures, perform very well, the performance can still be greatly improved by including a second stream for optical flow frames. The results across datasets show a marked improvement when both modalities are utilised, over just one or the other.

The I3D network is utilised in a number of works discussed in section 2.4 [8, 11, 23, 30] and throughout the work in chapter 3.

2.2.2 Temporal Relation Network

The authors of [62] point out that due to the limited and dense sampling of frames required for CNN-based activity recognition networks, they are intrinsically biased towards capturing short-term dynamics. In order to account for more long-term aspects of videos, they propose the *Temporal Relational Network* (or *TRN*). This is based on the Relation Network [42] which models the relations between inputs by classifying on an aggregation of pair-wise relationships, produced through individual MLPs. The Temporal Relation Network expands this prior approach to the realm of multi-scale relations in order to understand the more complicated video inputs. Several Relation Networks are employed, only each one has a different number of frames inputted.

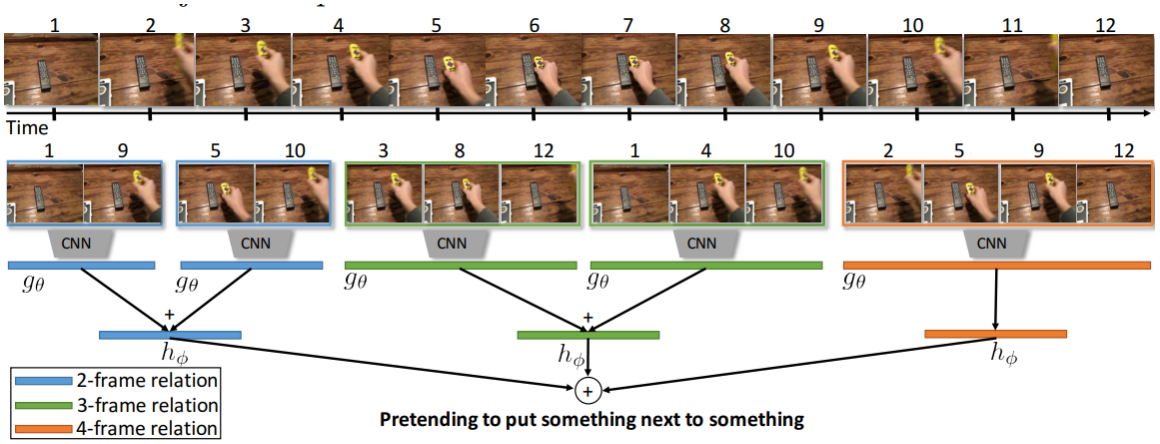


Figure 2.1: Example of how the Temporal Relation Network samples and ingest frames. Figure taken from [62].

For example, a standard Relation Network in one branch can model pair-wise relations, whilst another branch can extend the Relation Network to compute a relation between 3, 4 or more inputs. Each branch output, modelling the video in different time scales, can then be aggregated in order to produce a final representation of the video which captures both short and long term dynamics. An example illustration of the Temporal Relation Network can be seen in figure 2.1.

This network performs well across a number of datasets, but in particular outperforms the previous best result (which required both RGB and flow frames [43]) on the Charades dataset [45], by only sparsely sampling RGB frames.

The Temporal Relational Network plays an integral role in the construction of an early and popular unsupervised video domain adaptation method, called TA^3N [4]. This is discussed in section 2.4.

2.2.3 Temporal Binding Network

First introduced in [22], the *TBN* architecture is primarily designed to tackle the combination of multiple modalities for egocentric vision tasks. Egocentric video simply refers to recordings which attempt to mimic the human perspective. This kind of video makes up the entirety of the EPIC-Kitchens-55 [9] and EPIC-Kitchens-100 [10] datasets.

The authors of [22] make the observation that although networks which fuse multiple modalities already exist, such as the Temporal Segment Network [56], these often temporally aggregate each modality before combination, using late fusion. Furthermore, those works, such as [14], which do fuse modalities before temporal aggregation, do so for synchronised inputs. The authors propose that not only can fusion of modalities before temporal aggregation better exploit the available information, but that by using frames from different locations in each modality, the asynchronous audio-visual nature of videos can be better modelled. They achieve this by creating several multi-modality streams of BN-Inception models [19], each fed with frames randomly

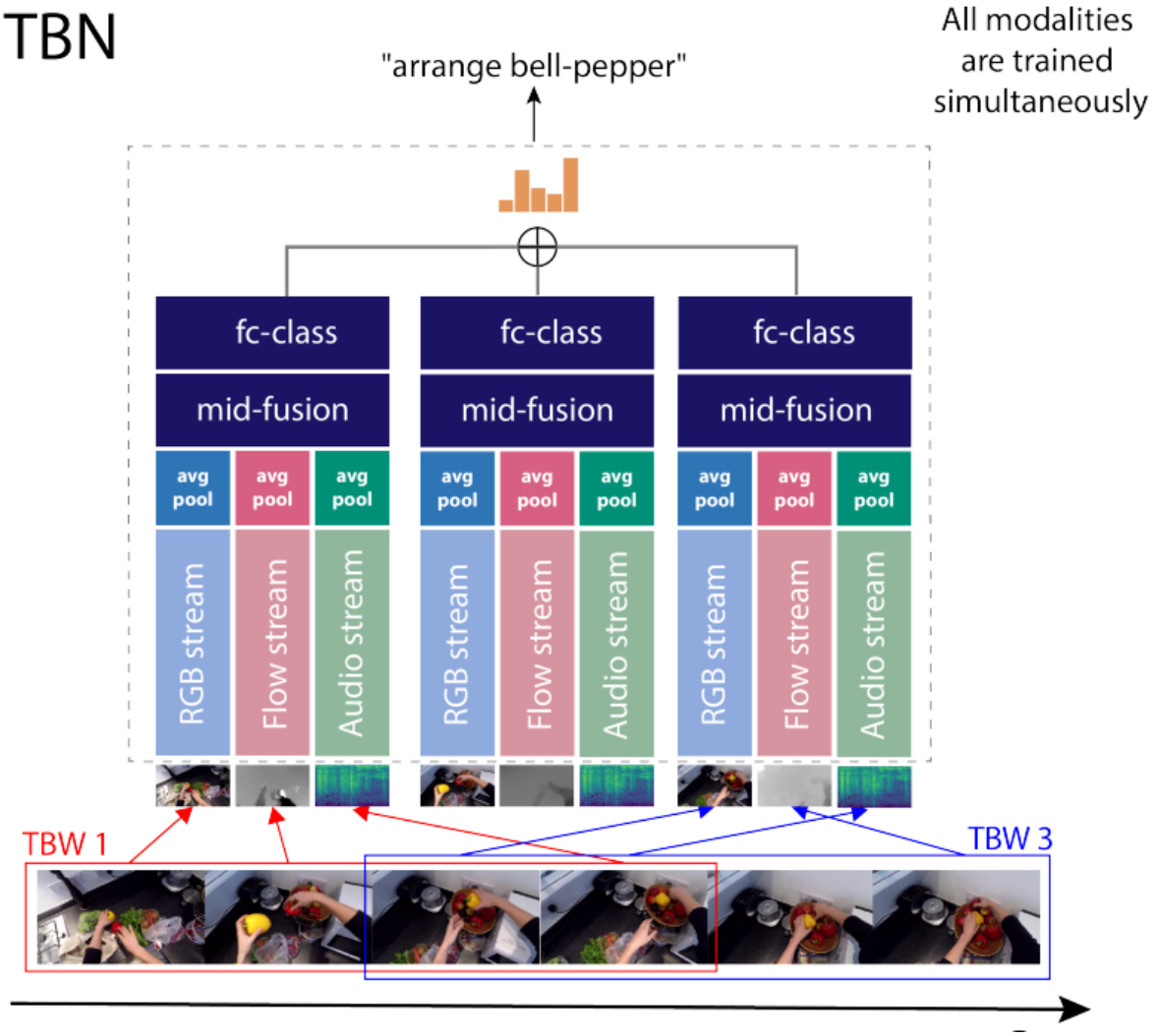
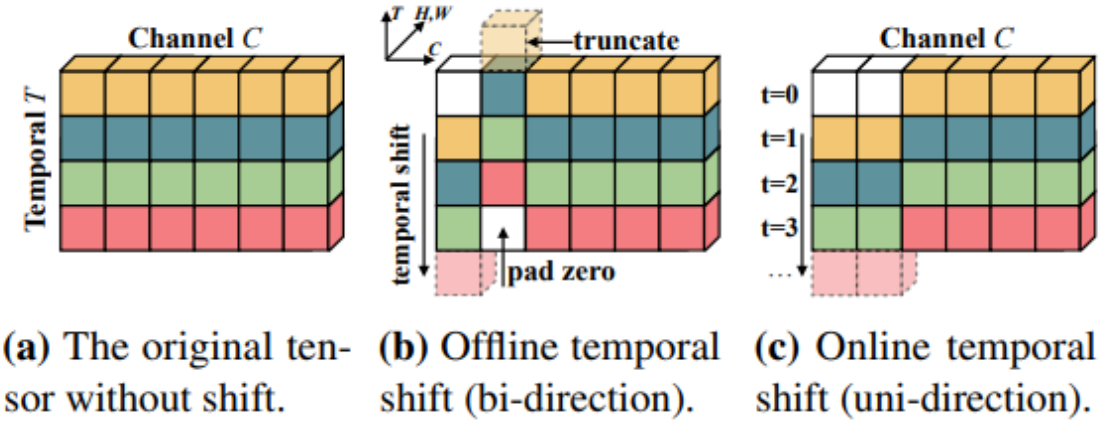


Figure 2.2: Temporal Binding Network architecture diagram. Figure taken from [22].

sampled from a different temporal window (referred to as a Temporal Binding Window, or TBW), as depicted in figure 2.3. Features from each modality-specific frame are then average pooled and fused before using a fully connected layer to produce one prediction per TBW. These predictions can then be summed to produce a final prediction.

Their results support the idea that mid-level fusion of modalities and asynchronous frame sampling are superior. Their architecture achieves a clear and marked improvement against prior works on the EPIC-Kitchens-55 dataset.

The Temporal Binding Network is used in order to extract the features provided by the coordinators of the EPIC-Kitchens-100 unsupervised domain adaptation challenge. These features are used, alongside features from another architecture (TSM [26]), for the submission detailed in chapter 4.



(a) The original tensor without shift. **(b)** Offline temporal shift (bi-direction). **(c)** Online temporal shift (uni-direction).

Figure 2.3: Temporal Shift Module methodology. Figure taken from [26].

2.2.4 Temporal Shift Module

Although 3D-CNNs are able to model temporal relations well and achieve good performance, they are much more computationally expensive when compared to 2D-CNNs, due to the extra dimension. The authors of [26] introduce the *Temporal Shift Module* (or *TSM*) as a way to achieve temporal modelling at a computational cost comparative to that of the 2D-CNN. This is achieved by simply shifting some of the channels forwards or backwards along the temporal dimension. Therefore, standard 2D convolutions can be used whilst also capturing relationships between frames from different points in time. However, they do not simply shift all of the channels otherwise this can cause a large memory overhead and damage the spatial reasoning of the model. Results show that only a small proportion of channels need to be shifted in order to improve accuracy without increasing memory usage too much. In order to stop any damage to the spatial reasoning of the model, the authors further place the Temporal Shift Modules on a residual branch which is then combined with the original input afterwards. The efficacy of both these approaches is confirmed in their results.

Their methodology, when compared against the TSN [56], increases performance across six difference datasets. For those datasets which rely more heavily on temporal understanding [17, 29], the performance increase against TSN [56] reaches double digits. Furthermore, the results show that similar or better performance is achievable without increasing the number of parameters across a number of different architectures [2, 56, 62].

The Temporal Shift Module is what is used to extract the second set of features used in the challenge submission, detailed in chapter 4.

2.3 Unsupervised Domain Adaptation

2.3.1 Motivation & Definition

Simply put, deep neural networks attempt to learn some function which can transform inputs into the desired format for the relevant task. For example, a regression task may transform a real number to a different real number, whilst classification tasks with C classes may transform a matrix of real numbers to a new C-dimensional vector of real numbers. The function which is learned however, is purely predicated on the distribution of the input data. For this reason, it is often assumed that training and testing data will be drawn from the same distribution. However, in real-world scenarios this is not always the case. Due to this formulation then, when unseen data is sampled from a distinctly different distribution to the one used in training, the model is much less capable at producing reliable results. An example of this can be seen in the "source only" results of [15]. The result shows that a model trained on one distribution ("source") is much less capable at identifying unseen data from a new distribution ("target") when compared with a model that has trained on data from the new ("target") distribution. This distribution difference is often described as *domain shift* and it is a common problem across many machine learning related tasks (e.g. NLP [39], semantic segmentation [50], object detection [32] and visual classification [28]). The work of this thesis, however, focuses predominantly on this domain shift in activity recognition tasks for videos.

Due to prevalence of domain shift, it is important for researchers to be able to assess the capability of models across different distributions. One popular framework for doing so is called *unsupervised domain adaptation*, which, broadly speaking, involves using some labelled samples from one distribution and some unlabelled samples from another distribution in order to adjust the model to better estimate the latter. Although a number of variations exist of this framework, the particular flavour of unsupervised domain adaptation which is relevant to this work is often referred to as *closed-set*. This simply means that the set of possible labels for both distributions is the same. In this work however, as is done in many prior works [4, 8, 11, 23, 30, 33, 60, 61], it will simply be referred to as *unsupervised domain adaptation*. To be more specific, the usual setup will include some source domain D^s which is made up of a series of n input-label pairs $((x_1^s, y_1^s), ..., (x_n^s, y_n^s))$. Additionally, there is an unlabelled target domain D^t, which is made up of a series of m unlabelled inputs $(x_1^t, ..., x_m^t)$. The key to the unsupervised domain adaptation task is to produce some model f which can accurately predict the class of samples from the unlabelled target domain D^t. When evaluating the performance of the models after domain adaptation, ground-truth labels for the target domain are used, but these are not available during training.

2.3.2 Common Solutions

Solutions to the unsupervised domain adaptation problem, defined prior, can often be assigned to a few broad categories. Three of the most popular and relevant categories to this work are

discussed, along with some examples of works which utilise them. These include image-based as well as video-based methods as the majority of earlier works focused on images, whilst video domain adaptation has only recently become popular.

Discrepancy-Based Methods These focus on defining some form of distance function which can record the difference that the model perceives between the two domains. This distance score can then be used as a loss value (commonly alongside a standard classification loss on the labelled source samples to avoid collapse) which is backpropagated throughout the model in order to disincentive any separation in feature or classification vectors based on domain difference. The hope of these methods is that this in turn enforces a focus on the class semantics instead and eliminates negative effects of the domain difference. There are a number of notable metrics to achieve this. Maximum Mean Discrepancy (often abbreviated to MMD) [18], for example, is popular among early image-based domain adaptation works. Simply put, it can be thought of as the distance between the mean of each domain's feature embeddings. Two examples of models which use this are the DDC [54], which freezes the first seven layers and minimises MMD^2 and cross-entropy loss simultaneously on the final layer, and the DAN [27] which minimises maximum mean discrepancy across multiple layers and kernels. Another popular distance metric, CORAL [49], instead uses the second order statistics. The distance between the covariance matrices of each domain's features is simply minimised in order to aid the adaptation. Interestingly, [38] shows how both MMD and CORAL can complement one another in the domain adaptation process. Lastly, the Kullback-Leibler divegence can also aid in domain adaptation by acting as a distance function between output prediction distributions of each domain. An example of this can be seen in a recent work [34], whereby it focuses on aligning the domains in the final classifier layer.

Adversarial-Based Methods Inspired by the rise of Generative Adversarial Networks [16], these usually consist of feature extractor and domain discriminator modules which are played off against one another in a minimax style game. Whilst the domain discriminator aims at determining which domain the extracted feature has come from, the feature extractor works in the opposing direction, to produce features which are similar across domains. The most popular and early example of this is the Deep Adversarial Neural Network [15], which shows that standard backpropagation can be used for adversarial training when a gradient-reversal layer (which simply multiplies the gradient by a negative constant on the backward pass only) is placed between the feature extractor and domain discriminator. This addition means that any knowledge gained by the domain discriminator is fed backwards so that the feature extractor can exploit it in order to fool the discriminator better next time around. Alongside this adversarial game, [15] also has a classifier network to produce loss values for the labelled source samples. Another example of a popular adversarial, image-based domain adaptation method is ADDA [53]. Unlike [15], the ADDA network uses separate feature extractors for each domain and utilises inverted labels instead of a gradient reversal layer to simplify the process of encouraging the feature

extractor to work in the opposite direction to the discriminator. Adversarial based approaches have become very popular over the last few years and there are a number of examples of them being used in the more difficult unsupervised **video** domain adaptation setting [4, 30, 33, 60] (discussed later in this chapter).

Pseudo-Labelling-Based Methods As the target domain data is unlabelled in the unsupervised domain adaptation setting, these methods are designed to provide labels which are as accurate as possible for those unlabelled samples, so that traditional classification losses (e.g. cross-entropy loss) can be used on both domains. Much like the prior methodologies, pseudo-labelling can be used as the sole method or as a supplementary method. Broadly speaking, pseudo-labels are often calculated using either some form of distance measurement between embedding vectors or via the output predictions of the final classification layer. [3] is an example of a method which assigns pseudo-labels to target samples based on the source class prototype (where a prototype is an average of all source feature vectors in a given class) with the maximum cosine similarity. They then employ a threshold which discards the pseudo-labels below a certain level of cosine similarity in order to stop likely incorrect labels from affecting performance. On the other hand, [40] utilises predictions from a number of classifiers to obtain pseudo-labels for samples from an unseen target domain. Pseudo-labelling techniques can also be successfully transferred to the video domain adaptation setting, as seen in [11, 23, 33, 61], which all use classifier output predictions to perform successful adaptation on egocentric video datasets.

2.4 Unsupervised Video Domain Adaptation for Activity Recognition

Having covered a few of the main categories of unsupervised domain adaptation solutions, a focus is now given to a few of the most recent and relevant video specific works in the field.

2.4.1 Temporal Attentive Adversarial Adaptation Network

The authors of [4] stress the importance of aligning features both spatially and temporally, across domains. In order to achieve this, they introduce the TA^3N architecture, which utilises the *Temporal Relation Network*, described in section 2.2, as a backbone. Spatio-temporal features are produced for each n-frame relation branch. To better align these across domains, the features from each branch are given a weighting depending on how well a domain discriminator is able to classify them. Those features which have a high entropy of domain prediction are given a lower weighting as these features will already be aligned quite well across domains, as they clearly have no domain discriminative characteristics. Conversely, those features which are easily identified to be from one domain or the other are given a much higher weighting. This is because those features clearly include semantics which are particular to their individual domain and that

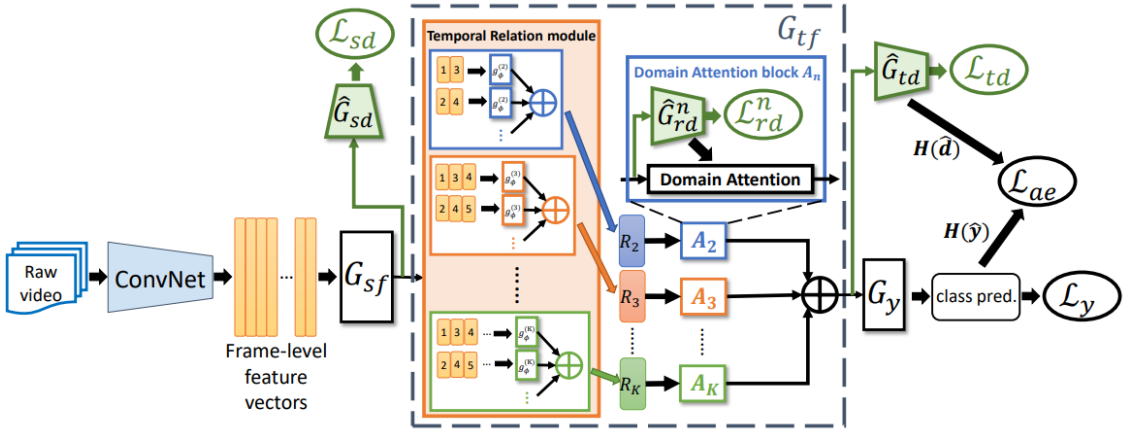


Figure 2.4: Diagram of TA^3N method. Figure taken from [4].

can distract from the actual semantics (of the classes) which are desired. These weighted features are then finally aggregated for classification and further domain adversarial training. A diagram of the TA^3N method can be seen in figure 2.4.

The idea behind this work is to better align those parts of the domain-specific videos which are causing the degradation in results across domains. They show a clear increase across a number of new dataset splits which are particularly curated in order to simulate a larger domain gap than seen before. Also shown is that the inclusion of the weighted TRN branches vastly improves on simple temporal pooling of spatial features.

2.4.2 TCoN

Due to the constraints of video understanding architectures, most domain adaptation methods have to work with frame-level or segment-level (clip-level) features. However, the authors of [33] point out that some segments of action videos can be less important than others. This can often be due to background frames that are irrelevant to actions and are common in naturally captured videos. Furthermore, they point out that simply matching distributions of segment features to align domains can be harmful, as those irrelevant segments will be included and the order of the action would be neglected. For this reason, they propose the *Temporal Co-Attention Network* which focuses on selecting certain segments of the video over others when aligning the source and target domains. This is achieved by first calculating intra-video and inter-domain attention scores for each segment in a random pairs of source and target videos, where the attention is simply calculated as the inner product between the segment-level features. They name this co-attention after a prior work in natural language processing [57]. These co-attention scores are then used to build a co-attention matrix which ensures that those segments deemed important inside each video, and similar across domains, would be given more attention than others. Additionally to the attention weights for each segment, target-aligned source segments are produced using

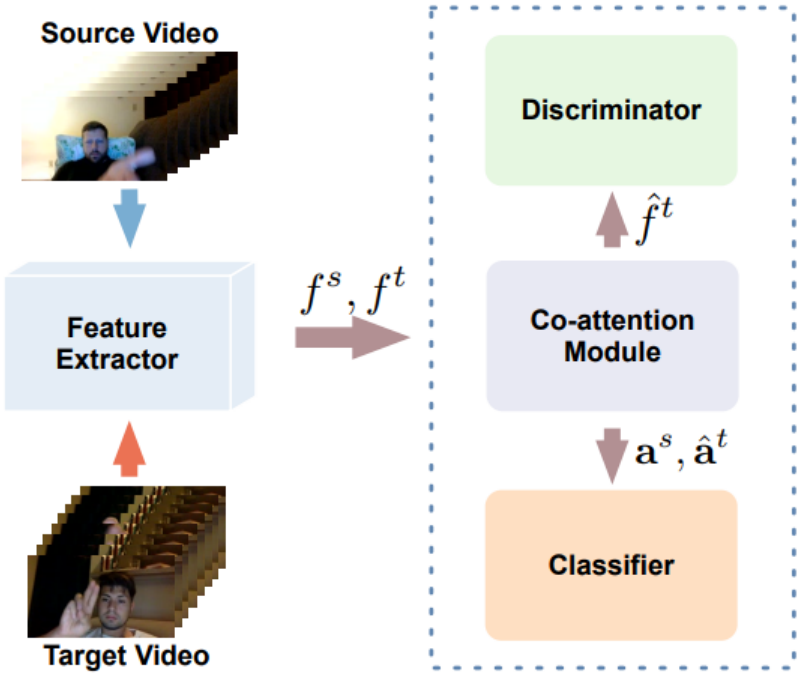


Figure 2.5: Overview of the *TCoN* method. Figure taken from [33].

the source segments and corresponding co-attention weight between the target segments and each source segment. The authors state that the idea behind this is to produce features that fall on the source distribution, but retain that semantic meaning of the corresponding target distributions. The co-attention scores and the target-aligned source segment features are then utilised in two different ways. The attention vectors are sent to a classifier in order to produce a weighted average prediction logit for all segments in a video. For source samples, this resultant prediction is simply compared against the ground-truth in cross-entropy loss, whilst the most confident target logits are compared against a pseudo-label which corresponds to their prediction. On the other hand, the target-aligned source segment features enter a two-branch discriminator module with the original source and target segments. The first branch of the module tries to distinguish target-aligned source segments from target segments. The second branch tries to distinguish target-aligned source segments from source segments. The idea behind this is to use the intermediate target-aligned source segment features as a way to align the target and source domains to a common space. Ablation results show that this method is effective in boosting performance over just using the co-attention weights. Furthermore, their *TCoN* performs consistently better than a number of networks [4, 20, 52, 56, 62] across multiple different domain adaptation dataset splits [4, 20, 29]. Figure 2.5 provides an overview of the *TCoN* method.

