FINE TUNING OF GLOBAL MODELS: LOCALIZATION OF ANOMALY DETECTION FOR VIDEO SURVEILLANCE AND USER-ADAPTATION FOR BCI SPELLERS

by
SONER ÖZGÜN PELVAN

Submitted to the Graduate School of Social Sciences
in partial fulfilment of
the requirements for the degree of Doctor of Philosophy

Sabancı University
January 2025

ABSTRACT

FINE TUNING OF GLOBAL MODELS: LOCALIZATION OF ANOMALY DETECTION FOR VIDEO SURVEILLANCE AND USER-ADAPTATION FOR BCI SPELLERS

SONER ÖZGÜN PELVAN

Electronics Engineering, Ph.D. Dissertation, January 2025

Dissertation Supervisor: Assoc. Prof. Dr. Hüseyin Özkan

Keywords: Knowledge transfer, global models, fine tuning, bias-variance tradeoff, context tree, domain adaptation, anomaly detection, brain computer interface

Fine-tuning is a technique that leverages knowledge from a source domain to enhance performance in a smaller, more limited target domain. Simple approaches, such as training separate local models for each domain or creating a single global model using all available data, often suffer from inherent limitations. Local models are prone to high variance due to limited data, while global models may exhibit high bias, failing to capture domain-specific nuances. This thesis aims to strike a balance in the bias-variance trade-off by carefully fine-tuning a global model to target domains, with a particular focus on applications in video-based anomaly detection and visually evoked EEG signals for brain-computer interface (BCI) spellers.

The central idea introduced in this thesis is the transition from global to local expertise, starting with a low-variance global model trained on all available data and progressively fine-tuning it to capture domain-specific nuances as local data becomes available. Furthermore, the thesis examines two distinct types of domain structures: hierarchical and non-hierarchical. Hierarchical domains exhibit natural relationships or similarities, allowing structured methods to identify and leverage related data effectively during fine-tuning. In contrast, non-hierarchical domains lack inherent structures, necessitating alternative strategies to manage inter-domain differences and select relevant data for fine-tuning. These strategies aim to optimize performance by addressing the unique challenges posed by each domain structure.

To exploit hierarchical relationships, this thesis employs context tree partitioning to group similar domains, enabling more effective fine-tuning of models. As new data arrives, the transition to local models enhances the localization of anomaly detection by refining both the anomaly labels and their corresponding spatial locations.

Applying this approach to anomaly detection, we observe improved performance on the Street Scene and Shanghai datasets, achieving an Area Under Curve (AUC) of 0.87 with the context tree partitioning method compared to 0.56 when using the entire dataset and 0.80 when using only the smallest partitions. For non-hierarchical data, such as those involving EEG signals, where constructing a hierarchy is not feasible,user adaptation is achieved through direct similarity measures to guide the fine-tuning process. We enhance SSVEP BCI speller performance by adapting a DNN model for each new user without calibration. Starting with a global model trained on labeled data from previous users, the adaptation process leverages unsupervised fine-tuning using pseudolabels generated from the new user's data. This iterative approach significantly improves the character identification accuracy on two publicly available large datasets (BENCH and BETA), particularly at short signal lengths. On the BENCH dataset, initial global model accuracy ranged from 21.75% to 71.32% for signal lengths of 0.2 to 1 second, improving after the first adaptation to 28.28%–88.34% and further to 29.85%–91.55% in subsequent iterations. Similarly, on the BETA dataset, initial accuracy ranged from 19.44% to 51.28%, increasing to 20.66%–66.90% and reaching 19.50%–75.53% after final adaptation. These results highlight the effectiveness of leveraging silhouette scores, normalized distances, and local regularity loss to refine pseudolabels and optimize model performance, particularly for short signals in new user adaptation scenarios.

ÖZET

KÜRESEL MODELLERIN İNCE AYARI: VIDEO GÖZETIMI İÇIN ANOMALI TESPITI YERELLEŞTIRMESI VE BCI YAZIMCILARI İÇIN KULLANICI UYARLAMASI

SONER ÖZGÜN PELVAN

Elektronik Mühendisliği, Doktora Tezi, Ocak 2025

Tez Danışmanı: Doç. Dr. Hüseyin Özkan

Anahtar Kelimeler: Bilgi transferi, küresel modeller, ince ayar, yanlılık-varyans dengesi, bağlam ağacı, alan uyarlaması, anormallik tespiti, beyin bilgisayar arayüzü

İnce Ayar, bir kaynak alandaki bilgiyi daha küçük ve sınırlı bir hedef alandaki performansı artırmak için kullanan bir tekniktir. Her bir alan için ayrı yerel modeller eğitmek veya tüm mevcut verileri kullanarak tek bir küresel model oluşturmak gibi basit yaklaşımlar genellikle sınırlamalara sahiptir. Yerel modeller, sınırlı veri nedeniyle yüksek varyansa yatkınken, küresel modeller, alanlara özgü ayrıntıları yakalamakta başarısız olduklarından yüksek yanlılık sergileyebilirler. Bu tez, özellikle video tabanlı anomali tespiti ve beyin-bilgisayar arayüzü (BCI) yazıcıları için görsel olarak uyarılmış EEG sinyallerine odaklanarak, küresel bir modeli hedef alanlara dikkatlice ince ayar yaparak yanlılık-varyans dengesini sağlamayı amaçlamaktadır.