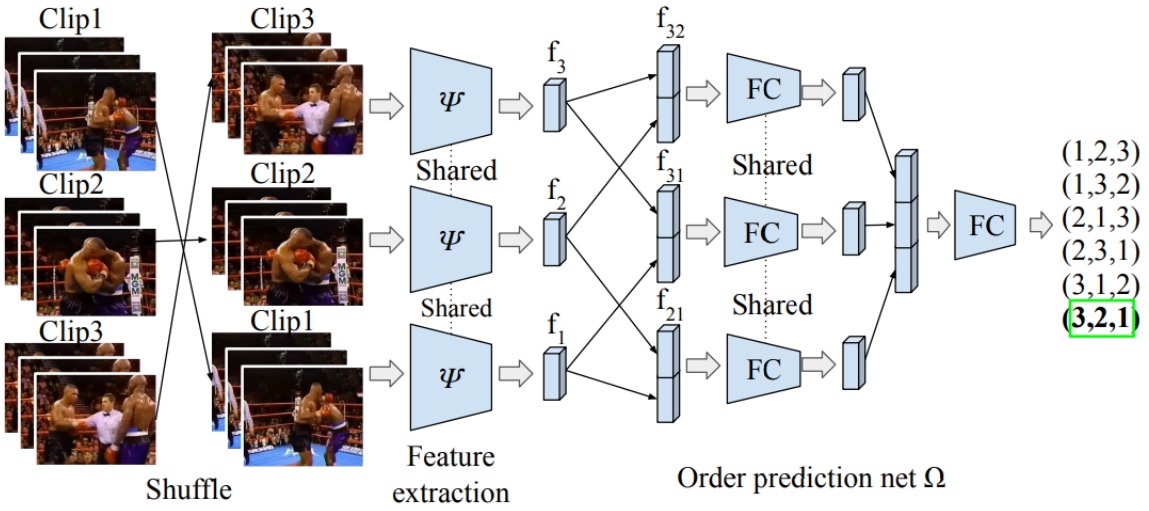


Figure 2.6: Diagram of clip-order prediction network used in the *SAVA* method. Figure taken from [8].

2.4.3 Shuffle and Attend

The *SAVA* method, introduced in [8], also utilises domain adversarial learning but puts additional emphasis on the ability of self-supervised tasks in order to encourage models to suppress context-specific information. Specifically, the authors of [8] point out that although a domain adaptive network would not want to memorise background information (that is likely specific to domains), it would most likely profit from modelling the temporal evolution of actions. This is due to the fact that the abstract notion of an action is more likely to be generalisable than the exact circumstances it is performed in. In order to achieve this, all clip-level features, from different parts of the input videos, are given indices and are then randomly shuffled. Seeing as the order of the video frames is already known, this requires no additional labelling. A clip order prediction network (as depicted in figure 2.6), which is made up of feature extractors and fully connected layers, which share weights, is used to produce an index prediction for each of the shuffled clips. This is implemented alongside both a domain adversarial learning branch, which attempts to align clip-level features from different domains, and a standard cross-entropy based action classification branch for the source samples. They are able to show a marked improvement over simple adversarial learning, when the self-supervised clip order prediction objective is included. Their method is able to outperform DANN [15], ADDA [53] and TA3N [4] methods on the NEC-drone dataset [7] after pre-training on Kinetics [21]. Although their method also boosts performance on the UCF-HMDB dataset [4] when compared with the TA3N model, it struggles to reach the best result on one of the two variants when compared to TCoN [33]. However, there is no mention of why this is.

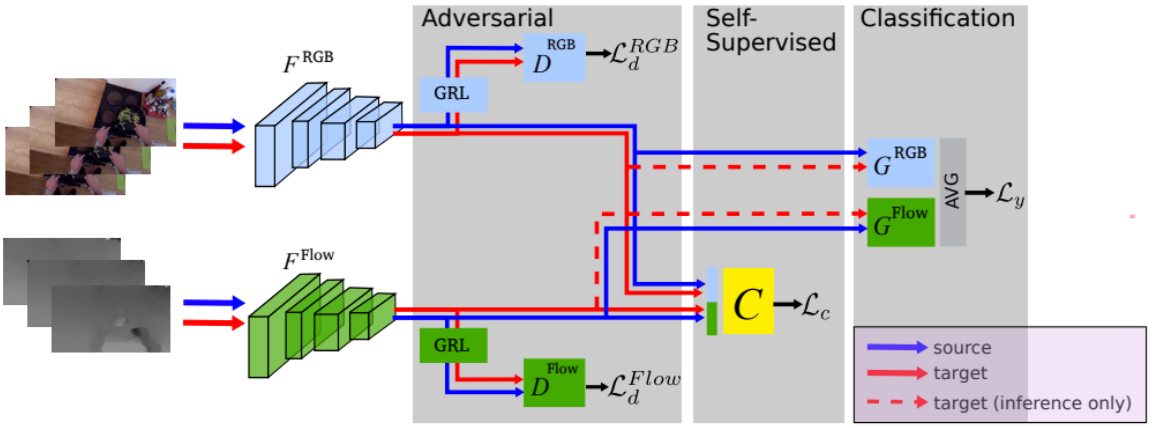


Figure 2.7: Diagram of *MM-SADA* method. Figure taken from [30].

2.4.4 MM-SADA

The authors of [30] are some of the first people to discuss unsupervised domain adaptation for fine-grained action recognition, by using the EPIC-Kitchens-55 dataset. This data is important (explained further in chapter 3) as it provides natural, first-person actions on a large scale, which popular unsupervised domain adaptation datasets at the time did not do [4, 21, 24, 48]. The *MM-SADA* method combines the multi-modal success of I3D [2] video backbones with domain adversarial learning and self-supervised methodologies. In particular, clip-level features for each modality are used in three simultaneous branches. The first performs adversarial domain learning on each modality separately. This supports intra-modality domain alignment. The second branch takes the features from both modalities and then performs self-supervised classification on whether they correspond. Modality-specific clips from the same action are classed as corresponding, whilst those from differing actions are not. This is meant to strengthen domain alignment and is shown to do just that in the ablation studies by providing a boost of a couple of percent when compared to using adversarial learning only. Lastly, there is a third branch for standard classification. The authors simply average the logits produced by each modality and then perform cross-entropy loss for backpropagation. A depiction of these three branches can be seen in figure 2.7. Of course, as the target samples are not labelled in this setting, only the source samples are sent through this stage during training. The combination of modalities in classification is proved by the authors to improve over the use of individual modalities. The results from this method show a clear improvement across all but one of the six domain gap settings in EPIC-Kitchens-55 when compared against three different methods [25, 27, 41].

2.4.5 Cross-Modal Contrastive Learning

The method presented in [23] utilises two contrastive learning objectives (whereby the features of similar instances are learned to be more similar than the features of dissimilar ones) in order

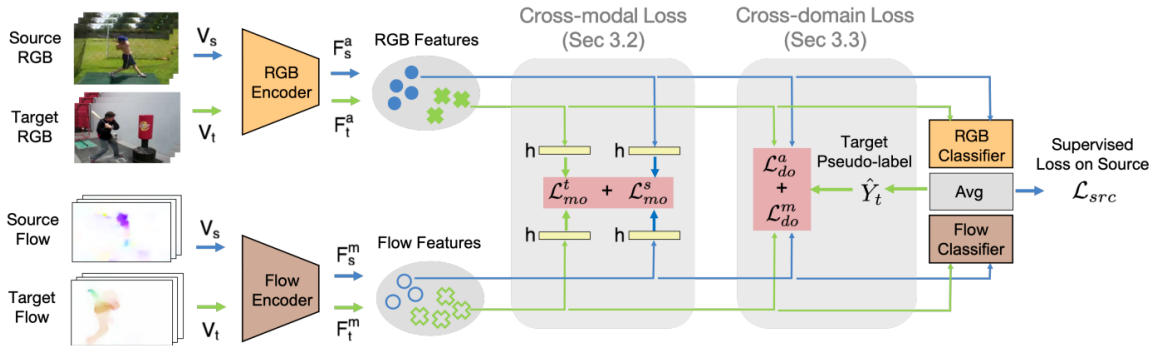


Figure 2.8: Diagram of cross-model contrastive learning method from [23].

to better align the domains. The first they call a cross-modal loss. The features produced by the different modalities of the same video are pushed to be more similar, whilst those features from other videos are used as negative sample references. The second contrastive learning objective is called the cross-domain loss. Features from different domains, but the same class and modality, are learned to be more similar than those videos from both different domains and different classes. However, this second contrastive loss depends on knowing the labels of the target samples, which is not the case in unsupervised domain adaptation. Figure 2.8 shows the process by which they acquire the labels needed for the target train samples. Features from the target domain are fed through a two stream I3D network [2], pre-trained on the Kinetics dataset [21], in order to produce predictions for each sample. The average of the logits produced from each stream is taken, and any samples with a prediction higher than a certain threshold T are used in the cross-domain contrastive learning. This is a common pseudo-labelling strategy and is used in another domain adaptation method [11], which is discussed later in this section.

Although their method is able to outperform TA3N [4] and SAVA [8] methods on both variants of the UCF-HMDB dataset [4], it does fall short of the performing better than TCoN [33] on one of the domain gaps in this dataset. Furthermore, the method shows marginal improvement over [30] on the same dataset but only manages to increase performance on 4 of the 6 domain gaps in the EPIC-KITCHENS-55 [9] dataset.

2.4.6 Cross-Modal Interactive Alignment

In [60], the authors hypothesise that different modalities can contain unique domain-transferable knowledge. They support this with an example of a "wash cup" action. As the sound of water is the same across domains, the authors claim it is more transferable across domains for classifying the verb. However, it does not include information that could help transfer across domains for the noun, which maybe something like the RGB modality could. For this reason, the *CIA* method is proposed to allow modalities to complement one another before the alignment process. This is different to other methods which either align domains separately for modalities or use self-supervision tasks to align modalities. The method itself is made up of two parts, the Mutual

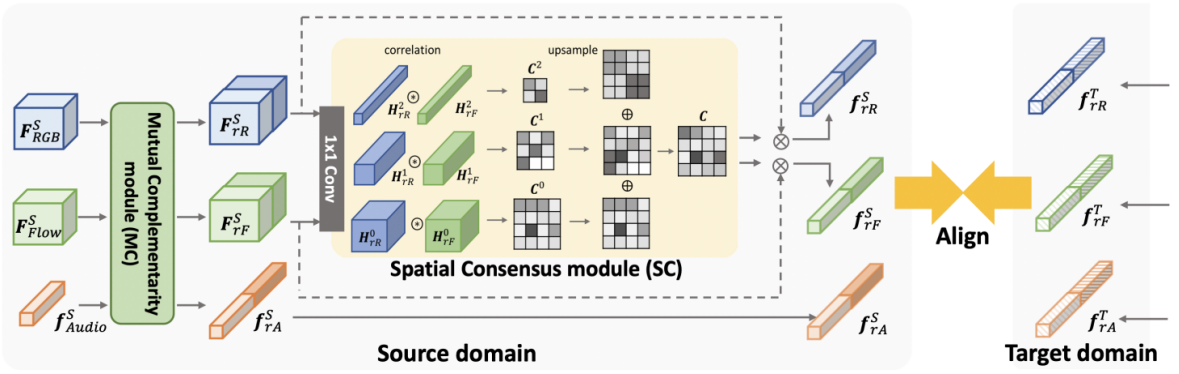


Figure 2.9: Diagram of *CIA* method. Figure taken from [60].

Complementarity Module and the Spatial Consensus Module, as seen in figure 2.9. The former outputs a feature for each modality which has been refined for better transferability. This is achieved by the concatenation of a cross-refined feature and a self-refined feature, per-modality. The cross-refined feature is produced by first applying global average pooling on the frame-level features from the other modalities, concatenating the result and then sending this through a non-linear, two-layer neural network. This is then used to weight the importance of the different channels in the current modality. As way of preserving any unique information from the current modality, a self-refined feature is also produced. Similarly to before, the current modality's feature is transformed with global average pooling and is then sent through a different non-linear, two-layer neural network. The original feature vector multiplied by the cross-refined weightings is concatenated with the original feature vector multiplied by the self-refined weightings to produce the final output of the Mutual Complementarity Module. The visual modality output vectors of this module then feed directly into the second step of the method, called the Spatial Consensus Module. This simply maps all visual modalities to the same space and then assesses their agreement on different spatial regions at a number of scales. The reason for this is to highlight those spatial regions which have shared consensus across modalities. As figure 2.9 shows, the outputs of the Spatial Consensus Module are then simply combined with the outputs of the previous module via element-wise multiplication. After both modules, the authors then perform simple adversarial domain alignment on the refined features alongside a standard cross-entropy loss on a fused video-level feature, where the video-level features of each modality are simply an average across frame-wise features. Ablation studies provided show that both modules improve performance when utilised together, as well as a clear increase when using both self and cross refinement inside the Mutual Complementarity Module.

The *CIA* method strongly outperforms a number of previous methods [4, 8, 23, 30, 33, 47] on both variations of the UCF-HMDB dataset [4]. Furthermore, the method performs consistently better than previous video domain adaptation works [23, 30, 47] on the egocentric EPIC-Kitchens datasets [9, 10] (only falling short of the performance of [47] on one of the six domain gaps of

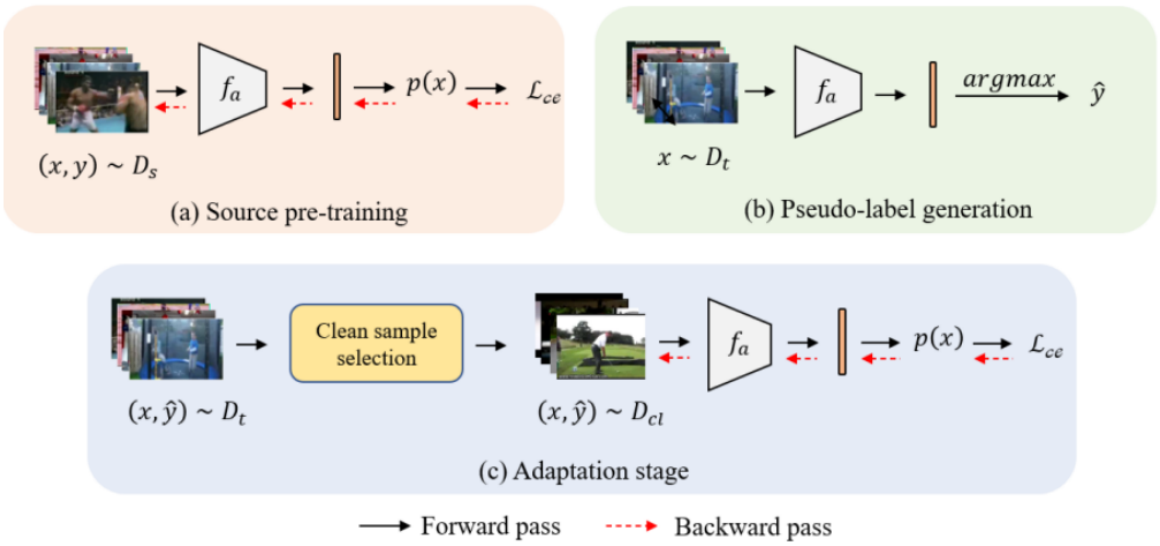
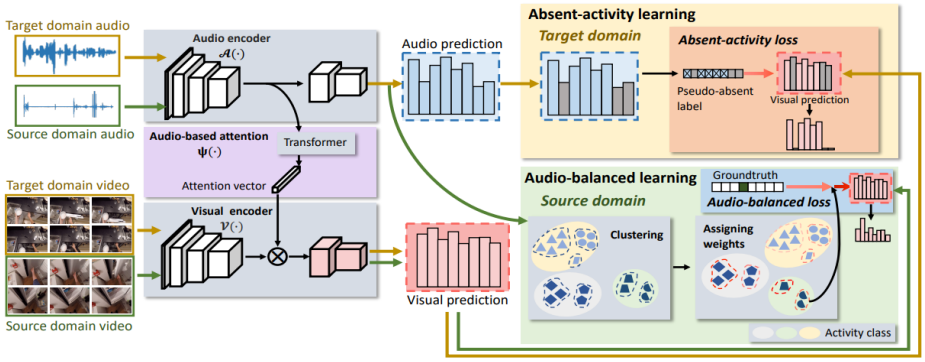


Figure 2.10: Stages of the *cleanAdapt* method. Figure taken from [11].

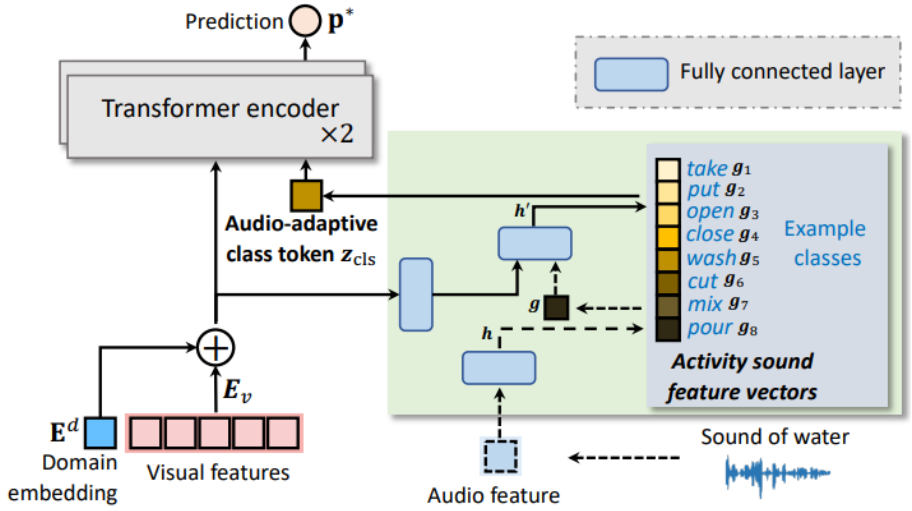
EPIC-KITCHENS-55).

2.4.7 cleanAdapt

Much like the pseudo-labelling stage of the method presented in [23], the cleanAdapt method from [11] uses the confidence of the predictions on target samples to decide which samples should be used for domain adaptation. What makes their method interesting is that selective pseudo-labelling of the target domain is all they rely on whilst not even accessing the actual source samples during domain adaptation. The particular setting of their paper is called "source-free" domain adaptation. It follows the same aims as all of the other methods discussed in this chapter, the only difference is that the actual source samples are not allowed to be used whilst the target samples are. The point of this setting is to simulate a real-world scenario whereby a model may be trained on sensitive data that needs to remain private for later use. Therefore, only the unlabelled target samples and source-trained model may be used for the domain adaptation process. In order to produce the pseudo-labels for the target domain samples, the authors of [11] first train a two-stream I3D network [2] (pre-trained on Kinetics [21]) with labelled source data. That source data is then discarded and the target domain samples are fed through the frozen source-trained model to produce prediction scores for each. The top x% most confident samples from each pseudo-label class are then picked to be used for a single epoch of adaptation training. With each new epoch, the process of pseudo-label generation is repeated. The reason for the filtering by confidence comes from a hypothesis they make which states that those target samples with confident predictions are often more likely to be correct. This notion is later investigated and confirmed for a particular setting in chapter 3 of this thesis.



(a) Stage one.



(b) Stage two.

Figure 2.11: Diagram of the audio-adaptive activity recognition method from [61]. Figures taken from [61]

Although they cannot outperform some of the prior methods on all of the EPIC-Kitchens-55 domain gaps, they still show good adaptation performance despite not using the source video samples during adaptation like most of the other methods quoted. Notably, they are able to consistently outperform the original MM-SADA baseline [30], with is not a source-free method. Furthermore, they showcase that they are able to greatly increase performance on the UCF-HMDB dataset [4], when compared to the previous state-of-the-art image-based source-free techniques.

2.4.8 Audio-Adaptive Activity Recognition

The authors of [61] put particular emphasis on utilising the audio modality as a tool to complement the visual, due to its higher domain invariance for certain characteristic actions (e.g. the sound of water pouring is not often very different between domains). They achieve this with

a two-stage network, as seen in figure 2.11. The first adjusts the visual features according to any complementary information from the audio modality. Audio features are extracted from an encoder (pre-trained for audio activity recognition) and then fed into a transformer network to produce an attention vector. This is then multiplied by the visual features extracted from a second encoder in order to suppress those channels which the audio deems to be irrelevant. Both the visual encoder and transformer are trained by two objectives, each focusing on data from one of the domains. The unlabelled target domain data is utilised by "absent-activity learning". This consists of finding the predictions of the sample's corresponding audio modality and then defining a set of absent activity classes. These are the classes with the lowest probability values from the audio modality. "Absent-activity learning" simply then encourages the network to suppress predictions for these classes in the visual modality. Conversely, the labelled source data is used in the second objective, called "audio-balanced learning". Due to the imbalance in the label distributions and frequencies of objects (e.g. the "open" activity may occur mainly on doors in one domain, whilst it may occur predominately on packages of food in another), they use the corresponding audio features to create sub-clusters of visual features within each activity class. The audio is used here as it can aid in identifying material, and therefore potentially different types of the same action. Those samples from smaller sub-clusters are then simply weighted higher inside the loss function. The idea behind this is to encourage the network to generalise well to data which has an imbalance of classes. These two objectives are used simultaneously to produce a visual encoder and transformer which can then extract both domain invariant visual features and pseudo-labels for the unlabelled target data. It is this output which is then used for the second, and final stage, of their method, the audio-infused recogniser. Simply put, this stage consists of a transformer encoder to produce a final prediction which, as input, is fed the audio feature, visual features and a learned domain-specific embedding. The audio feature is first sent through a couple of fully connected layers in order to produce an audio-adaptive class token. This initially biases the learning towards the audio modality so that domain-specific semantics from the visual modality do not dominate in the early stages of training. Moreover, the domain-specific embedding on the other hand is a learned vector which encourages the transformer to suppress the domain-specific information from the visual features by mapping both domains into the same space.

Although the first stage of training alone does suffice as a domain adaptation network, the authors show in ablation studies that the second stage improves the performance across two large egocentric datasets [9, 44]. Furthermore, they achieve the best results on the domain adaptation benchmark of [9] (as of the time of this thesis) when compared to previous works and show that the SlowFast [13] video backbone is more effective at video feature extraction for domain adaptation than the I3D [2].

2.5 Relation to Upcoming Work

The work in the following chapters focuses on the pseudo-labelling approach to unsupervised domain adaptation for video data. As can be seen from the literature explored in this chapter, pseudo-labelling is one of the more popular method choices, alongside domain adversarial learning, for video-based unsupervised domain adaptation. The work of [61] creates audio-advised pseudo-labels for the target data so that both source and target data can be used in the final classifier transformer. Additionally, the work of [11] uses solely pseudo-labels to adapt a source trained model to the new target domain, whilst never accessing the actual source samples during adaptation. Due to its competitive performance and pseudo-labelling only approach, it is used as a baseline in chapter 3 so that new pseudo-labelling methods can be both compared fairly and be easily implemented into an existing adaptation regime.

Furthermore, multi-modal data (in the form of RGB frames, optical-flow frames, and audio features) is used throughout the work of chapters 3 & 4. This is because it is shown to be an important factor in producing competitive models for both video modelling and domain adaptation. The I3D network [2] covered in section 2.2 shows that the inclusion of an optical-flow stream alongside the RGB one consistently improves video understanding and is utilised by the cleanAdapt method [11] covered in section 2.4 to bolster performance in domain adaptation. Additionally, the methods [60, 61] both include the audio modality and show that it improves generalisation performance.

CHAPTER 3

PSEUDO-LABELLING ON EPIC-KITCHENS-55

Actions are inherently variable-length phenomena; one individual may take 30 seconds to chop an onion, whilst another may take 300 seconds. Despite this, most architectures cannot handle this variability in its entirety. For example, convolutional neural networks require a standard sized input. As these are often used in domain adaptation works, it means that some form of concession is often employed to combat the variable nature of video when adapting. A couple of recent works [11, 30] simply sampled standard sized clips randomly each time a video was encountered by the network. In conventional training, where ground-truth labels exist, this can be good as it tends the network towards classifying all parts of the video as the same class, whilst not having to worry about the variable lengths of each video. However, this notion is not guaranteed to come about in a video that is from an unseen domain, which the network is not necessarily predisposed to understand. If these segments are then being used to produce pseudo-labels for adaptation to the unseen domain (which is the case in [11]), then there is a potential that simple random sampling of clips could include information harmful to adaptation, or miss information that could be more helpful. As a result, this chapter describes the work conducted to confirm that these variations do in fact exist in the unseen domain, how they affect adaptation and whether different clip sampling techniques could be introduced to improve both pseudo-label and adaptation accuracy.

The chapter begins in section 3.1 by explaining the unsupervised domain adaptation dataset that was used in this work - a variation of the EPIC-Kitchens-55 dataset [9] - before moving on to the motivation behind exploring pseudo-labels in the domain adaptation task, described in section 3.2. Following on from this, a recent work which utilised pseudo-labelling effectively is described, re-implemented and analysed in section 3.3 to form a basis for comparison against the various pseudo-labelling methods that are introduced in the following section, 3.4. Lastly, an analysis of

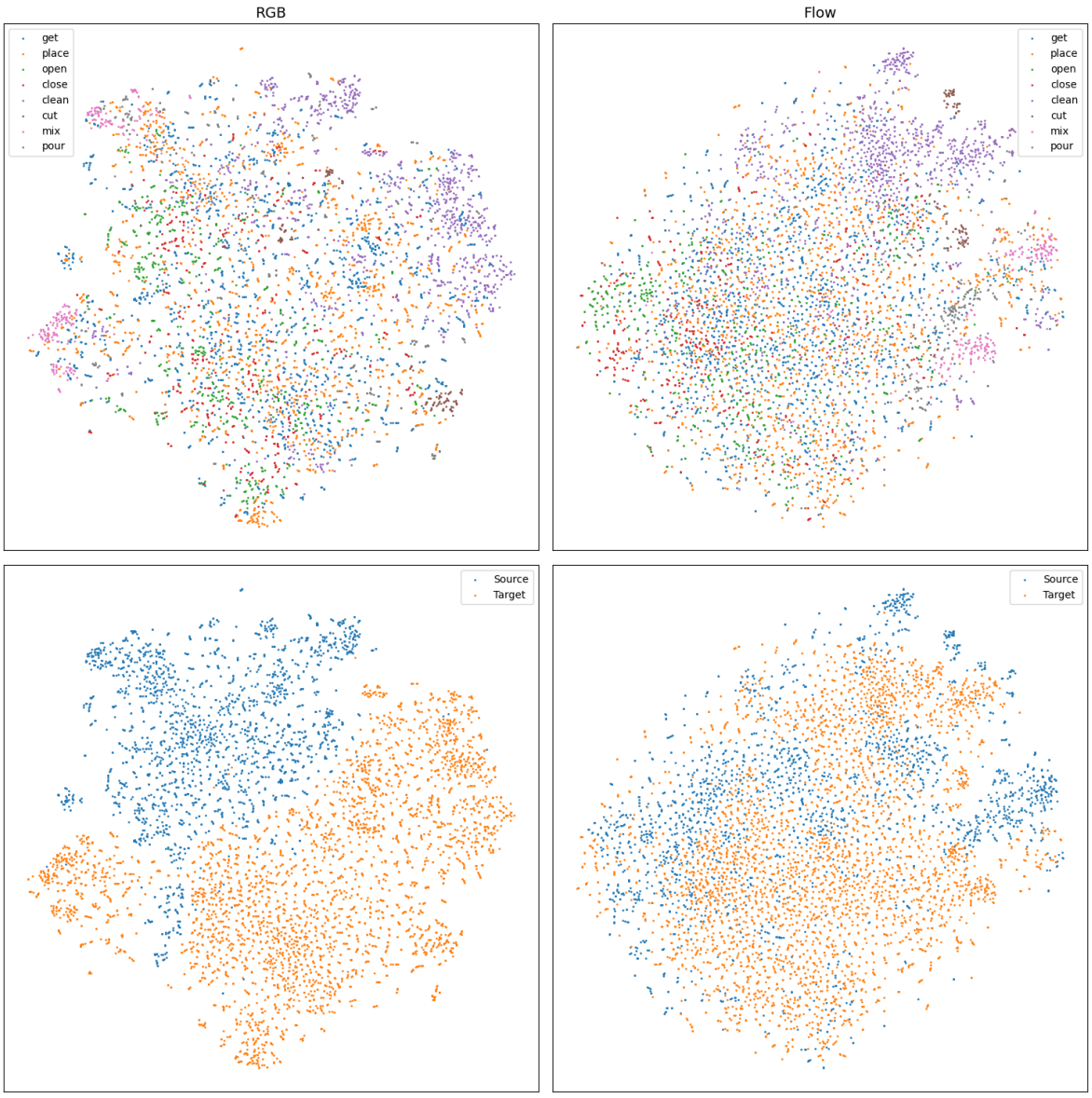
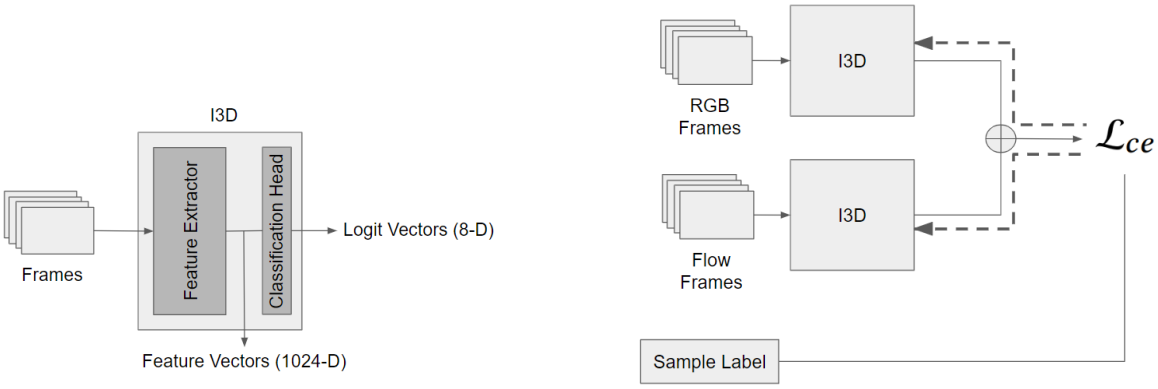


Figure 3.1: tSNE plot of the UDA variation of EPIC-KITCHENS-55. **Left column** are RGB features. **Right column** are optical flow features. **Top row** is coloured by class. **Bottom row** is coloured by domain.

how some of these enhanced pseudo-labelling methods compare against the re-implementation of [11] is detailed in section 3.6.

3.1 EPIC-Kitchens-55 UDA Dataset Variation

The EPIC-Kitchens-55 dataset, introduced in [9], was curated in order to provide various computer vision tasks with large-scale egocentric data from a diverse range of locations and participants. To



(a) Two stages of I3D network.

(b) Source-only training of I3D networks.

Figure 3.2: (a) shows the two basic stages of the I3D network. Where features are the intermediate 1024-D vectors of each sample extracted before the final layer. (b) shows a general structure of the source-only training, where $\mathscr{L}_{ce}$ is the standard cross-entropy loss and the dashed lines represent backpropagation.

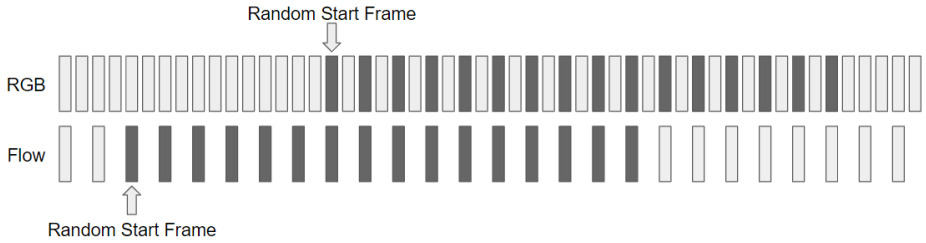
be more specific, the overall dataset consists of 55 hours of footage from 32 participants in their own kitchens (across several different cities and countries) as they conduct a variety of different cooking related activities. Each video is supplemented with narration from the individual filmed, as well as bounding boxes for object detection tasks. The aforementioned free-text narrations were further used to produce 125 distinct verb (take, close, turn-on, etc.) and 331 noun (pan, cupboard, cheese, etc.) classes for activity recognition tasks, by clustering parts of speech.

Although no particular emphasis was initially made on how the dataset could be utilised for the unsupervised domain adaptation task, this was later introduced by [30]. This variation utilises the three largest kitchens (based on the number of samples recorded) as three separate domains. Performance is only analysed on the 8 largest verb classes, as these constitute roughly 80% of the action samples from the chosen participants. A visualisation of the test set features from two of these participants, P08 (labelled "Source") and P01 (labelled "Target"), can be seen in figure 3.1. These are extracted from two (one for each modality) I3D networks, as described in [2], which only use data from P08 during training in order to showcase the domain shift present between the two kitchens.

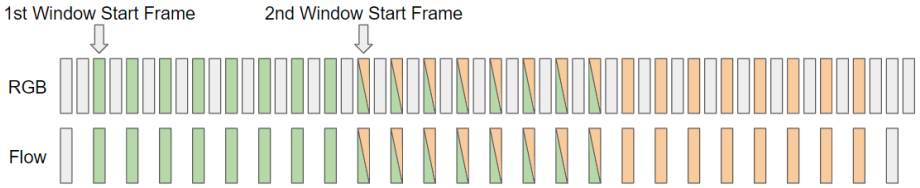
Since its inception in [30], this particular variation of the dataset has become a standard practice for determining the performance of domain adaptation models on the EPIC-Kitchens-55 data and is frequently referred to as the MM-SADA split.

3.2 Motivation for Pseudo-Labelling

The standard benchmark for unsupervised domain adaptation on the EPIC-Kitchens-55 dataset, as described in [30], utilises the I3D convolutional architecture from [2] as the video backbone model. Most successive papers on the topic perform experiments and analysis on the same setup



(a) Random frame sampling.



(b) Sequential frame sampling.

Figure 3.3: Visualisation of the different frame sampling approaches. Rectangles represent frames from each modality. Dark grey coloured frames are those which are sampled. Likewise, green and orange coloured frames are those which are sampled by window 1 and 2, respectively. Frames in (b) with two colours are sampled by both window 1 and window 2.

as way of a fair comparison. Although the I3D is covered in more detail in the previous chapter, as a reminder, figure 3.2a shows a basic visualisation of how feature and logit vectors are extracted.

The I3D network (briefly covered in section 2.2) relies upon a standard sized input to perform three-dimensional convolutional operations for feature extraction. For this reason, [30] extracted a set of 16 frames from a random position in each sample video in order to input into the I3D network. More specifically, two random frames (one for each modality) are first chosen within the video in such a way that, if possible, the resulting collection of frames do not overlap with surrounding actions. From these start points, 16 consecutive frames are then sampled from the RGB and flow modality frames (RGB frames are sampled with a stride of 2, due to the higher frame rate, as illustrated in figure 3.3a). Each time the sample is then re-encountered, a new part of the video is likely processed as the average length is much longer than 16. By randomly picking a different segment of the video each time, it was likely to enforce a common feature representation for the different parts of a single video. This is reflected in the testing regime of [30] which sums the logits from five equidistant windows across the video samples before making a prediction. However, it is of interest to determine whether such a notion does in fact come to fruition through such a training scheme.

In order to determine the behaviour of the I3D models on the different segments of videos, two I3D models (one for RGB and another for Flow inputs) are trained using the procedure described in [30] for classification on the P08 (source) domain, which is visualised in figure 3.2b. Once appropriately trained, the models are frozen and training-split samples from P08 (source) and P01 (target) are fed through the networks. Although, this time, a sliding window of the same number

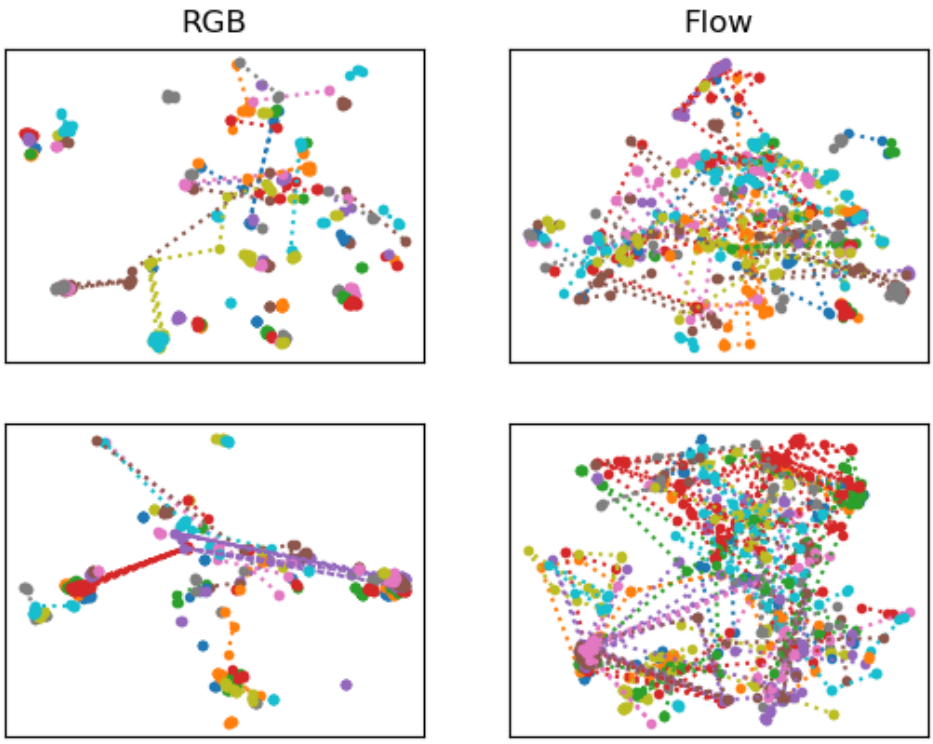


Figure 3.4: tSNE plots of 10 randomly chosen samples with 3 segments or more from each of the 8 verb classes. The first row shows seen training samples from D1 (P08) and the second row shows unseen samples from D2 (P01). Dashed lines connect different 16-frame segments from the same action video. Colours here are purely used in order to distinguish videos from one another.

of frames is employed to capture the feature representations of all the sections of the variable length videos. Of course, the entire video cannot be covered due to memory constraints. Therefore, an overlap of 16 consecutive frames between segments is used to achieve a relatively dense coverage of the video. An example of this can be seen in figure 3.3b. The overlap is visualised by those frames which have two colours in order to show that they are sampled by both windows. The start frames for the individual windows are calculated by first determining how many windows of length z^l can reasonably fit inside the total number of frames v^l . This is calculated as:

$$n = round(\frac{v^l}{z^l}) \tag{3.1}$$

where $round(\cdot)$ is a function that returns the nearest integer. In some scenarios, the value n may inadvertently produce windows which surpass the boundaries of the video. For that reason, the total length of the combined windows is calculated as:

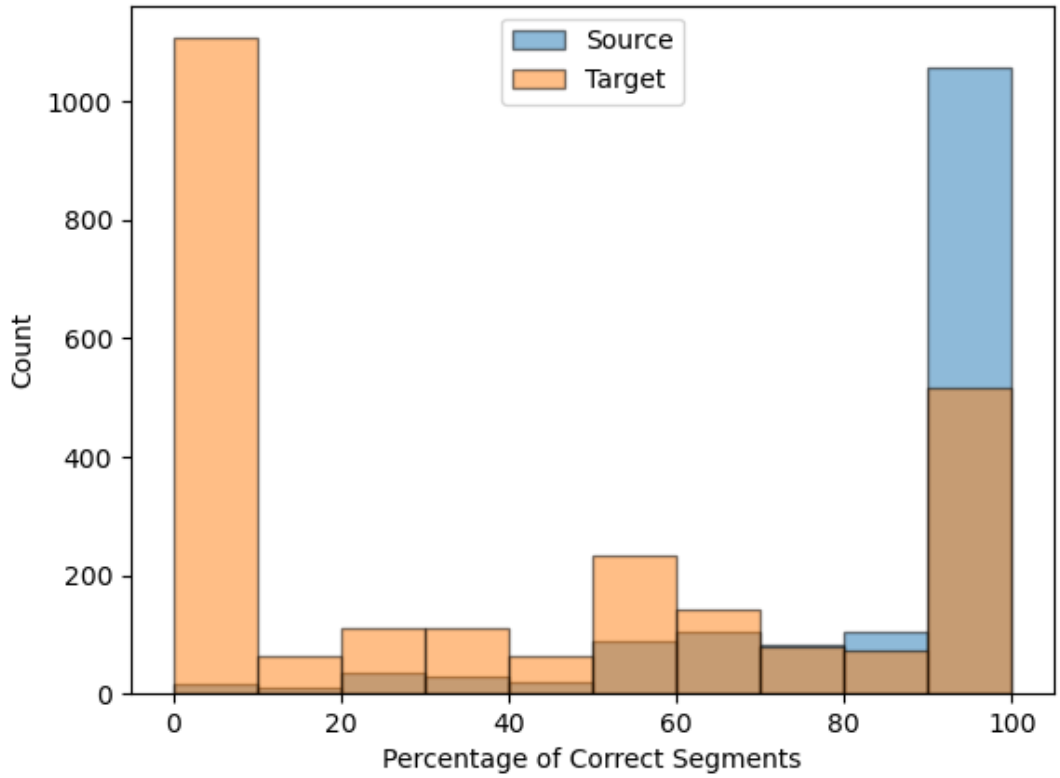


Figure 3.5: Histogram showing the percentage of segments for each video with the correct label predicted by the source-trained classifier. The source data has been seen already, whilst the target has not. The brown colour seen in the figure is simply a result of where Source and Target overlap. This is used so that the values for Source which are less than Target for the left most columns can still be seen and analysed.

$$z^t = (n \cdot z^l) - o(n-1) \tag{3.2}$$

where o is the number of overlapping frames between windows. Due to the fact that a few videos are shorter than 32 frames (the lowest is 30 frames in the UDA variation of EPIC-Kitchens-55), it is sometimes unavoidable to sample frames beyond the extend of the action. However, in order to mitigate it as much as possible, the value n is dropped by a value of 1 and the z^t is recalculated only if both of the following conditions are met: (i) $z^t > v^l$ **and** (ii) $z^l \leqslant v^l$. This produces a number of windows that only sample outside the extents of the action when absolutely necessary.

With a number of windows n, a series of starting frames $f = (f_1, f_2, ..., f_n)$ can be calculated with:

$$f_j = (\gamma - round(\frac{z^t}{2})) + (z^l \cdot j) - (o \cdot j) \tag{3.3}$$

where γ simply refers to the central frame of the video. Of course, more sophisticated methods of calculating the start frames are probably possible, but for the purposes of this work, this is sufficient.

The random window sampling strategy for training the I3D models is assumed to form a common representation for each video despite the variable length of the video. This is so stable predictions can be made without needing to handle the variable nature of the video within the architecture, as this can difficult and computationally costly. However, the results from sampling sequential windows, as described above, shows that this is not actually the case. Figure 3.4, which shows tSNE plots of randomly selected video representations, highlights that the feature vector representations within the model vary quite considerably despite the aforementioned goal of the training regime. This shows that a common representation of the video is not enforced across the temporal extent of the video and occurs in both seen and unseen samples. Without a common representation, it is more likely that a random window sampling approach could select sections of the video which are not optimised correctly for classification and cause the test accuracy to degrade.

A similar pattern of variations are also observed in the classification predictions. This points to a potential effect upon methods which pseudo-label the target domain using randomly sampled segments, such as [11], which is outlined in more detail in the next section. Figure 3.5, shows that a sizeable chunk (roughly 37.4%) of unseen target samples have mixed class predictions across segments, where at least one segment is predicted correctly. This means that if a random segment is used for pseudo-labelling with the source-classifier, then noise in a large portion of the pseudo-labels is very probable, which could in turn disrupt the effectiveness of the later adaptation. These same variations do exist in the seen source samples as well (albeit to a lesser extent), but have known, correct labels already, meaning that randomly sampled segments would simply bolster the correct representation in the feature space. This is in contrast to the target domain, which has no known labels at training time in the unsupervised domain adaptation setting. However, the figure shows that what a large portion of the target samples do have are regions of helpful information that can be recognised successfully by the source-trained model. This leads to two core questions of interest: **(i) how can a model be directed towards these areas of interest without supervision?** and **(ii) does this directed sampling improve over simple, random sampling of segments for pseudo-labelling and adaptation?**

3.3 cleanAdapt & Re-implementation

The work of [11] showcases how pseudo-labelling with randomly sampled video clips can perform considerably well on the EPIC-Kitchens-55 domain adaptation split. Named cleanAdapt, the

Model	Source Only	Best Adaptation	Difference
cleanAdapt [11]	46.00	52.70	+6.70
Re-run 1	46.93	50.93	+4.00
Re-run 2	46.93	52.53	+5.60
Re-run 3	46.93	51.87	+4.94

Table 3.1: Comparison of cleanAdapt results with re-implementation.

authors utilised the I3D setup from [30] and simply used the source-trained classifier to assign each target sample with a pseudo-label and confidence. In their method, the pseudo-label $\hat{y}$ is assigned by:

$$\hat{y} = \arg\max_{c} p_c \tag{3.4}$$

where p_c is the c^{th} value in the output probability vector obtained by using a softmax function on the addition of the logit vectors from each modality-specific network. The confidence score is then defined as the cross-entropy loss function, with the pseudo-label taking the place of the ground truth. As expected, those samples with the smallest cross-entropy loss are considered to be the most confident. The cross-entropy based confidence score is therefore calculated as:

$$s = -\sum_{c=1}^{C} \hat{y}_c^{one-hot} \log(p_c) \tag{3.5}$$

where $\hat{y}_c^{one-hot}$ is the c^{th} element of the one-hot encoding of the pseudo-label $\hat{y}$.

The authors explained that the model tended to be more confident of samples when the labels were correct. This is supported later in section 3.6 when varying the sampling rate did in fact show that on average the more confident samples were more accurate. They then used this measure of confidence to assess their earlier claim that a reduction in pseudo-label noise was the key to improving domain adaptation accuracy when training on those pseudo-labels. By utilising the top 10% most confident pseudo-labels, they are able to achieve a 6.70% increase in target knowledge after adaptation, on the "D1->D2" setting (P08 is the labelled source and P01 acts as the unlabelled target). This is a comfortable improvement upon the original domain adaptation method from [30], with a 3.20% increase.

Due to cleanAdapt's impressive adaptation results but reliance on randomly sampled segments, it is utilised as a baseline for comparison in this work. Unfortunately, the codebase for the paper is not publicly available (or at least not at the time of completing this work). For this reason, the methodology is first re-implemented using PyTorch so that it can be modified for new pseudo-labelling methods introduced in the next section. The re-implemented model is run for the "D1->D2" (where D1 = P08 and D2 = P01 domains from EPIC-Kitchens-55) adaptation process and compared against the results from the authors, as seen in table 3.1. The second re-run of the

model shows that it is able to achieve a similar adaptation result to the authors on the target domain, with only a 0.17% drop in final performance. Although the other two re-runs do not produce exactly the same results, they do still showcase marked adaptation improvement inline with the magnitude of the authors' results. It is for these reasons that the re-implementation is taken forward for further pseudo-label testing later on in the work.