Bu tezde tanıtılan ana fikir, tüm mevcut veriler üzerinde eğitilmiş düşük varyanslı bir küresel modelle başlayıp, daha sonra daha fazla yerel veri kullanılabilir hale geldikçe, alanlara özgü ayrıntıları yakalamak için, modeli kademeli olarak ince ayar yaparak küresel uzmanlıktan yerel uzmanlığa geçirmektir. Ayrıca tez, hiyerarşik ve hiyerarşik olmayan olmak üzere iki farklı alan yapısını inceler. Hiyerarşik alanlar, doğal ilişkiler veya benzerlikler sergileyerek, ince ayar sırasında ilişkili verileri etkili bir şekilde tanımlamak ve kullanmak için yapılandırılmış yöntemlere olanak tanır. Buna karşılık, hiyerarşik olmayan alanlar doğal yapılar içermez ve alanlar arası farklılıkları yönetmek ve ince ayar için ilgili verileri seçmek için alternatif stratejiler gerektirir. Bu stratejiler, her alan yapısının getirdiği benzersiz zorlukları ele alarak

performansı en uyguna getirmeyi amaçlar.

Hiyerarşik ilişkileri kullanmak için bu tez, benzer alanları gruplamak ve modellerin daha etkili bir şekilde ince ayarını yapmak için bağlam ağacı bölme yöntemini kullanır. Yeni veriler geldikçe, yerel modellere geçiş, hem anomali etiketlerini hem de bunların karşılık gelen mekansal konumlarını geliştirerek anomali tespitinin yerelleştirilmesini sağlar. Bu yaklaşımı anomali tespitine uygulayarak Street Scene ve Shanghai veri kümelerinde iyileştirilmiş performans gözlemledik. Bağlam ağacı bölme yöntemiyle Alan Altında Kalan Eğri (AUC) skoru 0.87'ye ulaşırken, tüm veri kümesi kullanıldığında bu değer 0.56 ve yalnızca en küçük bölümler kullanıldığında 0.80 olarak kaydedilmiştir.

Hiyerarşik olmayan veriler için, örneğin EEG sinyallerinde olduğu gibi, bir hiyerarşi oluşturmak mümkün olmadığında, kullanıcı uyarlaması, ince ayar sürecine rehberlik etmek için doğrudan benzerlik ölçütleri kullanılarak gerçekleştirilir. SSVEP BCI yazıcısının performansını, her yeni kullanıcı için kalibrasyon gerektirmeden bir DNN modelini uyarlayarak artırıyoruz. Önceki kullanıcıların etiketli verileri üzerinde eğitilmiş bir küresel modelle başlayarak, uyarlama süreci, yeni kullanıcıdan elde edilen verilerden oluşturulan sözde etiketleri kullanarak denetimsiz ince ayar uygular. Bu yinelemeli yaklaşım, özellikle kısa sinyallerde, iki halka açık büyük veri kümesinde (BENCH ve BETA) karakter tanımlama doğruluğunu önemli ölçüde artırır.

BENCH veri kümesinde, küresel modelin başlangıç doğruluğu, 0.2 ila 1 saniyelik sinyal uzunlukları için %21.75'ten %71.32'ye kadar değişmiştir. İlk uyarlamadan sonra doğruluk %28.28–88.34'e yükselmiş ve sonraki yinelemelerde %29.85–91.55'e ulaşmıştır. Benzer şekilde, BETA veri kümesinde başlangıç doğruluğu %19.44 ile %51.28 arasında değişirken, ilk uyarlamada %20.66–66.90'a, son uyarlamada ise %19.50–75.53'e yükselmiştir. Bu sonuçlar, sözde etiketleri iyileştirmek ve model performansını optimize etmek için silüet skorları, normalleştirilmiş mesafeler ve yerel düzenlilik kaybının kullanımının etkinliğini vurgulamaktadır. Özellikle yeni kullanıcı uyarlama senaryolarında kısa sinyaller için etkili bir kişiselleştirilmiş performans sağlanmaktadır.

ACKNOWLEDGEMENTS

This thesis represents the culmination of years of learning, research, and collaboration, and I am deeply grateful to all who supported me throughout this journey.

First and foremost, I would like to express my sincere gratitude to my advisor, Assoc. Prof. Dr. Hüseyin Özkan, for his unwavering guidance, insightful feedback, and endless patience. His expertise and encouragement have been instrumental in shaping my research and fostering my growth as a scholar.