3.4 Pseudo Labelling Methods

The aforementioned cleanAdapt method [11] emphasises just how much simple pseudo-label denoising can help improve domain adaptation. The random sampling of segments for the I3D source-only training, utilised by cleanAdapt, causes prediction variation across different segments of individual videos from the unseen target domain. This points to a source of refined information which might be able to improve pseudo-labelling accuracy over that of cleanAdapt. With this in mind, a range of new pseudo-labelling methods that target these variations with the aim of decreasing pseudo-label noise are proposed. If the label noise can be lowered with these new methods, then following on from the authors' hypothesis, it might also be able to improve domain adaptation as a result.

Therefore, the following subsections outline several pseudo-labelling approaches that are explored in order to compare against the existing cleanAdapt method. These are split into three categories: **(i) source-classifier based methods**, **(ii) target-centroid based methods** and **(iii) joint domain feature-space based methods**.

In all of the following methods, each video sample consists of two inputs, x_{RGB} and x_{Flow}, for the different modalities. The two modalities are of equal length (when accounting for the double stride used to sample RGB), but the length is some variable amount t which changes with each new sample. Moreover, segments are sampled across each video according to the process detailed in section 3.2. This means that an input for modality k can be thought of as:

$$x_k = (x_{k1}, x_{k2}, ..., x_{kn})$$

where n is the number of segment inputs and each x_{ki} consists of 16 consecutive frames. Once all are fed through the modality k network, a sequence of prediction logits (each with dimension 8, for the number of classes in the EPIC-Kitchens-55 unsupervised domain adaptation variation) is produced:

$$p_k = (p_{k1}, p_{k2}, ..., p_{kn})$$

It should be noted that the following methods do not represent an exhaustive list of pseudo-labelling methods for the different types.

3.4.1 Source-Classifier Based Methods

To begin with, the source classifier is employed, similar to [11]. The difference with the new methods described in this section is that all video segments produced from the sliding window are surveyed for potentially superior information. To be specific, three slightly different methods are compared against the re-implementation of the cleanAdapt method described prior.

Max. Segment takes the segment from each modality with the largest logit value, when compared to the other segment predictions in the same modality. These two most confident logits from each modality are then combined by addition and fed through a softmax to produce a pseudo-label. The confidence is then computed in the same manner described by cleanAdapt, in equation 3.5.

Mean Segment simply sums the logits p_k of each modality and then fuses the resulting modality-specific averages by addition to produce a single output logit. The pseudo-label and confidence for this averaged logit are then computed in the same manner as the first method.

Segment Majority centres around filtering segments based on the most popular predictions. To be specific, for each modality, pseudo-labels are produced for each of the p_{ki} logits using a simple softmax. c_k^{top} is chosen to be the class with the most segments classified as itself for modality k. All segment predictions with pseudo-labels that do not match c_k^{top} are then dropped. From the remaining subset, the most confident logit is chosen for each modality and then combined, as in the first method, for pseudo-label and confidence score calculation. It should be noted though that there can be scenarios where two or more classes get an equal amount of segments predicted for each class. In these tie-breaker scenarios, the logit with the highest value, out of those segments with a prediction that matches one of the tie-breaking classes, is chosen to go forward for pseudo-labelling and confidence calculation. This third method differs from "Max. Segment" as it chooses the most confident logits from a subset of the total segments which correspond to the most popular class. The idea behind this is to determine whether it is better at avoiding samples with very confident, yet incorrectly labelled, segments.

3.4.2 Target-Centroid Based Methods

Due to the capability of the source-trained classifier to pseudo-label unseen target samples, it is of interest to determine whether any further performance can be achieved by utilising the raw feature vectors as well. The recent work of [58] provides a good inspiration of how this may be achieved.

The authors of [58] utilised pseudo-labelling as a key component in their combinations of methods. For the pseudo-labelling section of their method, they first required a weighted video-level feature produced by aggregating the local-level features (which in the case of this work is translated to mean the segment features extracted using the sliding window). This is achieved by first computing a confidence (slightly different from the pseudo-label confidence discussed previously) for each of the local-level features x_{ki}, based on the predictions of the source-trained model p_{ki} :

$$c_{ki} = \sum_{c}^{C} \sigma_c(p_{ki}) \log(\sigma_c(p_{ki})) \tag{3.6}$$

where σ_c is the value from the c^{th} position in the output softmax vector.

A weight is then defined as $w_{ki} = 1 + c_{ki}$. The reason for this is that the confidence in equation 3.6 is minimised for those local-level features which have uncertain predictions. Therefore, those uncertain features are suppressed, whereas more confident local-level features are emphasised in the construction of the video-level feature v, as described below:

$$v = \frac{1}{k-1} \sum_{i} w_{ki} x_{ki} \tag{3.7}$$

These weighted video-level features v are then be fed back through the same source-trained classifier so that their softmax scores can be used to produce class centroids. These centroids are effectively weighted average feature vectors for each class. Samples are then assigned a pseudo-label based on the closest centroid (using $cosineDistance = 1 - cosineSimialrity(\cdot,\cdot)$). The centroids are calculated by the following:

$$centroid_i^{(0)} = \frac{\sum_{v \in D_T} \sigma_c(v) v}{\sum_{v \in D_T} \sigma_c(v)} \tag{3.8}$$

In the original paper, there is an additional step to refine these centroids. For example, for class i all of the video-level feature vectors assigned to $centroid_i^{(0)}$ are simply averaged to produce $centroid_i^{(1)}$. However, this addition is found to actually harm performance for this work, as discussed in section 3.6.1.

Although the pseudo-labelling method presented in [58] appears to help their work, the calculations do point out a potential issue. Due to the method's decision to prioritise those samples with high confidence during the aggregation, segments of incorrect, yet high confidence, predictions could potentially cause the video-level feature to shift into the wrong class. Specifically to the EPIC-Kitchens-55 dataset, 20.72% of RGB videos and 4.29% of flow videos (in P01 domain, after training on P08) have at least two segment features which predict above 0.75 probability for opposing classes.

For this reason, a slight modification to the centroid approach of [58] is proposed in order to see how it performs when pseudo-labelling target samples from the EPIC-Kitchens-55 dataset. The original, as described above, first aggregates the local-level features into video-level features, feeds these through the network and then uses the softmax score as a weight in the average of the video-level features. The difference with this new method is that no video-level aggregation is performed. The local-level features and their model softmax score are instead used in the centroid equation in order to more accurately represent what the model views as class specific regions in

the feature space, as we have already pointed out that the aggregation to video-level can cause confusion. The exact modification of the centroids is calculated as:

$$centroid_i^{new} = \frac{\sum_{x_{ki} \in D_T} \sigma_c(x_{ki}) x_{ki}}{\sum_{x_{ki} \in D_T} \sigma_c(x_{ki})} \tag{3.9}$$

The thought behind this is to place the target centroids in areas of the feature space that the source-classifier sees as most discriminative for each class by utilising any video segment, even if from a different class, as it might provide some helpful information. For example, a video of chopping could include a segment at the beginning where a cupboard door is still closing and therefore be predicted as "close". As this segment does include semantic information for the "close" class, it still might be helpful to include it in the "close" class centroid, despite coming from a video labelled something else. This is different to the placement of centroids calculated using equation 3.8 because they are placed based on which aggregated video-level features are most confident in prediction. It is discussed prior that this aggregation can cause predictions to change to the wrong class which in turn may cause the placement of centroids to be incorrect too, something which is hope to be mitigated by the new formulation of equation 3.9.

Once the new target centroids are created, a decision has to be made on how to pseudo-label the videos. In the original paper this is done simply as the the closest target centroid to the video-level feature. However, as this work deals with multiple segment features per video sample, the possibilities for pseudo-label assignment becomes more numerous. Taking inspiration from the methods described in section 3.4.1, three pseudo-labelling metrics are explored.

Segment to Centroid finds the closest centroid for each segment feature in the video sample. It then picks the class of the centroid with the smallest distance to the segment. The confidence is then calculated as the cosine distance between that segment and the chosen class' centroid.

Segment Average to Centroid picks the class of the centroid closest to the average of all segment features in a video. The confidence is calculated, similarly, as the cosine distance between the average of all segments and the chosen class' centroid.

Segment Majority finds a prediction for each segment feature based on the closest class centroids to each. It then picks the pseudo-label as the class predicted most frequently by the segments. As this scenario is more vague, there are two ways to calculate the confidence of the pseudo-labels. The first (labelled **Segment Majority V1** in section 3.6.1) takes the smallest distance of all the segments predicted for the chosen class with the associated class centroid. The second (labelled **Segment Majority V2** in section 3.6.1) averages the segment features predicted for the chosen pseudo-label class and computes the confidence as the cosine distance from this average to the chosen class' centroid.

The features of each modality are distinct from one another, as they come from separate I3D networks. Therefore, the methods described in this subsection produce two pseudo-labels, with two confidences, for each sample (for RGB and Flow). As a result, the final pseudo-label and confidence for each sample are simply taken from the most confident modality.

3.4.3 Joint Domain Feature-Space Based Method

The methods prior use the classification and feature vectors of the unlabelled target data. However, it is of interest to determine whether the inclusion of the feature vectors of the labelled source-train data could improve pseudo-labelling accuracy further. For this reason, a final approach is proposed to do just that. Inspiration for this particular method is taken from the work described in [31]. Although the methodology described in this work focuses on video retrieval across different domains, rather than domain adaptation, it still offers some helpful insights into how the source and target features can be utilised for pseudo-labelling.

In [31], each target sample is assigned the label corresponding to the closest source sample, when comparing the feature vectors. The confidence score for each pseudo-label is then decided based on the distance to the corresponding source class prototype. These class prototypes are simply the average of all the source features within a given class. Therefore, the confidence of a pseudo-label would be how far the target sample is from the area in the feature space that is on average classified as that pseudo-label.

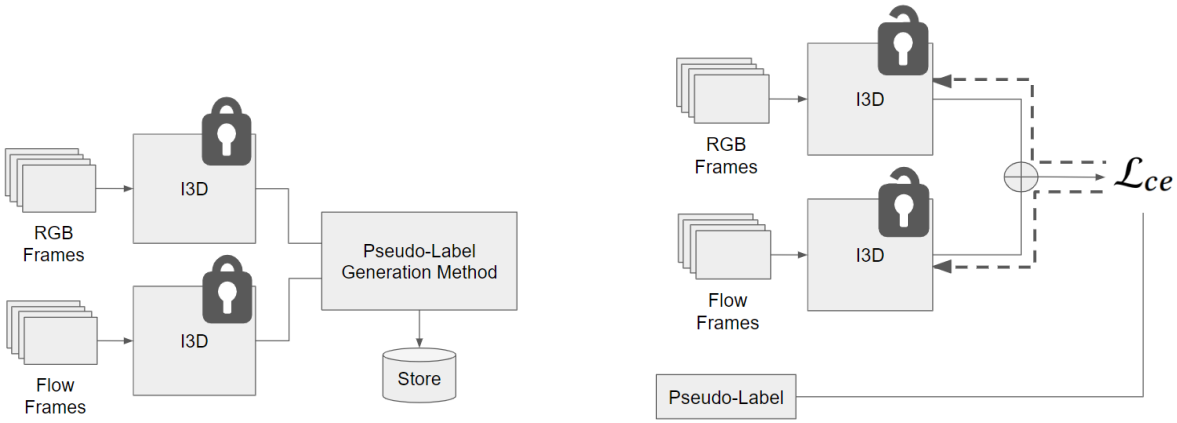
Using a similar idea to [31], the three methodologies of the prior methods are slightly modified to incorporate the source feature vectors.

Segment to Prototype finds the closest source segment feature to each of the target segment features. It then assigns the pseudo-label based on the class of the source segment which has the smallest distance to a target segment feature in the video sample. The confidence is then simply the distance from this target segment feature to the corresponding class prototype.

Segment Average to Prototype is the same as the method in [31] as the target segment features are first averaged to produce one vector for the video. The pseudo-label is then assigned based on the closest source segment feature.

Segment Majority is similar to the previous subsections. The main idea is to determine the prediction for each target segment feature based on the label of the closest source segment feature to it. The pseudo-label which is most present across segments is then used as the final pseudo-label for the overall sample. Much like before, the confidence can be computed in two ways. On the one hand, the confidence is simply the smallest distance to the corresponding source class prototype of those target feature vectors which have the chosen pseudo-label (labelled as **Segment Majority V1** in figure 3.9). Conversely, the confidence can also be measured as the the distance from the average of those target feature vectors with the chosen pseudo-label to the corresponding source class prototype (labelled as **Segment Majority V2** in figure 3.9).

Due to the need to search all source segment features in the space, for every target segment feature, a slight variation of the methods described above is additionally proposed. This simply involves using the source class prototypes for both pseudo-label and confidence assignment, so that only 8 distance measurements need to be computed for each target segment feature, instead of thousands. However, this only degrades performance when compared to using the source segments, as discussed in 3.9.



(a) Pseudo-label generation.

(b) Adaptation process.

Figure 3.6: The locked padlocks symbolise networks which have their parameters frozen. Conversely, the unlocked padlocks symbolise networks with parameters free to be adjusted via backpropagation (which again is visualised with the black dashed lines).

It should be noted that for simplicity, the same cosine distance from 3.4.2 is used for finding the closest source segments and prototypes, as well as quantifying the confidence. The exact distance metric used in [31] is actually cosine similarity and the confidence is computed as $e^{-cosineSimilarity(\cdot,\cdot)}$. However, this simply produces the same results as it is only an inversion of the very same distance. Therefore, the slight change of distance metric is deemed satisfactory.

As before in section 3.4.2, each of the methods discussed in this section produce two pseudo-labels and two confidence scores, one for each modality. The same process is followed as in section 3.4.2, whereby the pseudo-label is taken as the prediction of the most confident modality.

3.5 Implementation Details

The code used for the I3D network is from [1]. Much like [30] and [11], two I3D models, initialised with Kinetics [21] pre-trained weights, are trained simultaneously for the RGB and Flow modalities of the EPIC-Kitchens-55 unsupervised domain adaptation variation [30] by summing the logits outputted from each. The flow I3D network has 2 input channels, for vertical and horizontal, whilst the RGB has 3 input channels for each colour.

The source-train data is first used to produce the two source models (RGB and Flow). Similar to [11], the network is run for 100 epochs with a learning rate of 0.02 (reduced by a factor of 10 at 50 epochs), batch size of 48 and a stochastic gradient descent optimiser with 0.9 momentum and 1e-07 weight decay. At every epoch, a new randomly sampled set of 16 consecutive frames is used for each sample, and each modality, so that the network sees a different portion of the source data. This random segment position is different for each modality as in [30]. As covered earlier, a visualisation of the source-only training is shown in figure 3.2b.

For the adaptation, [11] is also followed. The main differences between this stage and source training are that pseudo-labels are generated before every epoch and the x% most confident are sampled. A visualisation of the pseudo-label generation process can be seen in figure 3.6a. Training is run for 60 epochs and the learning rate is lowered to 0.002, which then reduces further by a factor of 10 at epoch 10 and 20. The pseudo-labelling method used (whether cleanAdapt, "Max Segment" or "Max Agree") is simply plugged in to the pseudo-label generation step before every epoch (as shown in figure 3.6b). This change in method does not alter the training procedure of the network in order to produce a fair comparison with cleanAdapt [11].

Both the source-only training and adaptation stages utilise random 224x224 crops and random horizontal flip data augmentations on the input frames during training. During test time, a central crop of 224x224 is applied to each frame instead. Furthermore, following [11, 30], 5 equidistant segments from each video are used and the predictions averaged to produce the final classification of test split samples.

As the pseudo-labelling methods do not affect the overall training procedure of the network, the same I3D networks are used to produce the features and resulting predictions for each of the segments in each video clip. This means that any comparison of pseudo-label accuracy is fair and fully reflects the effectiveness of each method.

It should be noted that the re-implementation of the [58] (analysed and compared against in the target-centroids section of 3.6.1) work only includes the weighted video-level features, target centroid construction and pseudo-label assignment sections. Furthermore, the original paper produces its local-level features by feeding frames through a 2D-CNN and a TRN-style [62] module. Conversely, the method described in this chapter simply passes frames through an I3D network to produce segment-level features which are then used to replace the local-level features described in [58].

3.6 Results & Comparisons

As mentioned before, the authors of [11] hypothesise that by cleaning samples it can positively affect the domain adaptation accuracy. For this reason it is important to assess two key areas for the new pseudo-labelling methods. Firstly, the initial pseudo-labelling accuracy. This reflects the noise within the target pseudo-labels. By showing that a method outperforms cleanAdapt's random sampling approach, it can be selected to be used in adaptation to see if it performs as well or even better. Secondly, the adaptation performance. The accuracy on the target test set after utilising more accurate pseudo-labels is key to understanding whether just the removal of noise directly translates into better adaptation.

Subsection 3.6.1 outlines the initial pseudo-labelling accuracy of all of the methods described in section 3.4 and compares them to the re-implementation of cleanAdapt. Following on, subsection 3.6.2 analyses the performance of the two best methods (based on initial pseudo-labelling

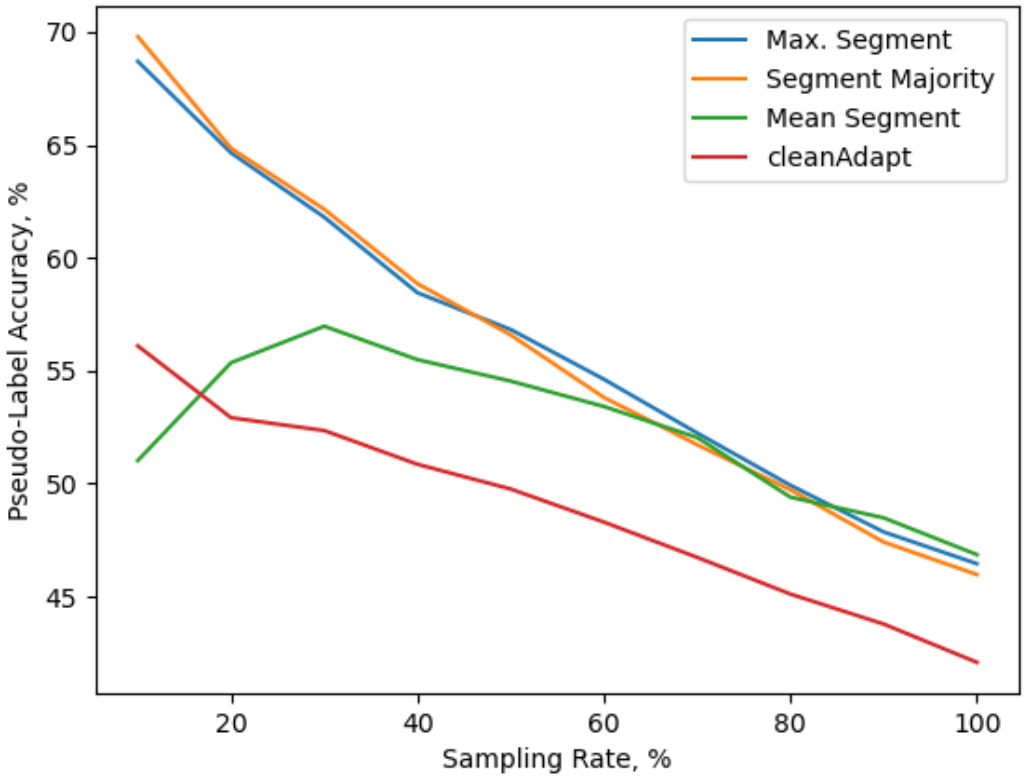


Figure 3.7: Initial pseudo-labelling accuracy of the new source classifier based pseudo-labelling methods from section 3.4.1 compared against a re-implementation of cleanAdapt.

accuracy) against the adaptation score achieved by the cleanAdapt re-implementation.

3.6.1 Initial Pseudo-Label Accuracy Comparison

In the following results, the analysis is based around varying something called the "sampling rate". Following [11], this simply involves sampling the most confident $x\%$ from each pseudo-labelled class in the target train data. This is used by the authors in order to assess whether the confidence scores do indeed correlate with accuracy. Furthermore, it is also utilised as a method to exclude samples which are assumed to be more noisy, and therefore, likely to harm the adaptation process.

It should also be noted that a single run of the cleanAdapt [11] pseudo-label generation process (described in section 3.3) is used to compare against all those the new methods. This method is of course random, so can fluctuate slightly. However, it is sufficient for this work to take one run, as the area of interest is to see whether any methods which improve upon the results also improve upon adaptation. The labels from this one cleanAdapt pseudo-label generation are

used in the adaptation of section 3.6.2.

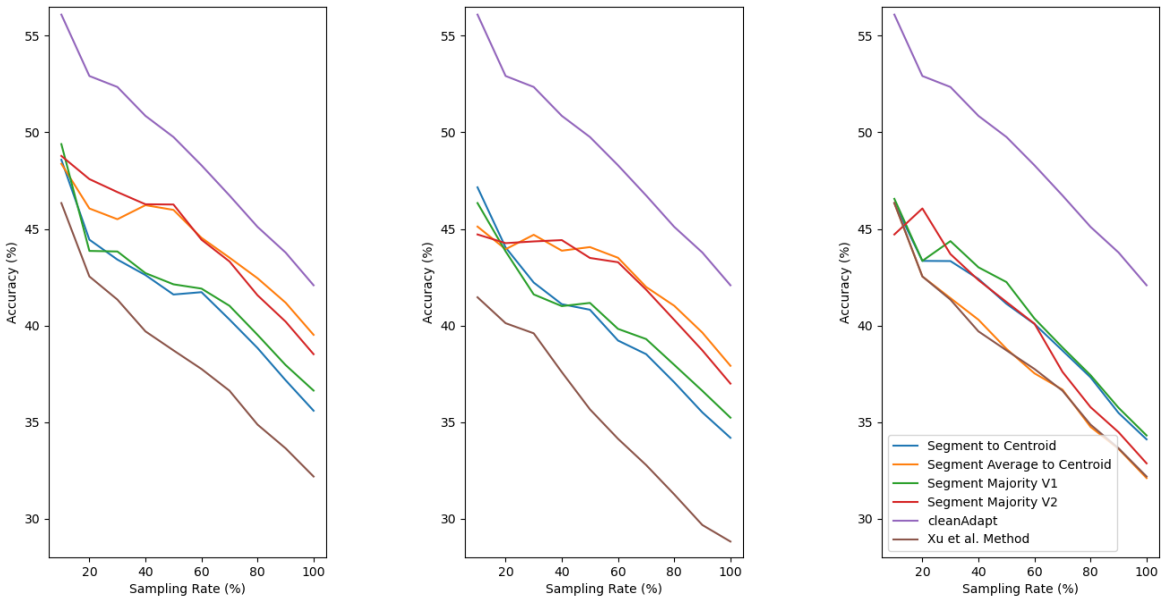
Source Classifier Methods The initial pseudo-label accuracy of the three variations discussed in section 3.4.1 can be seen in figure 3.7. Also included within this figure is the initial pseudo-label accuracy achieved for the random run of the re-implementation of the cleanAdapt method.

As expected, directing the network to regions of the videos which appeared to be more confident and discriminative than others outperforms the random sampling of segments from cleanAdapt. The most successful are the "Max Segment" (which chooses the most confident windows from each modality), and "Segment Majority" (which chooses the most confident windows from those windows which were pseudo-labelled as the most popular class prediction for the video) methods. It is clear that these two methods are in fact quite similar and capable. However, this does not mean that they select the same samples. On average across sampling rates, 6.05% of the "Segment Majority" method's correct pseudo-labels are different to the correct ones sampled by "Max. Segment". Both "Max. Segment" and "Segment Majority" methods perform markedly better than cleanAdapt, with average increases over all sampling rates of 7.35% and 7.29%, respectively. Furthermore, the confidence measurements of both exhibit the desired behaviour of increased pseudo-label accuracy when sampling the most confident samples, just like cleanAdapt.

The method labelled "Mean Segment", which averages the logits before determining the pseudo-label and confidence, performs reasonably similar for the larger sampling rates (60%-100%). However, for smaller sampling rates, the confidence scores clearly do not reflect the actual confidence of samples. The accuracy of this method's pseudo-labels do not increase as quickly as the other two methods and furthermore, actually perform worse than the random sampling of cleanAdapt when reviewing the most confident 10% of samples. This shows that although the average representation is on the whole better than simply random sampling, it does not reflect the confidence of pseudo-label accuracy as well as other methods.

Target Centroid Methods Although none of the methods introduced in section 3.4.2 are able to improve upon the initial pseudo-label accuracy of the cleanAdapt baseline, the results are nonetheless discussed here. This is because it is important to compare them against one another and to further compare them against the re-implemented sections of [58], which acts as an inspiration for the work.

Figure 3.8 shows the results for 3 ablation scenarios on the target-centroid based methods introduced in section 3.4.2. Sub-figure 3.8a compares the initial pseudo-labelling accuracy of the cleanAdapt re-implementation with all the new methods when the single-stage centroid construction of equation 3.9 is used. Alongside this can also be seen the re-implementation of [58] (labelled "Xu et al. Method") with its original single-stage centroid construction of equation 3.8. This can be compared with sub-figure 3.8b which shows those same methods using the two-stage centroid construction, described in section 3.4.2, similar to the original intention of [58]. It is clear from these two graphs that the second stage centroid refinement actually harms the

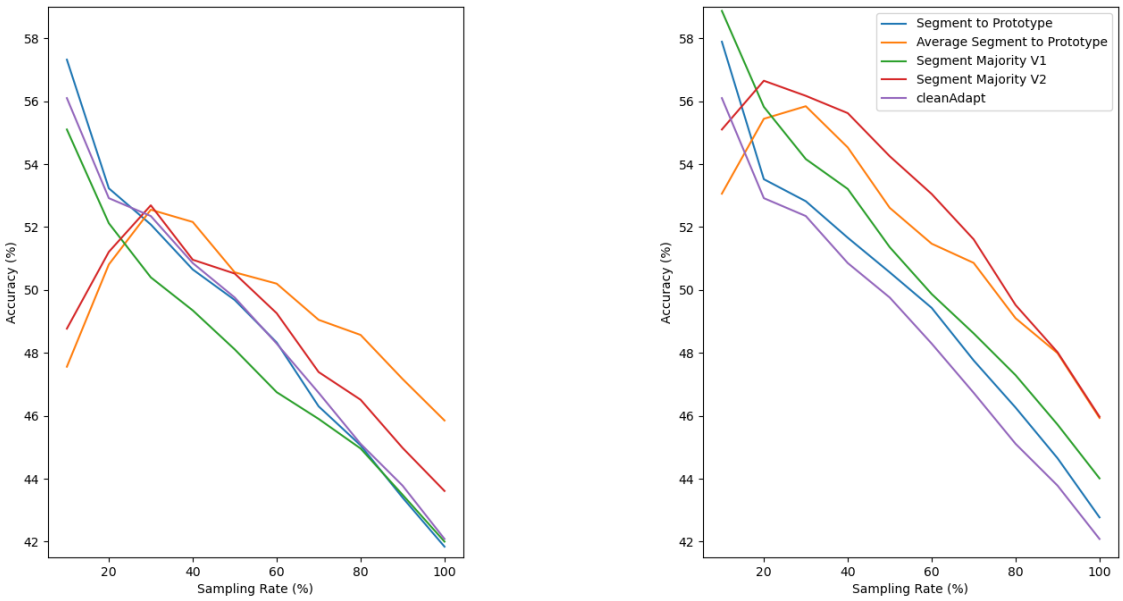


(a) One-Stage Centroids (b) Two-Stage Centroids (c) One-Stage [58] Centroids

Figure 3.8: Initial pseudo-labelling accuracy of target centroid methods, compared with the re-implementation of methods from cleanAdapt [11] and Xu et al. [58].

pseudo-labelling results for the EPIC-Kitchens-55 dataset. By first calculating the difference in accuracy at every sampling rate, for each of the pseudo-labelling methods (excluding cleanAdapt of course), when using one-stage and two-stage centroids, this can then be averaged to show that the second stage refinement of centroids causes a 2.04% decrease on average across all of the methods. However, the drop in performance is felt greater in the smaller sampling rates, where an average performance drop of 3.33% is seen for the 10% sampling rates, compared with just 1.86% for the 100% sampling rate. Of particular interest is that this performance drop is seen in the re-implementation of the pseudo-label method from [58]. This paper particularly chose the two-stage centroid construction, so it is strange that it performs worse than the single-stage for EPIC-Kitchens-55. There is little detail in [58] on ablations for this aspect and seeing as they do not test on the EPIC-Kitchens-55 dataset, it is hard to determine whether this is unique to the particular dataset used in this chapter. Furthermore, as discussed in the method section, the features used here are segment-based (from an I3D network), whilst the original method utilised a combination of a 2D-CNN and TRN-style [62] module to produce local-level features. This difference could also contribute to a disparity in the behaviour of the re-implemented method.

All of the methods described in section 3.4.2 (which use the modified centroid formulation) improve considerably upon the results of the re-implemented pseudo-label section of [58]. In particular, those methods which prioritise aggregating segment feature distances perform the



(a) To Prototypes

(b) To Source Segments

Figure 3.9: Initial pseudo-labelling accuracy of feature space methods, compared with the re-implementation of methods from cleanAdapt [11].

best, compared with those which rely on the closest individual segments. This is likely due to the fact that individual segments could be misclassified and then shift the classification in the wrong direction. It is possible that aggregation is able to suppress these negative variations better.

To conclude the results of the target centroid methods, it is important to determine how effective the new centroid formulation is. Sub-figure 3.8c shows the same pseudo-labelling methods from 3.4.2 but with centroids constructed using the single-stage equation 3.8, taken from [58]. It can be seen that all of the new methods introduced in this chapter compete well with the original method (labelled "Xu et al. Method" in 3.8c), where most outperform on the majority of sampling rates. However, all of the methods experience a considerable drop compared to when using the new formulation of centroids, outlined in equation 3.9 and reported in sub-figure 3.8a. This supports the notion that the new centroids are more informative and helpful for pseudo-labelling on EPIC-Kitchens-55 when using multiple segment features.

Joint Domain Feature-Space Methods Figure 3.9 shows the results for the methods described in section 3.4.3. The graph in sub-figure 3.9a shows results when the pseudo-label and confidence are both assigned based on distance to the source class prototypes. On the other hand, the graph in sub-figure 3.9b shows how these results differ when the pseudo-label is assigned based on the closest source segment feature. The confidence of these pseudo-labels are then

computed to be the distance to the corresponding source class prototype (similar to [31]). To begin with, it is clear once the labelled source segment features are introduced into the method that a substantial increase in accuracy is witnessed when compared to the prior target centroid results. All of the methods, for each ablation scenario, seen in figure 3.9 either produce results relatively similar to the cleanAdapt re-implementation or, in some cases, outperform them. To be specific, the "Segment to Prototype" and "Segment Majority V1" methods are the clear winners of the four methods introduced in section 3.4.3. They both compete well with cleanAdapt, although "Segment Majority V1" does slightly under-perform cleanAdapt when utilising prototypes only. Furthermore, both methods showcase that their confidence scores are accurate, as the accuracy of pseudo-labels increases as less confident samples are filtered out. Although both methods struggle to outperform cleanAdapt when using class prototypes for both pseudo-labelling and confidence, a consistent increase for all sampling rates is achieved when the closest source segment features are utilised for pseudo-labelling (refer to 3.9b). However, when compared against the results of the source classifier methods (introduced in section 3.4.1), they do not reach the same height of accuracy. This supports the idea that although the feature vectors are rich with information, a simple distance metric is not as informative as the learned source-only classifier in the case of this work.

Lastly, it should be mentioned that the two methods which aggregate the features of the video in some way ("Average Segment to Prototype" and "Segment Majority V2") are actually the most accurate when compared with the other methods and cleanAdapt overall (figure 3.9b). However, as the sampling rate decreases, a similar issue to that seen in some of the source classifier methods emerges. It can be seen from both ablation figures that the two aforementioned methods decrease quite considerably in accuracy once the sampling rate is below 30%. This means that despite being slightly more accurate, they produce confidence measurements that are inaccurate. Furthermore, despite their increase at higher sampling rates, they still cannot perform better than the best source classifier methods described prior.

3.6.2 Adaptation Comparisons

Only the two best methods are taken forward for comparison against cleanAdapt. These are the "Max. Segment" and "Segment Majority" pseudo-labelling methods from section 3.4.1. Furthermore, due to computational constraints, only one pseudo-label sampling rate is explored (20%) on the "D1->D2" (P08 as source and P01 as target) setting.

The adaptation results for the two chosen methods and the cleanAdapt re-implementation can be seen in figure 3.10. The results stop at epoch 15 rather than the 60 epoch mark outlined in [11] because it became clear that neither of the new methods could adapt successfully and were furthermore damaging the source gained knowledge. It therefore made sense to halt training at this point for the analysis as the training time for the two methods is roughly 5 times as long as cleanAdapt, when running on an 11GB GTX 1080 Ti GPU.

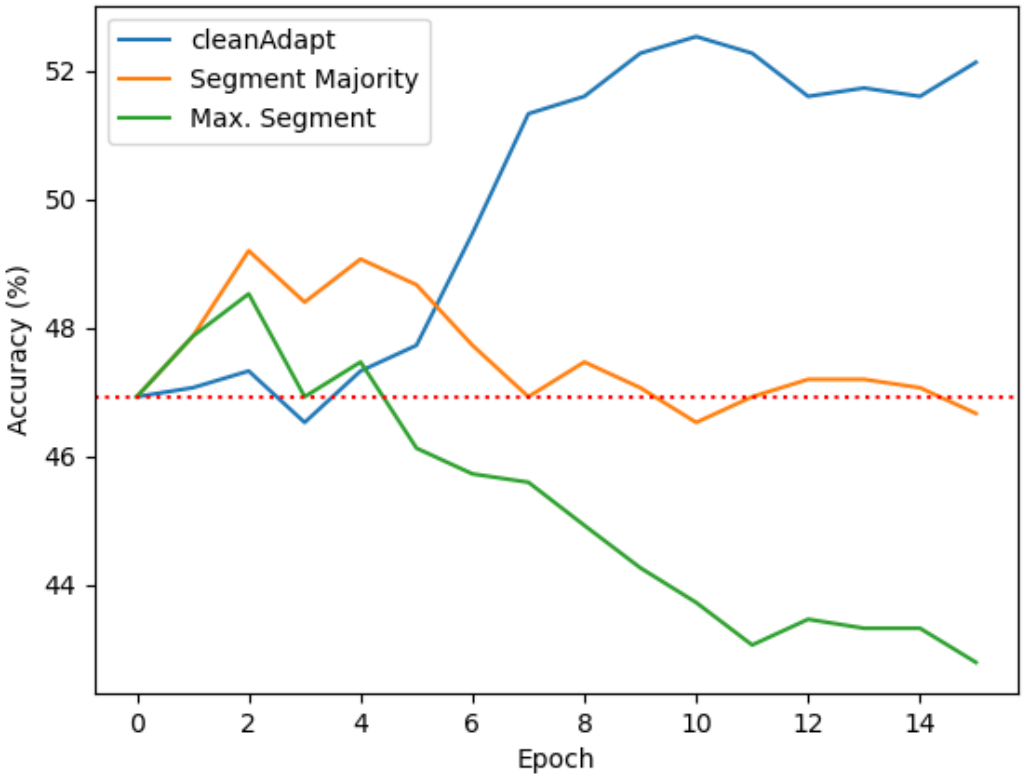


Figure 3.10: Accuracy on target test split after 15 epochs of adaptation with pseudo-labels from cleanAdapt, "Max. Segment" and "Segment Majority" methods.

Method	RGB	Flow
cleanAdapt [11]	43.90	43.75
Max. Segment	22.74	23.09
Segment Majority	22.87	23.11

Table 3.2: Average percentage of frames seen per video throughout 15 epochs of adaptation for each pseudo-labelling method.

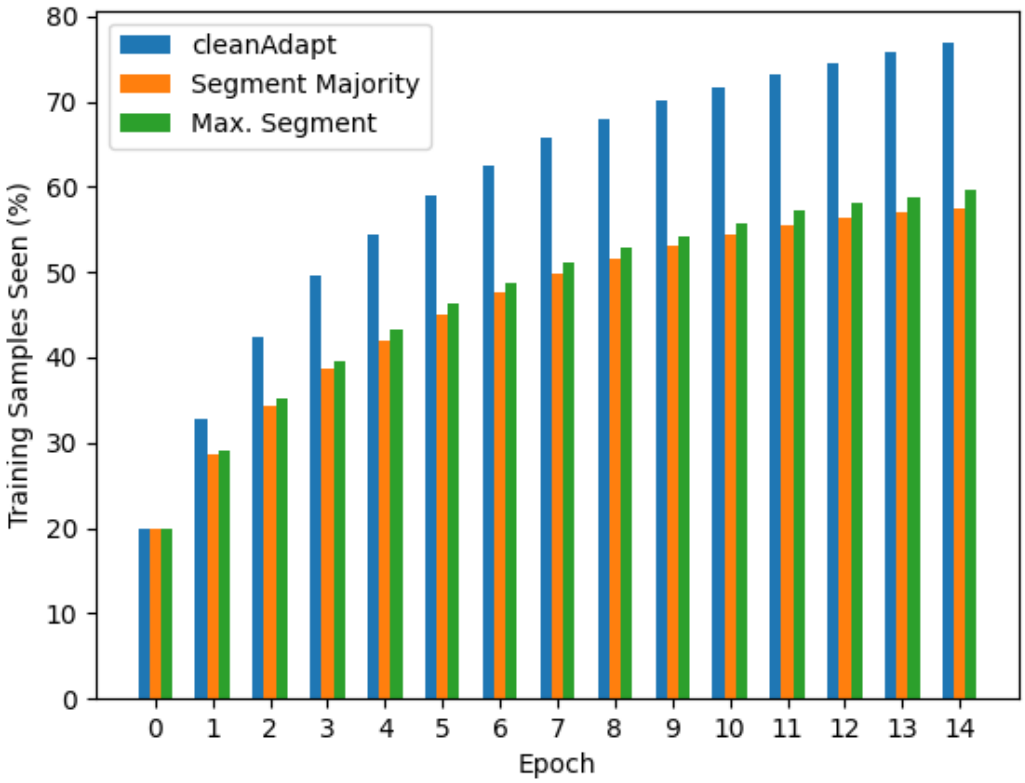


Figure 3.11: The percentage of training split samples seen by each model over the course of 15 epochs of training.

From the results it is clear that neither of the two new methods come close to adapting to the target domain as successfully as cleanAdapt does. Although both new methods show initial adaptation strength, reaching peaks on the second epoch, this is quickly reversed, resulting in degraded performance for later epochs. The exact reasons for this are difficult to determine empirically, however three key differences between cleanAdapt and the new methods can offer some explanation. Firstly, as the cleanAdapt re-implementation is random in its selection of video segments, it means that the samples seen across epochs grows rapidly. Contrary to this, both "Max. Segment" and "Segment Majority" direct the network to a much narrower range of samples throughout time. Figure 3.11 shows the comparison of pseudo-label rotation (more simply put, the amount of new video samples seen each epoch) for each of the methods. As expected, the cleanAdapt method increases at a much faster rate than the two new methods, reaching a peak of 76.83% by the final epoch. This is in stark contrast to "Max. Segment" and "Segment Majority" which both fall considerably short of cleanAdapt's rotation and only reach 59.56% and 57.56% coverage of the training set by the end of the 15 epochs, respectively. It is possible that this stagnation causes the networks to quickly overfit to similar samples. Secondly,

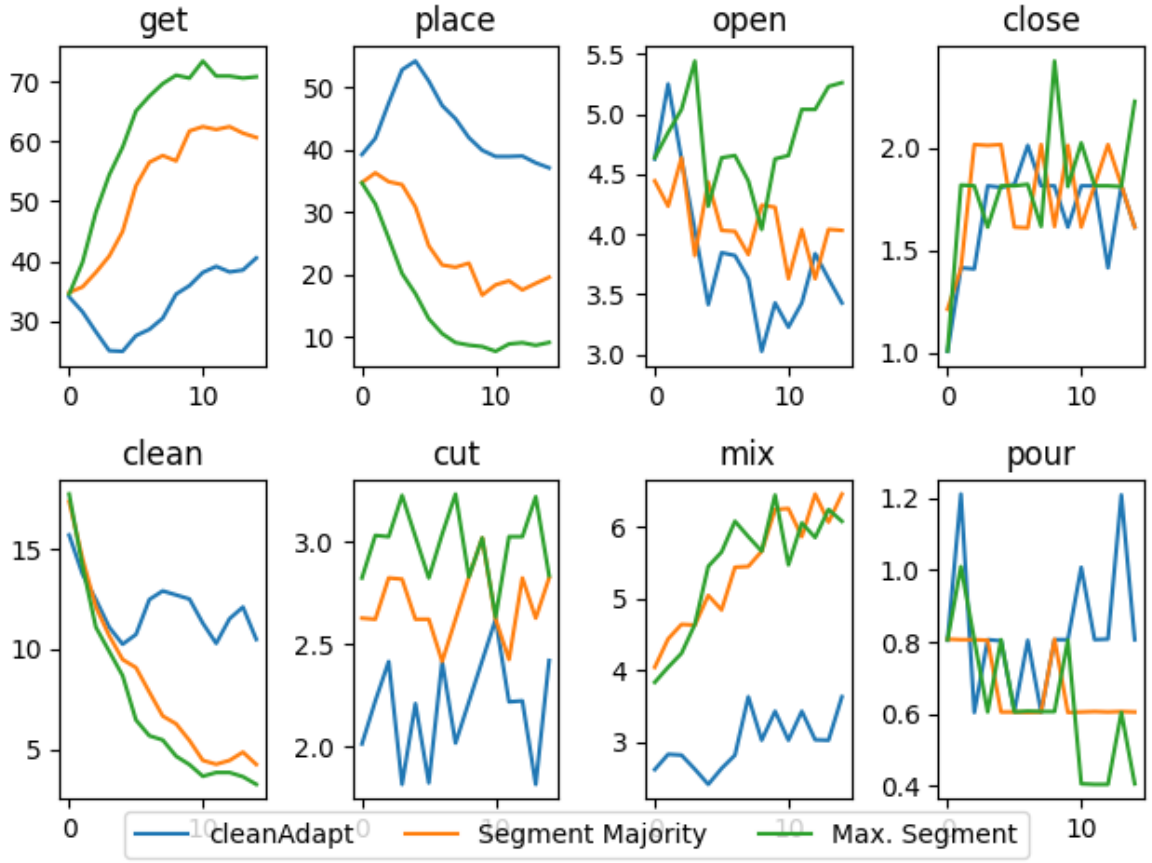


Figure 3.12: The percentage contribution of each class to the pseudo-labels used in training for each of the 15 epochs. The y-axis is the percentage contribution. The x-axis is the epoch number.

even though the two new methods see the same samples much more often than cleanAdapt, they do not, on average, ingest the same range of the video as cleanAdapt does. Table 3.2 records the average percentage of frames for each video that are used in the segments seen by the models across the 15 epochs of training. On average, the cleanAdapt model utilises 44% of the frames in each video. This is almost double the amount of frames which the "Max. Segment" and "Segment Majority" models use, averaging roughly only 23% coverage of frames between them. It is again possible that this narrow range of samples leads the new models to overfit due to lack of diversity. Lastly, the "Max. Segment" and "Segment Majority" methods both become dominated by the largest class "get" (refer to 3.13). Figure 3.12 shows, for each method, the percentage of target pseudo-labels at each epoch which are predicted for each of the eight verb classes. Although cleanAdapt also shows some bias towards certain classes, it is much smaller than what can be seen for the two new methods. By the end of the 15^{th} epoch, 70.85% and 60.69% of the pseudo-labels in the "Max. Segment" and "Segment Majority" methods, respectively, are predicted for the largest class "get". Figure 3.13 shows that just under 30% of the source and target domains are labelled for the "get" class, meaning the pseudo-label accuracy for the new

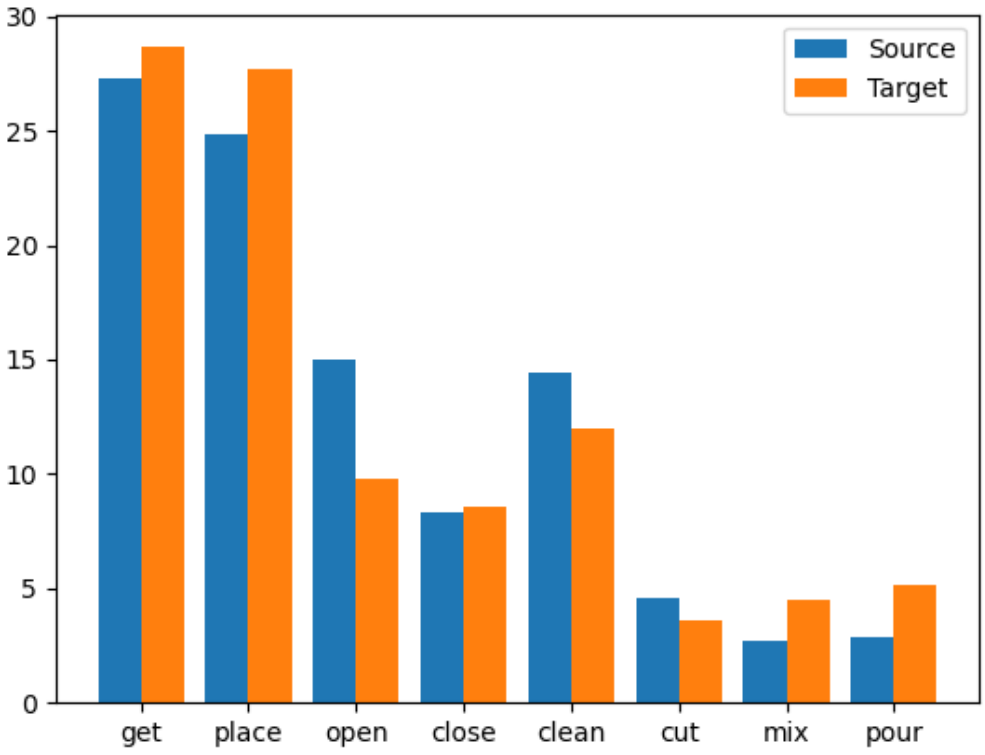


Figure 3.13: The percentage contribution of each class for the source (P08) and target (P01) domains. The x-axis shows the class names and the y-axis shows the percentage contribution to the total number of samples.

methods must be dropping as the method continues to assign more to the single class. Therefore, the bias towards the one class not only narrows the scope of the model's knowledge but it also proliferates incorrect pseudo-labels as training continues. This is in contrast to cleanAdapt which has a much better balance between classes, whereby "place" only ever makes up just over 50% of labels before dropping to accommodate more samples from "get". However, it should be noted that cleanAdapt also seems to bias towards the largest classes, it just does not target a single class like the new methods presented in this chapter.

Although both "Max. Segment" and "Segment Majority" methods degrade source-only obtained knowledge, it is clear from figure 3.10 that they do behave slightly differently. One potential reason for this can also be found in the visualisations of figure 3.12. As discussed previously, both of the new methods bias heavily towards pseudo-labelling most target samples as a single class ("get"). However, it can be clearly seen from the same figure that the "Max. Segment" method biases towards the largest class at a much faster rate than the "Segment Majority" method does. This results in "Segment Majority" only reaching 60.69% of pseudo-labels assigned to the largest

class by the final epoch, which is 10.61% lower than that of "Max. Segment". It is possible that this dampening of the class bias in the pseudo-labels causes the method to lose accuracy at a slower rate than "Max. Segment". Furthermore, the reason for the reduced bias in the "Segment Majority" method towards the largest class could be that it relies on sampling its segments from clips of prediction consensus. This could potentially stop the system from relying too heavily on single, overbearing segments that may have been outliers, and that can fool the "Max. Segment" approach. Furthermore, as "Max. Segment" picks up class "get" samples at a faster rate, the majority of the additional samples accepted into the pseudo-labelling sample (when compared to "Segment Majority") are actually incorrect. Most likely, this increases the descent of adaptation accuracy for this method compared to "Segment Majority".

From the results reviewed in this section, it is clear that simple pseudo-label accuracy is not a good indicator on how effective those same pseudo-labels are in the adaptation process. Many more factors could have a deciding factor on the process of domain adaptation with pseudo-labels. Discussed are the potential issues of pseudo-label rotation, video coverage and class bias. However, there could be many other reasons that contributed to the disparity between [11] and the new methods introduced in section 3.4.1. It is likely that the success of [11] relied on the random sampling approach of video segments, to avoid bias, as much as it did on cleaning the pseudo-labels using confidence score filters. Clearly further work is needed in order to tackle this problem.

3.7 Conclusion

This chapter explores new pseudo-labelling methods which utilise a wider range of the available video frames when compared with the random sampling approach of [11]. To begin with, the variations in predictions across different video clips of the same sample are explored to identify a potential source of refined information that is sometimes lost by the randomly sampled clip methods. Moving forward, several different pseudo-labelling methods are described and explored in relation to the initial pseudo-labelling accuracy. A few are able to perform markedly better than a random sampling approach. Namely, the "Max. Segment" and "Segment Majority" methods from section 3.4.1 achieve an average increase over cleanAdapt of 7.35% and 7.29% across all ten sampling rates, respectively.

Although two new methods are identified which greatly improve the initial pseudo-labelling accuracy of the cleanAdapt re-implementation, this does not translate into an improvement in the final adaptation training. Both new methods, although starting with more accurate labels, greatly underperform the cleanAdapt re-implementation. Furthermore, both actually harm the knowledge obtained through the source-only training by resulting in a lower accuracy after adaptation than before. Three potential reasons are discussed which show how the methods differ from cleanAdapt. Specifically, both methods seem to bias heavily towards both the largest class

in the training set and the same segments within videos. This is in contrast to cleanAdapt, which balances better between classes and rotates samples more often due to the random sampling approach.

From the work described, it is clear that although less noise in pseudo-labels is of course helpful, it is not the sole factor in improving adaptation performance. Further work is clearly required to confirm the exact causes of better adaptation with pseudo-labels.

CHAPTER 4

EPIC-KITCHENS-100 UDA CHALLENGE

The EPIC-Kitchens-100 unsupervised domain adaptation challenge for action recognition was started in 2021 and has been run by the creators of the dataset every year since. The aim of the challenge is to adapt a model to the target domain of the EPIC-Kitchens-100 dataset using the labelled source and unlabelled target data. Each entry simply has the goal of maximising the target action accuracy and therefore can use whichever method (or combination of methods) they see fit. This chapter offers an explanation of the challenge and dataset, as well as an analysis of the submission made to the 2023 challenge, in which pseudo-labelling was evaluated for its effectiveness on this difficult domain adaptation setting.

Specifically, two pseudo labelling methods, based around the recent work of [11], are evaluated and compared on the target validation and test sets. Similar to [11], the first method takes the source model's classifier and uses it to assign a pseudo-label and confidence score to each of the target samples. A selection of the most confident pseudo-labels are then used in the adaptation process. For the second method, a number of classes are chosen based on whether they are well-represented in the source data. The most discriminative target samples for these classes are then chosen based on the knowledge of the source-trained classifier and assigned a corresponding pseudo-label. More details on the two methodologies can be found in this chapter, along with a detailed analysis of the results on the EPIC-Kitchens-100 dataset. Furthermore, ablations of different sampling rates, feature types and ensembling methods are described as well as experiments that help support the link between poor adaptation with pseudo labels and poor similarity between the class distributions of sampled pseudo-labels and the test set.

	EPIC-Kitchens-55			EPIC-Kitchens-100	
Split	D1	D2	D3	Source	Target
Train	1542	2495	3897	16115	26115
Test	435	750	974	4298	5909

Table 4.1: Comparison of dataset sizes. Where "EPIC-Kitchens-55" represents the UDA variation of the dataset used in chapter 3.

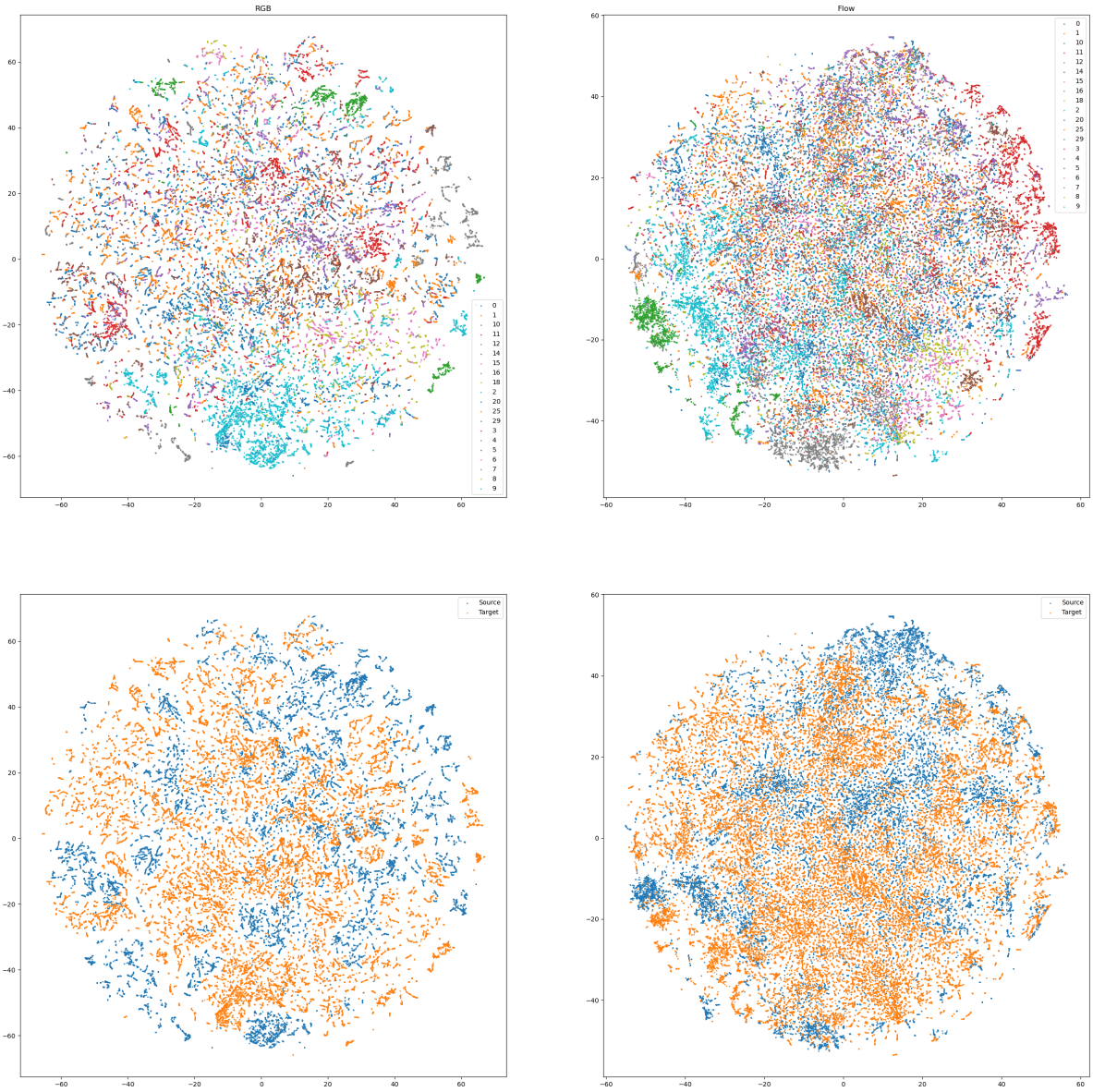


Figure 4.1: tSNE plot of the largest 20 verb classes in target validation (only 25% of all samples in each class are shown for visualisation purposes). **Left column** are RGB features. **Right column** are optical flow features. **Top row** is coloured by class. **Bottom row** is coloured by domain.

4.1 EPIC-KITCHENS-100 Dataset

The challenge described in this chapter centres around the EPIC-Kitchens-100 dataset. It was first introduced in [10] as an extension to the EPIC-Kitchens-55 dataset [9], an unsupervsised domain adaptation variation of which is utilised in chapter 3. Consisting of 100 hours of footage, it requires the classification of 97 verbs, 300 nouns and 4053 action classes (the number of possible actions is much larger but most are nonsensical and never performed by the participants). The dataset includes all the data present in EPIC-Kitchens-55, released in 2018, as well as new footage by the same participants, filmed two year later and released in 2020. Furthermore, specific to the unsupervised domain adaptation, those videos from 2018 constitute the labelled source domain, whilst the videos acquired during 2020 are utilised as unlabelled samples in a new target domain.

There are a number of reasons that make classification on this extended dataset much more difficult than the EPIC-Kitchens-55 variation [30], outlined in the chapter 3, section 3.1. Firstly, the new dataset and setup are part of a multi-domain setting. Unlike EPIC-Kitchens-55, which uses a single kitchen for source and a single kitchen for target, EPIC-Kitchens-100 utilises all participants in each domain, separated by a shift in time. Secondly, whilst EPIC-Kitchens-55 deals with an environment based domain shift only, the challenge dataset must deal with a much more complicated setup. Not only does the time difference create a domain shift, but so does the fact that some participants remained in the same environment, whilst others moved to a new setting. Lastly, the size of the new dataset plays a factor as well. To begin with, EPIC-Kitchens-100 requires classification on a much wider range of classes, as well as on a multitude of different class types. This is compared to the EPIC-Kitchens-55 variation used in chapter 3 which only requires classification on 8 verb classes. An example of the increased level of complexity within the feature space can be seen in figure 4.1. Furthermore, the number of samples in this extended dataset greatly outweighs the amount used before. Table 4.1 shows that the source domain is roughly 4 times the size of the largest single domain in the EPIC-Kitchens-55 variation, whilst the target domain is nearly 7 times as large.

4.2 Method

As expected from the previous chapters on unsupervised domain adaptation, the method used in the submission consists of three stages: (i) pre-training on source, (ii) extraction of pseudo labels and (iii) adaptation. Stages (ii) and (iii) are repeated at the end of each epoch of training.

Sections 4.2.2 and 4.2.3 below outline two possible pseudo-label extraction methods used in stage (ii), whilst stage (iii) is explained in section 4.2.4

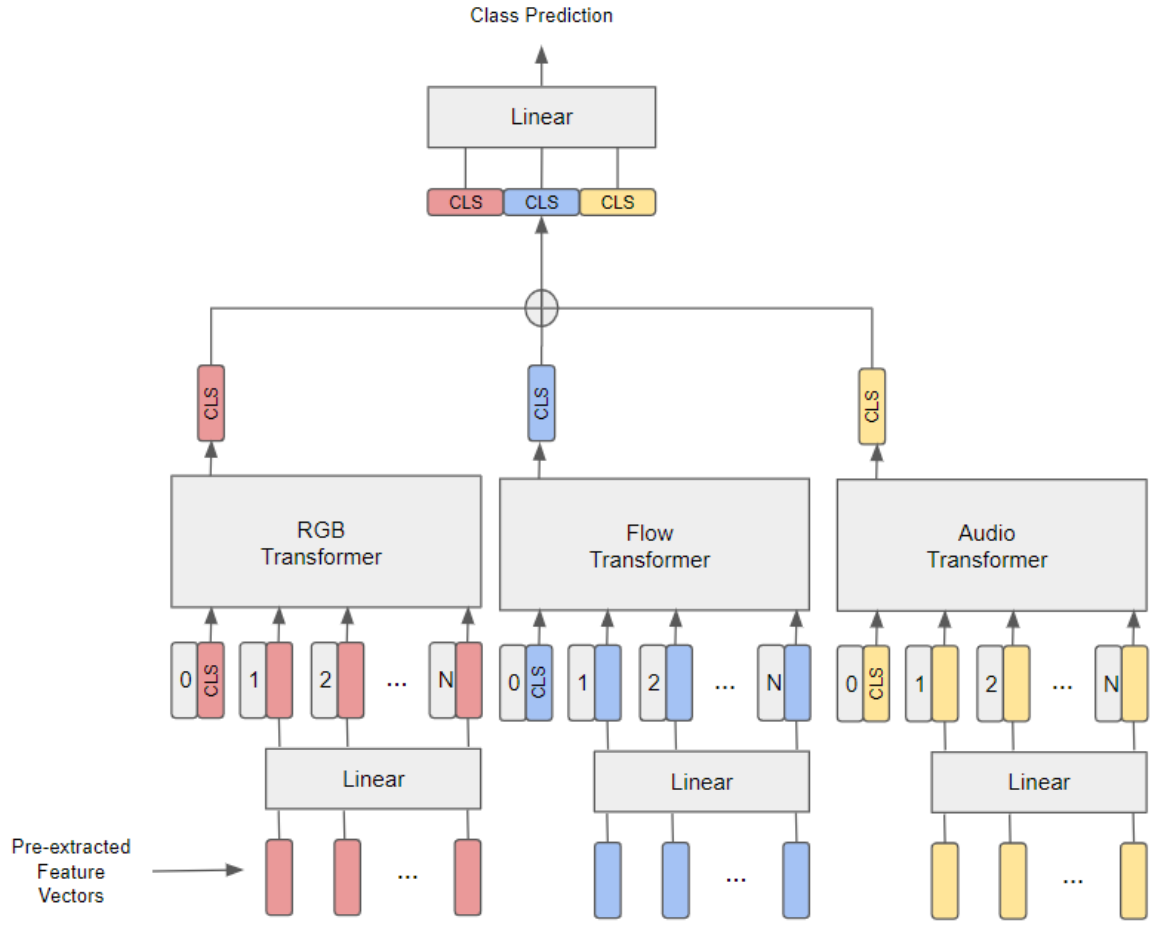


Figure 4.2: The transformer architecture used in the submission. This example depicts three modality streams, however when using TSM [26] features, the audio stream is simply omitted.

4.2.1 Pre-Training on Source

Most unsupervised domain adaptation works follow one of two approaches for adaptation training. One way is to train on source data only first, before using both source and target together for adaptation. The other is to use source and target data throughout the entire process. Due to the necessity to utilise source knowledge for pseudo-label extraction (otherwise the pseudo-labels would constitute unusable noise), this submission uses the former.

An overview of the architecture that is used in the source pre-training can be seen in figure 4.2. However, in order to explain the source training process more specifically, consider a single source train sample x. This sample corresponds to a ground-truth label y and a certain number of modalities k (e.g. RGB, Flow, Audio, etc.), depending on the features used. Each modality then consists of a sequence of frame-level features of length n. For example, the set of features for the source sample x in modality i would be:

$$x_i = (x_{i1}, x_{i2}, ..., x_{in})$$

For each of the k modalities, there is a corresponding encoding stream which transforms the frame-level features into a video-level representation that can be used for classification. Each of these modality streams consist of a projection layer and a transformer encoder, denoted by $l(\cdot)$ and $f(\cdot)$, respectively. Every frame-level feature in the modality i is first embedded via $l(x_{ij})$ and then added to a corresponding sinusoidal positional encoding PE_j, which is standard practice and notably described in [55]. All embedded frame-level features are then fed through $f(\cdot)$, simultaneously, alongside a learnable classification token CLS_i, following the methodology of [12], which is itself also supplemented with a positional encoding PE_{n+1}. The classification tokens outputted by each of modality streams CLS_i^{out} are concatenated to form a final vector which is used for classification:

$$CLS_{all}^{out} = Concat(CLS_1^{out}, CLS_2^{out}, ..., CLS_k^{out}) \tag{4.1}$$

The resulting concatenated token CLS_{all}^{out} is inputted into the a final linear layer $g(\cdot)$ in order to produce a set of predictions p:

$$p = g(CLS_{all}^{out}) \tag{4.2}$$

Standard cross-entropy loss is then employed in order to calculate the error rate of the network

$$\mathscr{L}_{CE} = -\sum_{c}^{C} y^c \log(p^c) \tag{4.3}$$

where y^c and p^c are the c^{th} elements of the one-hot encoding of the ground-truth label and the model predictions, respectively, whilst c represents each class in the training set.

Traditionally, the accuracy of the network would be evaluated on data from the source domain that has not been seen. However, due to the measure of success for the submission being the accuracy on the target domain, the source model during this training phase is assessed on the target validation dataset. This means that the source model is tuned according to the best adaptation process.

The following two subsections provide detail of the pseudo-labelling methods used in the submission to the challenge.

4.2.2 Top-Confidence Pseudo-Labelling

A motivation for the use of pseudo-labelling comes from the cleanAdapt methodology, introduced in [11]. In this work the authors point out the effectiveness of simply using the source-trained classifier to extract and evaluate the confidence of pseudo-labels for adaptation. Although this work focuses on the unsupervised domain adaptation variation of EPIC-Kitchens-55, introduced in [31], and uses random clips from the video samples, the main idea of using the source classifier is a major motivation for the first method attempted.

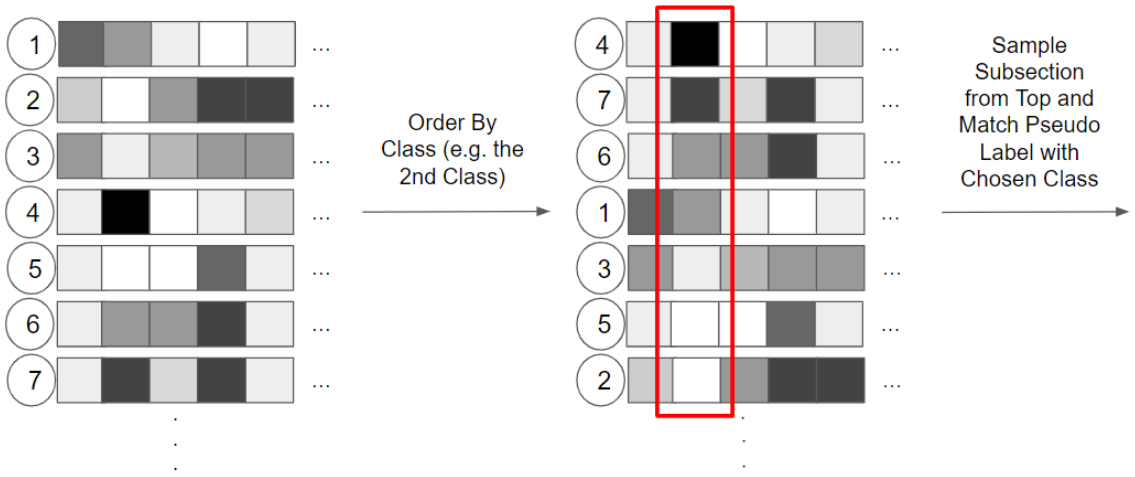


Figure 4.3: An example of Per-Class Confidence Pseudo-Labelling (outlined in section 4.2.3) for the second class. Each row of squares represents the class logits for a given sample, whereby the darker the square, the larger the logit value. The numbers to the left are simply a reference to help visualise the order change. The leftmost rows are initially ordered by which fed through the network first. The rightmost rows are re-ordered based on the class logit value for the particular class that pseudo-labels are being produced for. The red box symbolises those logits which are used in this particular example to order by. Out of these re-ordered logits, a certain percentage of the most confident are taken and given the specified pseudo-label (in this example, this would be the second).

To be specific, once the source-trained model is finalised, it is frozen so that pseudo-labelling can begin. A pseudo-label $\tilde{y}$ for each target sample is chosen using the predictions on that target sample by the source-trained model:

$$\tilde{y} = \arg\max_{c} p^c \tag{4.4}$$

As covered in chapter 3, pseudo-labels are often noisy due to the domain gap between source and target. In [11], they mitigate this with the assumption that the most confident pseudo-labels are most likely to be correct. Similar to that work, each pseudo-label is assigned a confidence score s using the cross-entropy loss

$$s = -\sum_{c=1}^{C} \tilde{y}^c_{one-hot} \log(p^c) \tag{4.5}$$

where $\tilde{y}^c_{one-hot}$ is the c^{th} element of the one-hot encoding of the pseudo-label $\tilde{y}$.

Unlike prior works, such as [11, 31], which sample an even amount from each pseudo-label class, simply the top $N\%$ most confident pseudo-labels are sampled. This is because even sampling of the classes causes worse performance than simply taking the most confident samples in the case of this submission, due to the extremely large number of classes available.

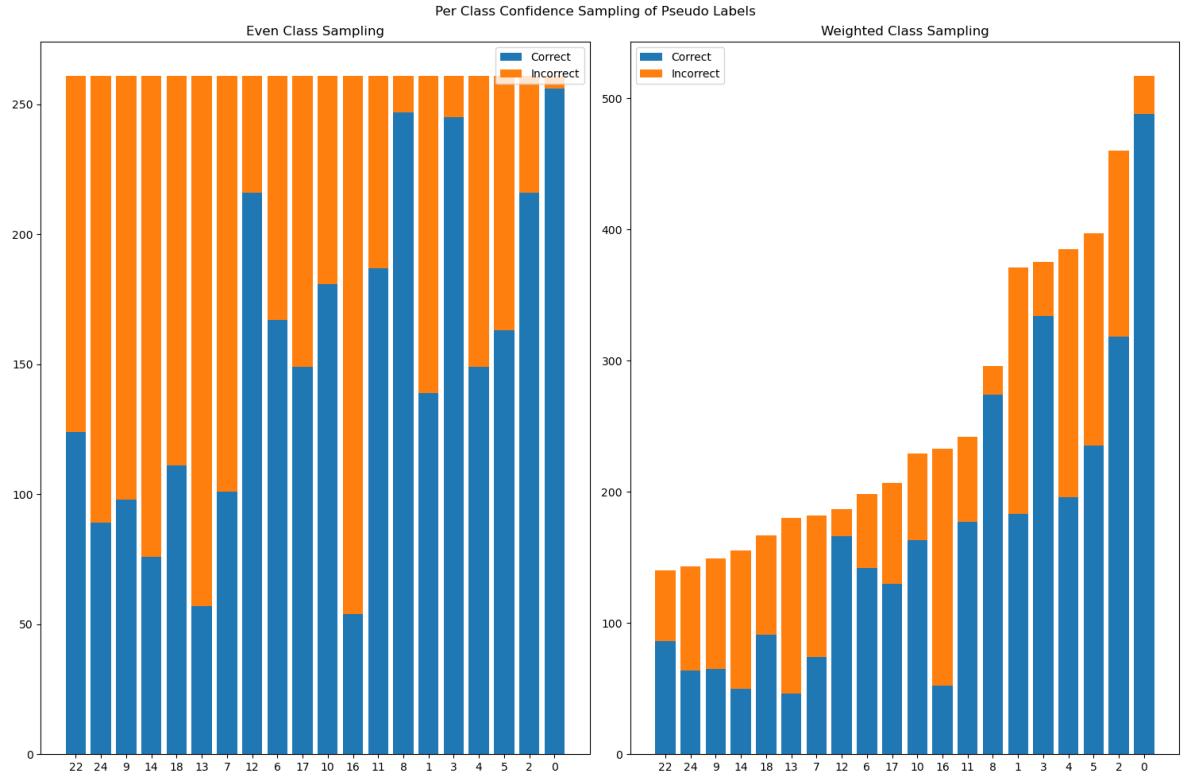


Figure 4.4: A comparison of target train pseudo-label accuracy when applying the per-class-confidence method with the top 20 source train classes. Left - accuracy when sampling equally for each of the classes in target. Right - accuracy when sampling for each target class based on the class' corresponding contribution to the source train top 20.

4.2.3 Per-Class-Confidence Pseudo-Labelling

Due to difficulties faced with the adaptation of nouns (which is analysed in more detail in the ablations section 4.4), a second pseudo-labelling method is explored and compared against the prior approach.

The idea behind the second method is to find samples for each class which best represent that class, according to the knowledge learned by the source model. This differs from the method in the previous section as that simply picks those samples which are most confident overall, regardless of their underlying class prediction.

A simple depiction of the per-class-confidence method is illustrated in figure 4.3 in order to visualise how the method works for an example. However, to explain it more formally, consider a specific class c. Every target sample is fed through the network to produce a model prediction p, as before. The confidence score of each sample for the given class c is computed by using equation 4.5 and setting the designated pseudo-label equal to the current class $\tilde{y} = c$. Once a class c confidence score is computed for each sample, all the samples are then ranked from most confident to least and a sub-sample of the top are taken and given the pseudo-label c. This process

is repeated for all of the C available classes so that samples for every class which the source model views as the most discriminative are picked.

It should be noted that due to the long-tailed nature of the source split (and for that matter, all of the splits), the source-only trained model fails to learn any knowledge of many of the tail classes. This is very likely then to carry over to the target data, as it would be very strange for the model to be able to classify samples from a class it had built no understanding of, especially when it is from another domain. For this reason, the decision is made to only pick samples which are most discriminative for those classes that the source-trained model has a reasonable knowledge of. However, in this setting, the target train labels cannot be used to determine which classes are recognised well by the source model. Therefore, this method is only performed for the M largest classes from the source train split. This is because source train labels are available and show which classes constitute the knowledge of the model. As expected, these classes are often from the head and hence the choice of using a variable number of the largest classes from source train when pseudo-labelling target train.

Selecting pseudo-labels for the M largest source classes is achieved by sampling N_c pseudo-labels for class c according to its percentage contribution to the source train top M classes:

$$N_c = N(\frac{m_c}{m}) \tag{4.6}$$

where N is the total number of pseudo-labels to sample (from all the classes), m is the number of samples in the top M largest source train classes and m_c the number of samples for class c.

The thought behind this contribution-based sampling is that smaller classes are more likely to have a larger portion of incorrect pseudo-labels. With the earlier assumption (that the most confident pseudo-labels are more likely to be correct) in mind, it is hypothesised that by sampling a lower amount of smaller classes, then those pseudo-labels more likely to be incorrect would be filtered out of the final selection. Figure 4.4 showcases the difference this weighted sampling has over simply sampling an even amount from all of the top M classes. As can be seen, the balanced sampling includes a lot of incorrect labels, whilst the sampling based on contribution does in fact increase the percentage of correct pseudo-labels for the smaller classes (the smallest 10 all saw increases) whilst also maintaining reasonable accuracy on the larger classes. Furthermore, the even sampling is in fact lower in accuracy at 57.95% compared with 63.96% for weighted class sampling.

4.2.4 Adaptation

Once the model is trained using the source data only and the pseudo-labels are assigned and filtered based on one of the previous two methods, the final stage of work is the adaptation itself.

The source model is unfrozen and is then trained conventionally and quite similarly to section 4.2.1. The only difference for adaptation is that the batch consists of half source train samples and half pseudo-labelled target samples. The process of assigning, filtering and adapting with

pseudo-labels is repeated after every epoch. The exact number of pseudo-labels used in each epoch depends on the sampling rate percentage used. The effect of varying this amount is discussed in the result section, 4.3.

4.3 Results

4.3.1 Implementation Details

Following the general trend of previous submissions to the challenge, the submission described employs ensembling for the final results. The details of experiments including ensembling can be found in section 4.4, however, it is important to mention them briefly here too, for the sake of implementation detail. In particular, two feature types are used in training the model: (i) TBN [22] features, provided by the challenge coordinators[1], which have three modalities (RGB, Flow and Audio), and (ii) TSM [26] features extracted using [37], which include two modalities (RGB and Flow). Each network utilises a single feature type and classifies on either verb or noun. After training, these are ensembled at inference, as discussed in sections 4.4.1.

Each individual network consists of k modality streams and one final classification layer. A modality stream is made up of a single linear layer without activation function, which reduces to 512 dimension, and a transformer with one encoder layer which has 4 self-attention heads, dropout with a probability of 0.1 and a hidden dimension of 2048. Networks which use the TBN features have $k = 3$ modality streams, whilst networks that employ the TSM features have $k = 2$. The final classification layer of the network is again a single linear layer without activation function which reduces to either 97 dimension or 300 dimension, depending on whether verbs or nouns are being classified, respectively.

Each of the transformers inside the modality streams receive either 25 or 8 tokens for TBN or TSM features respectively, plus a learnable CLS token, initialised randomly and appended to the sequence for classification.

For the source-only training, outlined in section 4.2.1, each network (regardless of TBN/TSM or verbs/nouns) is trained using the source train split with a batch size of 48, an Adam optimiser with weight decay of 1e-06 and learning rate of 3e-05 for 5 epochs, with a step of 10 after epoch 2 and 4. The model checkpoint from the epoch with the best target validation accuracy is then chosen.

For adaptation, as described in section 4.2.4, each network is trained for 10 epochs and the model checkpoint from the epoch with the best target validation accuracy is picked. Networks which classify nouns use a learning rate of 3e-07, whilst those that classify verbs use 3e-06. The same optimiser settings are used as described for source-only training and the same batch size. However, the batch in adaptation consists of 24 labelled source samples and 24 pseudo-labelled target samples.

[1]Available at https://github.com/epic-kitchens/C4-UDA-for-Action-Recognition

Split	Verb	Noun
Target Val.	0.48%	0.11%
Target Test	1.13%	-0.41%

Table 4.2: Improvement from source-only when ensembling the best adaptation models from section 4.4.3.

Top-1 Accuracy			Top-5 Accuracy		
Verb	Noun	Action	Verb	Noun	Action
54.07	32.73	23.96	81.55	58.27	41.72

Table 4.3: Final submission scores on target test set.

Lastly, it should be noted that action scores are normally calculated by combining the most confident verb and noun predictions. However, in this case all action scores are a result of implementing the verb-noun co-occurrence weighting, as introduced by the 1^{st} place winner of the 2022 challenge [6]. This is because it shows a consistent improvement of action scores without any need for additional learning.

4.3.2 Final Choice

For the work described in this chapter, there is a total of four model architectures when computing a final performance on the three classification metrics; two noun classification models and two verb classification models, each for one feature type (TBN or TSM). For this reason, it is important to pick the best combination of models according to the performance of the challenge's final target test set. As can be seen from Table 4.2, there is an adaptation increase in both verb and noun when selecting the best adaptation models, however, this does not translate into an increase for nouns in the final test set. Therefore, for the final score described in table 4.3, adapted verb models and source-only noun models are utilised as these provide the best result. The exact models chosen for each network are described in section 4.4.3 as part of the ablations.

4.4 Ablations

4.4.1 Ensemble Method Choice

Ensembling - whereby multiple models focused on the same task are combined in order to increase performance - is a common strategy in the EPIC-Kitchens-100 unsupervised domain adaptation challenge, being employed by nearly all of the recent entries [5, 6, 35, 36, 59, 61]. This is because it often offers a relatively simple way to boost performance without any need to alter training procedures.

Prior submissions to the challenge either ensemble different model architectures, if training end-to-end, or ensemble different sets of features, if training with pre-extracted features. The

Method	Verbs	Nouns	Actions
TBN Challenge Baseline*	46.32%	27.13%	18.70%
TBN-Only	49.90%	30.23%	21.30%
TSM-Only	48.47%	28.91%	19.09%
Training+Inference Ens.	52.48%	32.99%	22.94%
Inference-Only Ensemble	54.12%	33.37%	23.61%

Table 4.4: Target validation performance comparison of single and ensembled source-only models. *obtained by running the source-only model from https://github.com/jonmun/EPIC-KITCHENS-100_UDA_TA3N

Verbs	Nouns	Actions
+6.03%	+5.06%	+4.95%

Table 4.5: Increase over "EPIC_TA3N" adaptation baseline on target test when using the best source-only trained ensemble model.

submission detailed in this chapter focuses on training with pre-extracted features (as they are readily available and much quicker to train with) so it makes sense to explore the performance of ensembling with different feature types. The specific detail of these features types is explained in section 4.3.1.

Two kinds of feature ensembling are experimented with during the work for this submission, which can be seen in table 4.4. The first is to ensemble throughout the training process (named "Training+Inference Ens."). This involves combining the logits of the final classification layer of each feature type network, via addition, and then using it for comparison against a label. The second ensembling method is similar to the first, however, the combination of logits only occurrs during testing, once the networks have learned separately and have been frozen (named "Inference-Only Ensemble").

As can be seen from the results in table 4.4, both ensembling methods greatly outperform the single feature networks (labelled "TBN-Only" and "TSM-Only") to show the effectiveness of this procedure. Furthermore, inference-only ensembling has a clear advantage over the other method tested and for this reason it is chosen for the production of the final results.

4.4.2 Source-Only Performance

When pre-training on source, the target validation accuracy is an important metric for this work as it facilitates the selection of the best model. The source-only performance of the transformer architecture can be seen in table 4.4 alongside the results achieved by re-running the source-only model baseline provided by the challenge (labelled "TBN Challenge Baseline"). It is clear that the network architecture used in this submission has an advantage over that used in the baseline as it improves on all metrics. Furthermore, table 4.5 shows the performance increase over the "EPIC_TA3N" baseline (found at `https://codalab.lisn.upsaclay.fr/competitions/1241#`

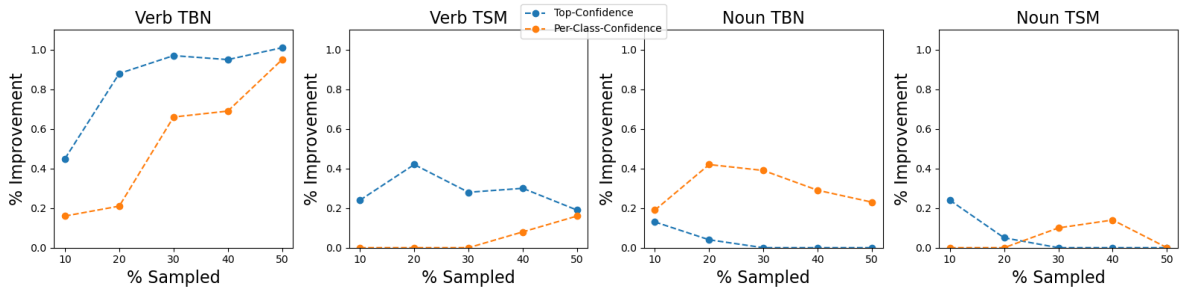


Figure 4.5: Target validation accuracy change on the best TBN and TSM networks after adaptation. Blue represents results for the top-confidence pseudo-labelling whilst orange represents results when employing per-class-confidence. A 0% improvement here simply means the best model after adaptation was still the source-only model.

`results`) when using the ensembled source-only models on the final target test split. Despite not seeing any target data at this point, the inference-only ensembling architecture is able to generalise better than the adaptation baseline provided by the challenge, which does use the target train data. In order to check that this increase is not purely from ensembling, the TBN network is compared against a re-run of the source-only TA^3N baseline provided by the challenge (see URL in caption of table 4.4), which also uses TBN features, on the target validation split. Table 4.4 shows that using TBN features only on the transformer architecture still outperforms the TA^3N [4] architecture of the source-only challenge baseline. This supports the suggestion that the architecture employed in this submission is superior to even the adaptation process employed in the original baseline.

4.4.3 Adaptation Methods

In this section, the two pseudo-labelling methods from sections 4.2.2 and 4.2.3 are compared using five different sampling rates, from 10% to 50%. These lower values are chosen as usage of all pseudo-labels (100% sampling rate) consistently hurt adaptation. However, future work could explore a wider range of sampling rates to determine if further improvement can be achieved.

The top 20 largest classes from source train dataset split are used for experiments on the per-class-confidence pseudo-labelling in this section. Discussion of the behaviour seen when varying this number and why it is not used in the challenge submission can be found later, in section 4.4.4.

Adaptation of Verbs. Figure 4.5 shows that taking a sample of the most confident pseudo-labels achieved an adaptation increase on the target validation dataset for both feature types. However, the best sampling rate for each does not correspond and the TSM verbs struggle to increase more than half of what TBN is capable of.

Per-class-confidence pseudo-labelling, when using the top 20 source train classes, performs

worse for both feature types. Although an accuracy increase is witnessed for TBN features, albeit less than the top-confidence method, TSM features cause a degradation from source-only performance when using the new method for the lower sampling rates. Furthermore, larger sampling rates offer little improvement, but do not decrease accuracy any lower than achieved with source-only training.

Therefore, the verb adaptation models that are chosen to evaluate against target test are those using the top-confidence method described in section 4.2.2 with sampling rates of 50% and 20% for TBN and TSM networks, respectively. Table 4.2 shows that when these networks are ensembled they achieve an adaptation increase of 0.48% over source-only on the target validation. Furthermore, this translates to an adaptation increase of 1.13% on the target test dataset used for the final submission.

Adaptation of Nouns. It should be noted that the adaptation of noun networks turned out to be a considerably more difficult task than that of the verb counterparts. Some thoughts on this possibly arising from the difference in the number of classes and class distributions are discussed later, in section 4.4.5.

Although the top-confidence pseudo-labelling method works quite well in adapting verb classification networks, this is not necessarily the case for noun related networks. For TBN noun adaptation, the top-confidence method causes a decrease in target accuracy after adaptation for most of the sampling rates. Moreover, the little increase seen on the lower sampling rates can not compete with that seen when using per-class-confidence with the top 20 source train classes, which consistently achieves an adaptation increase, although quite small compared to what is seen in verbs.

Similar to verb adaptation, the behaviour between different features types is not consistent. Adaptation of the TSM noun network using per-class-confidence pseudo-labelling, with the top 20 source train classes, causes a negative result for the three lowest sampling rates and only a minuscule increase for the last two. Top-confidence pseudo-labelling however, achieves the best adaptation increase for the TSM noun network when using the 10% sampling rate. In line with TBN nouns, this is a relatively small increase when compared to the verb adaptation.

Therefore, the noun adaptation models that are chosen to evaluate against the target test are per-class-confidence using the top 20 source train classes, with a sampling rate of 20%, for TBN and top-confidence pseudo-labelling with a sampling rate of 10% for TSM features. When ensembled, these networks do produce an adaptation increase of 0.11% over source-only ensembling, on the target validation split. However, this translates into a decrease in accuracy when applied to the target test set. It is for this reason that the final submission has to utilise the source-only trained noun networks, as they offer the best performance on the challenge metrics.

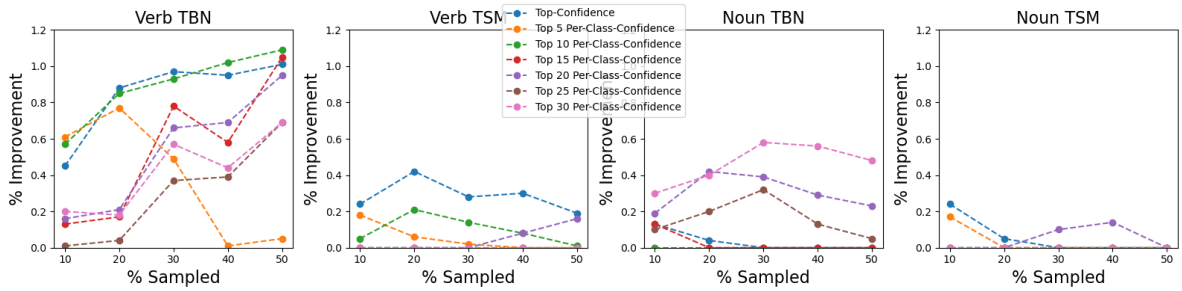


Figure 4.6: Updated adaptation scores when varying the number of top classes from source train for per-class-confidence pseudo-labelling.

Split	Verb	Noun
Target Val.	0.24%	0.20%
Target Test	0.72%	-0.10%

Table 4.6: Improvement from source-only when ensembling the best adaptation models from section 4.4.4.

4.4.4 Picking Top Classes for Per-Class-Confidence

Due to the time constraints of the challenge, the exploration into varying the number of largest source classes for per-class-confidence pseudo-labelling is not completed in time to be utilised in the final submission choice. However, the experiments are completed afterwards and the data can be seen in figure 4.6. It should be noted that per-class-confidence can actually perform as well as top-confidence for verb TBN networks when the number of source train classes is lowered to 10 or 15. Furthermore, when using the top 10 largest source train classes and a sampling rate of 50%, per-class-confidence pseudo-labelling is able to out-compete the top-confidence model picked in the previous section by 0.08%. This increase is likely due to the source-only verb model's limited learning of head classes. As can be seen from the upper verb section of figure 4.8, the most confident pseudo-labels (picked using the top-confidence method) represent in total only 12 classes, where two of which are only a few samples. It is therefore probable that by decreasing the number of classes and sampling more heavily from them that the per-class-confidence method accesses the learned head-class knowledge of the source-only model better than the previous choice which would have put more emphasis on lesser understood classes. Similarly, the noun TBN results can be increased by a maximum of 0.16% on target validation by simply increasing the number of source train classes used in per-class-confidence. This further supports the claim made for understanding the increase in verb per-class-confidence. As can be seen in the noun section of figure 4.8, the most confident noun pseudo-labels span a larger number of classes, therefore by expanding the number of classes in the per-class-confidence it makes sense that the adaptation would do better as the model can uncover knowledge that was previously hidden from it.

Experiments for TSM verb and noun networks show that slight increases are possible but nothing competes against the results achieved and analysed in the previous adaptation section. It is possible that further values for noun and verbs could have increased the results even more, but a range of values had to be chosen to experiment on with the time left available.

Table 4.6 shows the target validation and test performance after ensembling the best adaptation models from the extended experiments in this section. Comparing this to the results of table 4.2, it can be seen that the slight increase in the TBN verb model does not translate to an improvement once ensembled. In fact, it performs worse than simply using the best top-confidence models identified in section 4.4.3. The reason for this could be that the newly improved verb results are achieved by limiting the class labels further than the previous top-confidence choice. Although this improves on the validation set, it is likely that degrades generalisability to the test set which might have appeared from the slightly larger pseudo-label count of top-confidence. On the other hand, the enhanced TBN per-class-confidence adaptation accuracy for nouns does increase the ensemble accuracy slightly, by 0.09%. Despite the increase, this still results in a degradation of the source-only performance when applied to the target test split. However, the drop is reduced to only a quarter of the results in table 4.2, showing that an increase in noun adaptation on target validation does translate somewhat onto the target test split. This again seems to support the previous explanation as for nouns the new method choice increase the number of classes, therefore it would make sense that it is more generalisable to the test set which seems to be the case as it does not degrade performance as much as was previously seen in the more restricted per-class-confidence choice.

4.4.5 Test Class Distribution Matching

As briefly mentioned in prior sections, adapting the models that classify for nouns is found to be much more difficult than those which classify verbs. Although per-class-confidence pseudo-labelling does improve nouns, it still does not compete with the level of adaptation that is witnessed in verbs. This section discusses this problem and attempts to find a potential cause for this difference.

When top-confidence pseudo labelling adaptation is applied to the TBN noun network with a sampling rate of 20% , it can be seen that the degradation in accuracy is mostly caused by two classes, 9 and 7 (sponge and bowl, respectively). Figure 4.7 shows that these noun classes are highly important to the test set (which is target validation during training) as they are among the 5 largest classes. However, when the target training pseudo-labels are examined in figure 4.8, it is possible to see that both are very poorly represented in the adaptation training. Furthermore, the majority of those classes' pseudo-labels are incorrect.

In order to investigate the effect of this label distribution difference, experiments are performed using the target validation label distribution. It should be noted that this is not appropriate for an actual entry into the challenge as it assumes target labels. However, it is used here in

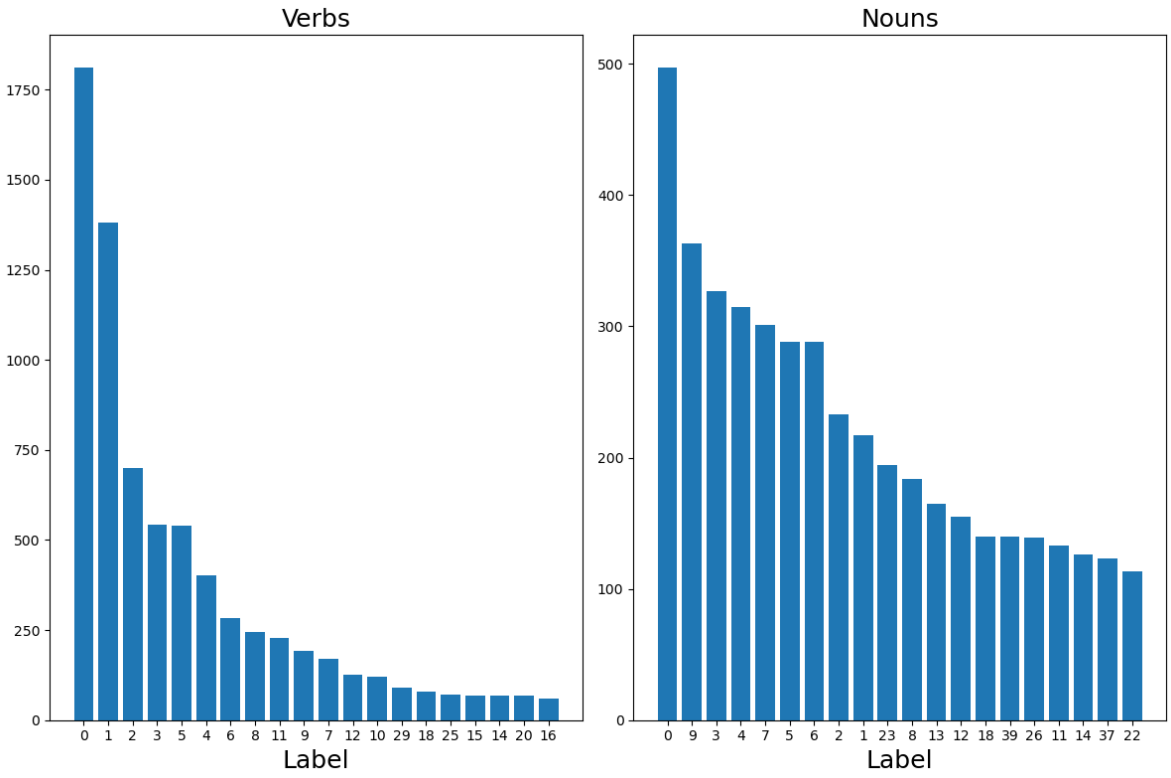


Figure 4.7: Distribution of top 20 classes in target validation dataset.

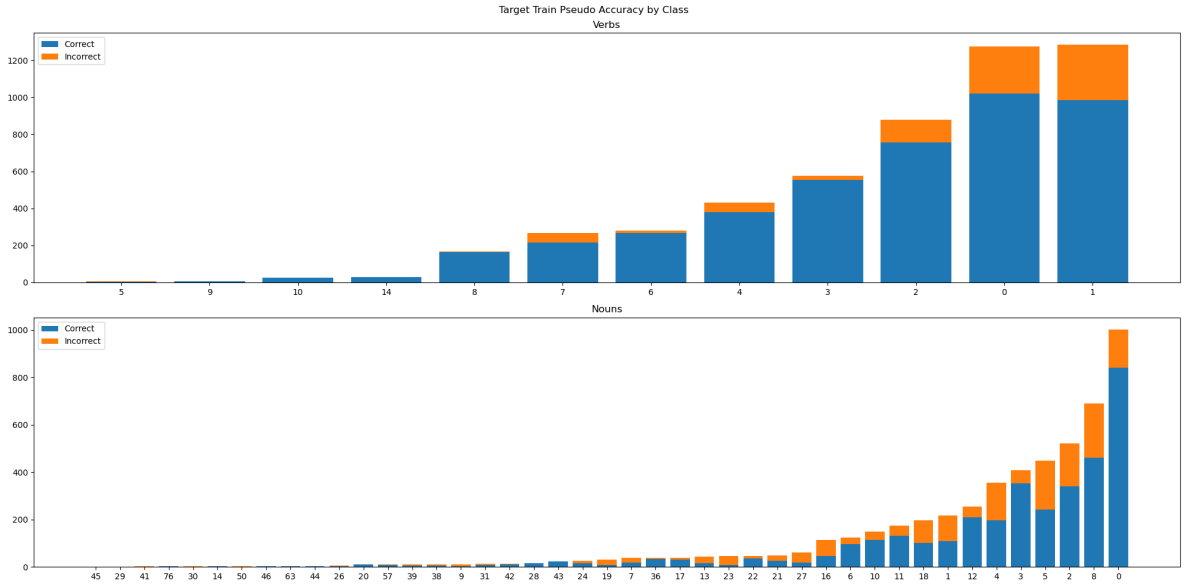


Figure 4.8: TBN target train pseudo-labels and their accuracy using the top-confidence method with a sampling rate of 20%.

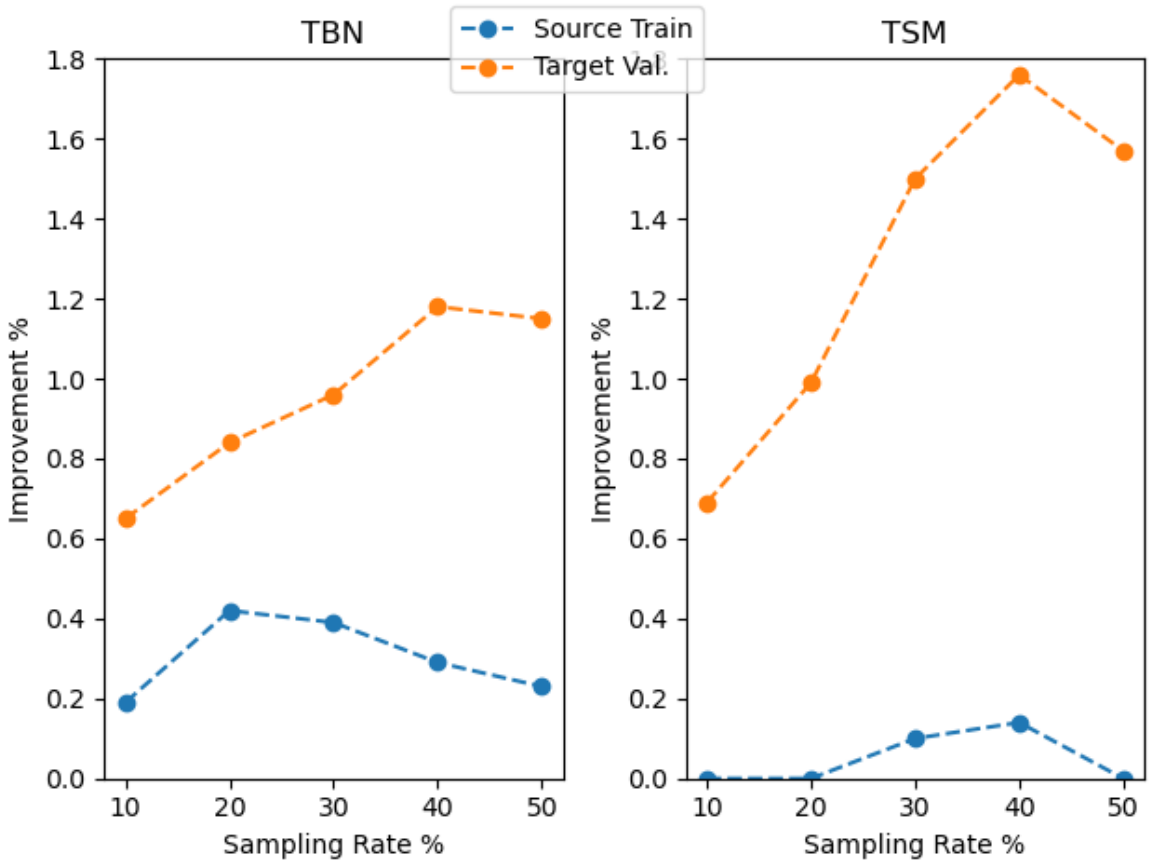


Figure 4.9: Target validation accuracy improvement using per-class-confidence pseudo-labelling on the best TBN and TSM noun networks after adaptation. Blue is using the top 20 source train classes. Orange is using the top 20 target validation classes.

order to support a claim about why nouns are much harder to adapt in this setting.

Figure 4.9 shows a comparison of per-class-confidence pseudo-labelling on TBN and TSM noun networks when sampling for pseudo-labels from the top 20 classes in source train and target validation datasets. Knowledge of the testing set (which is target validation in this scenario, as it is used for evaluating performance of the trained model) greatly increases results for every sampling rate and feature type. Even those that see degradation in TSM networks see a reasonable adaptation increase. Furthermore, with this knowledge, the TSM features are actually able to outperform the TBN feature adaptation, whilst both are able to compete with the adaptation seen in verbs when using top-confidence. These results help support the suggestion that if the pseudo-label class distribution does not match the class distribution of the test set very well, then it is likely to harm adaptation, rather than aid in it, in this setting of many classes.

If the distribution of verb classes in figures 4.7 and 4.8 are compared, it can be seen that they match the actual target validation distribution (the temporary test set) much better than the noun equivalents. With this knowledge and the results from the prior paragraph, it seems a

Split	Nouns
Source-Only	32.75%
Source Train	31.27%
Target Train	33.09%

Table 4.7: Target test noun accuracy when using the top 20 largest classes from different dataset splits for per-class-confidence pseudo-labelling.

likely explanation that the reason verbs are able to adapt with pseudo-labels better than nouns is because their pseudo-label distributions are already similar to the test set.

To investigate this hypothesis further, per-class-confidence pseudo-labelling adaptation is run for nouns using the top classes from the target train class distribution. The thought behind this is that target train is likely be more representative of the hidden target test set than source train is. Table 4.7 shows the results of ten epochs of per-class-confidence adaptation training, using the top 20 classes from the source and target train splits. As expected, knowledge of the target train distribution increases noun accuracy on the target test dataset, whilst source train causes a degradation.

Although this information cannot be used to improve results in the challenge, it does highlight the limitations of these pseudo-labelling methods in this setting in particular, where label distributions between source and target differ considerably.

4.5 Conclusion

This chapter explores a submission to the EPIC-Kitchens-100 unsupervised domain adaptation challenge for action recognition in 2023. Two different pseudo-labelling methodologies, one of which is similar to [11], are explored and compared when using a multi-modal transformer-based architecture.

For the submission, it is found that the network can achieve a 1.13% increase in verb adaptation on the target test split when using top-confidence pseudo-labelling and ensembling. However, difficulty in adapting noun networks means that a small increase of 0.11% in target validation dataset does not translate to an increase in the target test dataset. In fact, both top-confidence and per-class-confidence pseudo-labelling adaptation on noun networks cause degradation on the target test split. For this reason, the submission detailed above utilises source-only noun networks for the final entry.

Despite the set backs in noun adaptation, analysis shows that the difficulty in some adaptation protocols with pseudo-labels is most likely caused by a misalignment of the pseudo-label class distribution and the test set class distribution. This highlights an interesting problem for future work in pseudo-labelling methods for this dataset and similarly difficult ones.

CHAPTER 5

CONCLUSION

The aim of this work is to explore pseudo-labelling approaches for the unsupervised video domain adaptation of egocentric datasets. A better understanding of this problem as a whole is first discussed in chapter 2, along with an explanation of popular video understanding architectures and methodologies. The prior work of [11] is then used as a baseline in chapter 3, as this is a rather successful piece of research which solely uses pseudo-labelling for adaptation. However, it uses the I3D training procedure from [30] as its backbone, whereby random, short clips of variable length videos are sampled for the network to use in training. As is shown in chapter 3, this means that variations in both feature vectors and prediction vectors exist across the entire range of the videos. This is identified as a source of additional information that is not utilised by the cleanAdapt method [11]. New pseudo-labelling methods which do utilise this information are therefore developed and analysed alongside the baseline to explore the two key areas of pseudo-labelling: initial pseudo-labelling accuracy and adaptation performance with said pseudo-labels.

Chapter 4 further explores the pseudo-labelling regime of [11], and how it scales to other problems, by applying similar techniques to a much larger and complex dataset (EPIC-Kitchens-100 [10]) and a new architecture, the transformer. Experiments are conducted in order to see how the methods fair with a much larger amount of classes and different class types, which need to be solved for simultaneously. Furthermore, work is conducted on how the distribution of the chosen pseudo-label classes affects the adaptation performance.

5.1 Contributions

To conclude, the main contributions of this work are as follows:

1. The I3D training procedure (randomly sampled clips, as in [11, 30]) is shown to cause variations across different segments of the same video. Furthermore, this additional information is shown to include a potential source for increasing pseudo-labelling accuracy, which is not exploited by the previous methods discussed.

2. More sophisticated methods, which utilise the full extent of video predictions (rather than simply one segment), are shown to be able to increase pseudo-labelling accuracy over the simple random sampling of segments from cleanAdapt [11]. However, not all of the methods discussed are able to do this, pointing out that the source-trained classifier does appear to be more reliable for pseudo-labelling, on the EPIC-Kitchens-55 [9] at least.

3. Simple noise reduction in pseudo-labels is found not to be the only factor in improving domain adaptation performance on the target domain when utilising a pseudo-labelling only approach. Although two new methods are identified to have much less noise than the re-implementation of cleanAdapt [11], they both degrade performance after adaptation due to bias towards a single large class and a lack of coverage of diverse samples.

4. Similar pseudo-labelling techniques to [11] are found to struggle to scale to the EPIC-Kitchens-100 dataset [10], which has many more classes and also requires classification on different class types (verbs and nouns, as well as their combination into actions). The techniques employed achieve reasonable adaptation increase on verbs, but nouns prove much harder. It is hypothesised, and supported by ablation studies, that the disparity in class distributions between the test and pseudo-label train set has a large role to play in this.

5.2 Implications & Future Directions

The work in this thesis points out many issues with pseudo-labelling only approaches for unsupervised domain adaptation, especially on much larger datasets. However, there are many areas that are not explored here which would be of importance to further investigate the possibilities of pseudo-labels. This is due to constraints on both time and compute. The following are a list of ideas and experiments that would constitute future work:

1. The new pseudo-labelling approaches, discussed in chapter 3, section 3.4, are not an exhaustive list. Further investigation is needed into whether better methods can be developed that avoid the complications of the methods tested in this work.

2. All of the pseudo-labelling approaches, even those with more noise than the re-implemented cleanAdapt [11] run, should be tested for use in the adaptation process. It would be of interest to see whether the other methods fall foul to similar problems of heavy class bias and lack of pseudo-label diversity, and if not, how they avoid such a problem.

3. Due to time constraints, only one of the domain gaps is tested for adaptation on the EPIC-Kitchens-55 dataset [9] with one of the pseudo-label sampling rates. The other five domain gaps, as well as a range of other sampling rates should be explored for both the two methods and the rest in order to build a wider picture of the behaviour of adaptation with differing levels of pseudo-label confidence.

4. The submission to the unsupervised domain adaptation challenge in chapter 4 uses pre-extracted features as a way of speeding up training to meet the deadline. If more time is available, it would be of interest to compare the results of the architecture if it was trained end-to-end with a number of different video backbones (e.g. I3D [2], SlowFast [13], TRN [62]). This would help to determine how much the quality of features played in the final adaptation performance.

5. It would also be interesting to determine how the different pseudo-label approaches discussed in this work may perform if coupled with other methods (e.g. domain adversarial learning). This is often a technique used by unsupervised video domain adaptation works (as seen in [23, 33, 61] for example) and would be interesting to see how the different methods might complement or harm one another's performance.

BIBLIOGRAPHY

[1] *Pytorch implementation of i3d*.
https://github.com/piergiaj/pytorch-i3d.
Accessed: 2022-11-21.

[2] J. CARREIRA AND A. ZISSERMAN, *Quo vadis, action recognition? a new model and the kinetics dataset*, in proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6299–6308.

[3] C. CHEN, W. XIE, W. HUANG, Y. RONG, X. DING, Y. HUANG, T. XU, AND J. HUANG, *Progressive feature alignment for unsupervised domain adaptation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 627–636.

[4] M.-H. CHEN, Z. KIRA, G. ALREGIB, J. YOO, R. CHEN, AND J. ZHENG, *Temporal attentive alignment for large-scale video domain adaptation*, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6321–6330.

[5] Y. CHENG, F. FANG, AND Y. SUN, *Team vi-i2r technical report on epic-kitchens-100 unsupervised domain adaptation challenge for action recognition 2021*, arXiv preprint arXiv:2206.02573, (2022).

[6] Y. CHENG, D. LIN, F. FANG, H. X. WOON, Q. XU, AND Y. SUN, *Team vi-i2r technical report on epic-kitchens-100 unsupervised domain adaptation challenge for action recognition 2022*, arXiv preprint arXiv:2301.12436, (2023).

[7] J. CHOI, G. SHARMA, M. CHANDRAKER, AND J.-B. HUANG, *Unsupervised and semi-supervised domain adaptation for action recognition from drones*, in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2020, pp. 1717–1726.

[8] J. CHOI, G. SHARMA, S. SCHULTER, AND J.-B. HUANG, *Shuffle and attend: Video domain adaptation*, in Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16, Springer, 2020, pp. 678–695.

[9] D. DAMEN, H. DOUGHTY, G. M. FARINELLA, S. FIDLER, A. FURNARI, E. KAZAKOS, D. MOLTISANTI, J. MUNRO, T. PERRETT, W. PRICE, ET AL., *Scaling egocentric vision:*

The epic-kitchens dataset, in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 720–736.

[10] D. DAMEN, H. DOUGHTY, G. M. FARINELLA, A. FURNARI, E. KAZAKOS, J. MA, D. MOLTISANTI, J. MUNRO, T. PERRETT, W. PRICE, ET AL., *Rescaling egocentric vision*, arXiv preprint arXiv:2006.13256, (2020).

[11] A. DASGUPTA, C. JAWAHAR, AND K. ALAHARI, *Overcoming label noise for source-free unsupervised video domain adaptation*, in Proceedings of the Thirteenth Indian Conference on Computer Vision, Graphics and Image Processing, 2022, pp. 1–9.

[12] A. DOSOVITSKIY, L. BEYER, A. KOLESNIKOV, D. WEISSENBORN, X. ZHAI, T. UNTERTHINER, M. DEHGHANI, M. MINDERER, G. HEIGOLD, S. GELLY, ET AL., *An image is worth 16x16 words: Transformers for image recognition at scale*, arXiv preprint arXiv:2010.11929, (2020).

[13] C. FEICHTENHOFER, H. FAN, J. MALIK, AND K. HE, *Slowfast networks for video recognition*, in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 6202–6211.

[14] C. FEICHTENHOFER, A. PINZ, AND A. ZISSERMAN, *Convolutional two-stream network fusion for video action recognition*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 1933–1941.

[15] Y. GANIN AND V. LEMPITSKY, *Unsupervised domain adaptation by backpropagation*, in International conference on machine learning, PMLR, 2015, pp. 1180–1189.

[16] I. GOODFELLOW, J. POUGET-ABADIE, M. MIRZA, B. XU, D. WARDE-FARLEY, S. OZAIR, A. COURVILLE, AND Y. BENGIO, *Generative adversarial nets*, Advances in neural information processing systems, 27 (2014).

[17] R. GOYAL, S. EBRAHIMI KAHOU, V. MICHALSKI, J. MATERZYNSKA, S. WESTPHAL, H. KIM, V. HAENEL, I. FRUEND, P. YIANILOS, M. MUELLER-FREITAG, ET AL., *The" something something" video database for learning and evaluating visual common sense*, in Proceedings of the IEEE international conference on computer vision, 2017, pp. 5842–5850.

[18] A. GRETTON, K. M. BORGWARDT, M. J. RASCH, B. SCHÖLKOPF, AND A. SMOLA, *A kernel two-sample test*, The Journal of Machine Learning Research, 13 (2012), pp. 723–773.

[19] S. IOFFE AND C. SZEGEDY, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in International conference on machine learning, pmlr, 2015, pp. 448–456.

[20] A. JAMAL, V. P. NAMBOODIRI, D. DEODHARE, AND K. VENKATESH, *Deep domain adaptation in action space.*, in BMVC, vol. 2, 2018, p. 5.

[21] W. KAY, J. CARREIRA, K. SIMONYAN, B. ZHANG, C. HILLIER, S. VIJAYANARASIMHAN, F. VIOLA, T. GREEN, T. BACK, P. NATSEV, ET AL., *The kinetics human action video dataset*, arXiv preprint arXiv:1705.06950, (2017).

[22] E. KAZAKOS, A. NAGRANI, A. ZISSERMAN, AND D. DAMEN, *Epic-fusion: Audio-visual temporal binding for egocentric action recognition*, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 5492–5501.

[23] D. KIM, Y.-H. TSAI, B. ZHUANG, X. YU, S. SCLAROFF, K. SAENKO, AND M. CHANDRAKER, *Learning cross-modal contrastive features for video domain adaptation*, in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 13618–13627.

[24] H. KUEHNE, H. JHUANG, E. GARROTE, T. POGGIO, AND T. SERRE, *Hmdb: a large video database for human motion recognition*, in 2011 International conference on computer vision, IEEE, 2011, pp. 2556–2563.

[25] Y. LI, N. WANG, J. SHI, X. HOU, AND J. LIU, *Adaptive batch normalization for practical domain adaptation*, Pattern Recognition, 80 (2018), pp. 109–117.

[26] J. LIN, C. GAN, AND S. HAN, *Tsm: Temporal shift module for efficient video understanding*, in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 7083–7093.

[27] M. LONG, Y. CAO, J. WANG, AND M. JORDAN, *Learning transferable features with deep adaptation networks*, in International conference on machine learning, PMLR, 2015, pp. 97–105.

[28] Y. MADADI, V. SEYDI, K. NASROLLAHI, R. HOSSEINI, AND T. B. MOESLUND, *Deep visual unsupervised domain adaptation for classification tasks: a survey*, IET Image Processing, 14 (2020), pp. 3283–3299.

[29] J. MATERZYNSKA, G. BERGER, I. BAX, AND R. MEMISEVIC, *The jester dataset: A large-scale video dataset of human gestures*, in 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), 2019, pp. 2874–2882.

[30] J. MUNRO AND D. DAMEN, *Multi-modal domain adaptation for fine-grained action recognition*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 122–132.

[31] J. MUNRO, M. WRAY, D. LARLUS, G. CSURKA, AND D. DAMEN, *Domain adaptation in multi-view embedding for cross-modal video retrieval*, arXiv preprint arXiv:2110.12812, (2021).

[32] P. OZA, V. A. SINDAGI, V. V. SHARMINI, AND V. M. PATEL, *Unsupervised domain adaptation of object detectors: A survey*, IEEE Transactions on Pattern Analysis and Machine Intelligence, (2023).

[33] B. PAN, Z. CAO, E. ADELI, AND J. C. NIEBLES, *Adversarial cross-domain action recognition with co-attention*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 11815–11822.

[34] Y. PAN, T. YAO, Y. LI, Y. WANG, C.-W. NGO, AND T. MEI, *Transferrable prototypical networks for unsupervised domain adaptation*, in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 2239–2247.

[35] M. PLANAMENTE, G. GOLETTO, G. TRIVIGNO, G. AVERTA, AND B. CAPUTO, *Polito-iit-cini submission to the epic-kitchens-100 unsupervised domain adaptation challenge for action recognition*, arXiv preprint arXiv:2209.04525, (2022).

[36] C. PLIZZARI, M. PLANAMENTE, E. ALBERTI, AND B. CAPUTO, *Polito-iit submission to the epic-kitchens-100 unsupervised domain adaptation challenge for action recognition*, arXiv preprint arXiv:2107.00337, (2021).

[37] W. PRICE AND D. DAMEN, *An evaluation of action recognition models on epic-kitchens*, arXiv preprint arXiv:1908.00867, (2019).

[38] M. M. RAHMAN, C. FOOKES, M. BAKTASHMOTLAGH, AND S. SRIDHARAN, *On minimum discrepancy estimation for deep domain adaptation*, Domain Adaptation for Visual Understanding, (2020), pp. 81–94.

[39] A. RAMPONI AND B. PLANK, *Neural unsupervised domain adaptation in nlp—a survey*, arXiv preprint arXiv:2006.00632, (2020).

[40] K. SAITO, Y. USHIKU, AND T. HARADA, *Asymmetric tri-training for unsupervised domain adaptation*, in International Conference on Machine Learning, PMLR, 2017, pp. 2988–2997.

[41] K. SAITO, K. WATANABE, Y. USHIKU, AND T. HARADA, *Maximum classifier discrepancy for unsupervised domain adaptation*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 3723–3732.

[42] A. SANTORO, D. RAPOSO, D. G. BARRETT, M. MALINOWSKI, R. PASCANU, P. BATTAGLIA, AND T. LILLICRAP, *A simple neural network module for relational reasoning*, Advances in neural information processing systems, 30 (2017).

[43] G. A. SIGURDSSON, S. DIVVALA, A. FARHADI, AND A. GUPTA, *Asynchronous temporal fields for action recognition*, in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 585–594.

[44] G. A. SIGURDSSON, A. GUPTA, C. SCHMID, A. FARHADI, AND K. ALAHARI, *Actor and observer: Joint modeling of first and third-person videos*, in proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7396–7404.

[45] G. A. SIGURDSSON, G. VAROL, X. WANG, A. FARHADI, I. LAPTEV, AND A. GUPTA, *Hollywood in homes: Crowdsourcing data collection for activity understanding*, in Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, Springer, 2016, pp. 510–526.

[46] K. SIMONYAN AND A. ZISSERMAN, *Two-stream convolutional networks for action recognition in videos*, Advances in neural information processing systems, 27 (2014).

[47] X. SONG, S. ZHAO, J. YANG, H. YUE, P. XU, R. HU, AND H. CHAI, *Spatio-temporal contrastive domain adaptation for action recognition*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 9787–9795.

[48] K. SOOMRO, A. R. ZAMIR, AND M. SHAH, *Ucf101: A dataset of 101 human actions classes from videos in the wild*, arXiv preprint arXiv:1212.0402, (2012).

[49] B. SUN AND K. SAENKO, *Deep coral: Correlation alignment for deep domain adaptation*, in Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14, Springer, 2016, pp. 443–450.

[50] M. TOLDO, A. MARACANI, U. MICHIELI, AND P. ZANUTTIGH, *Unsupervised domain adaptation in semantic segmentation: a review*, Technologies, 8 (2020), p. 35.

[51] A. TORRALBA AND A. A. EFROS, *Unbiased look at dataset bias*, in CVPR 2011, IEEE, 2011, pp. 1521–1528.

[52] D. TRAN, L. BOURDEV, R. FERGUS, L. TORRESANI, AND M. PALURI, *Learning spatiotemporal features with 3d convolutional networks*, in Proceedings of the IEEE international conference on computer vision, 2015, pp. 4489–4497.

[53] E. TZENG, J. HOFFMAN, K. SAENKO, AND T. DARRELL, *Adversarial discriminative domain adaptation*, in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 7167–7176.

[54] E. TZENG, J. HOFFMAN, N. ZHANG, K. SAENKO, AND T. DARRELL, *Deep domain confusion: Maximizing for domain invariance*, arXiv preprint arXiv:1412.3474, (2014).

[55] A. VASWANI, N. SHAZEER, N. PARMAR, J. USZKOREIT, L. JONES, A. N. GOMEZ, Ł. KAISER, AND I. POLOSUKHIN, *Attention is all you need*, Advances in neural information processing systems, 30 (2017).

[56] L. WANG, Y. XIONG, Z. WANG, Y. QIAO, D. LIN, X. TANG, AND L. VAN GOOL, *Temporal segment networks for action recognition in videos*, IEEE transactions on pattern analysis and machine intelligence, 41 (2018), pp. 2740–2755.

[57] C. XIONG, V. ZHONG, AND R. SOCHER, *Dynamic coattention networks for question answering*, arXiv preprint arXiv:1611.01604, (2016).

[58] Y. XU, J. YANG, H. CAO, K. WU, M. WU, AND Z. CHEN, *Source-free video domain adaptation by learning temporal consistency for action recognition*, in European Conference on Computer Vision, Springer, 2022, pp. 147–164.

[59] L. YANG, Y. HUANG, Y. SUGANO, AND Y. SATO, *Epic-kitchens-100 unsupervised domain adaptation challenge for action recognition 2021: Team m3em technical report*, arXiv preprint arXiv:2106.10026, (2021).

[60] ——, *Interact before align: Leveraging cross-modal knowledge for domain adaptive action recognition*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 14722–14732.

[61] Y. ZHANG, H. DOUGHTY, L. SHAO, AND C. G. SNOEK, *Audio-adaptive activity recognition across video domains*, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 13791–13800.

[62] B. ZHOU, A. ANDONIAN, A. OLIVA, AND A. TORRALBA, *Temporal relational reasoning in videos*, in Proceedings of the European conference on computer vision (ECCV), 2018, pp. 803–818.