I am also profoundly thankful to the jury members, Prof. Dr. Berrin Yanık, Prof. Dr. Özgür Gürbüz, Prof. Dr. Ayşın Baytan Ertüzün, Assoc. Prof. Dr. Erdem Akagündüz, for their invaluable input, constructive criticism, and for challenging me to think critically and creatively.

I extend my heartfelt gratitude to my colleagues, friends, and co-authors. I am deeply thankful to Dr. Başarbatu Can for his invaluable insights and contributions on our conference paper. I also appreciate Semih Zaman, for his dedication, engaging discussions, and extraordinary effort in running experiments and collecting results. Finally, I would like to thank Osman Berke Güney, whose prior work and expertise significantly supported the development of our research.

I extend my deepest gratitude to my family for their unconditional love, encouragement, and belief in me, even when the road seemed difficult. To my wife, Hanife Elif, my son Sarp, my parents, Beyhan and Erdoğan, and my brother, İlker Deniz, your support has been my anchor throughout this process.

Finally, this thesis was supported by The Scientific and Technological Research Council of Turkey (TUBITAK) under the Grant Number 121E452. I thank to TUBITAK for their supports.

To my lovely wife and son

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction

Fine-tuning is a pivotal machine learning technique that adapts global models to new domains via knowledge transfer by leveraging prior knowledge from source domains. It is widely applied across tasks such as natural language processing (NLP) (e.g., sentiment analysis, question answering) and computer vision (e.g., image classification, anomaly detection) [21; 22]. Compared to training models from scratch, fine-tuning significantly reduces the need for extensive data and computational resources by reusing general features learned during initial training, allowing models to focus on domain-specific nuances. This makes fine-tuning particularly effective for tasks with limited or no labeled data [23].

One of the key strengths of fine-tuning is its ability to balance generalization and specialization, effectively managing the bias-variance trade-off [24]. Fine-tuning strategies vary based on the relationship between source and target domains: closely related domains require minimal adjustments, while unrelated domains may demand extensive modifications [25]. The process can be further optimized by strategically selecting related data to enhance training efficiency and improve the bias-variance trade-off [26].

A common machine learning scenario involves multiple datasets from different domains (e.g., images presented as art, sketches, photographs, or cartoons [27]) with the goal of developing a robust model effective across all domains, including unseen ones. Two simplistic yet suboptimal approaches include: (1) training a local model for each domain independently, which risks high variance due to limited domain-specific data, and (2) training a single global model with all data, which risks high bias by ignoring domain-specific variations.

The presented thesis addresses these limitations by fine-tuning global models for visual signals, focusing on video-based anomaly detection (Chapter 3) and visually evoked EEG signals in brain-computer interfaces (Chapter 4). The core objective is to achieve a reasonable bias-variance trade-off through controlled fine-tuning. A global model, trained on all available data to minimize variance, is carefully fine-

tuned to reduce bias while maintaining controlled variance. The main challenge lies in preventing excessive variance relaxation during fine-tuning, which requires targeted guidance using domain-specific data. Designing this careful fine-tuning process constitutes the central contribution of this thesis.

Our work begins by demonstrating how dynamically transitioning from global to local expertise can enhance and accelerate an existing system's performance. We study this in the context of video based anomaly detection in Chapter 3. When local data (i.e. data from the target domain) is sparse, global insights provide valuable support; as local data accumulates in time, expertise shifts dynamically to localized models that better reflect domain specific conditions. While also addressing the slow-start problem [28], our approach is particularly relevant in online learning environments [29], where local data characteristics evolve over time. On the other hand, the above mentioned transitioning from global to local expertise can benefit from quantifying the statistical variations across domains. If two domains are sufficiently similar (in terms of statistical properties) then merging is perhaps possible or at least the corresponding local models can benefit from the other's data more than they do when they are less similar. To that end, we propose to use a hierarchy in the space of domains based on the context tree partitioning that was first used for data compression [30] and more recently for classification, regression, contextual bandits and active learning [31–34] and for anomaly detection [35; 36]. In addition, transitioning to local models enables the localization of anomalies, allowing the method to differentiate decisions across different regions and effectively localize both anomaly detection outcomes and their spatial locations. If constructing a hierarchy is not feasible, then we only use a direct similarity measurement without a structure as in the case of our study with EEG signals in Chapter 4. In this case, fine-tuning the model with a similarity measure allows it to adapt to specific users, thereby enhancing its performance.

1.1 Applications of Fine-Tuning Global Models in the Thesis

Visual Signals. To illustrate our approach, in the first part of the thesis (in Chapter 3), we apply it to the challenge of online local anomaly detection, as shown in Fig. 1.1, in video surveillance—a problem that aligns well with our method's adaptation to local and global data as it accumulates [37]. In video surveillance, traffic patterns within a scene can vary significantly throughout the day; for example, early

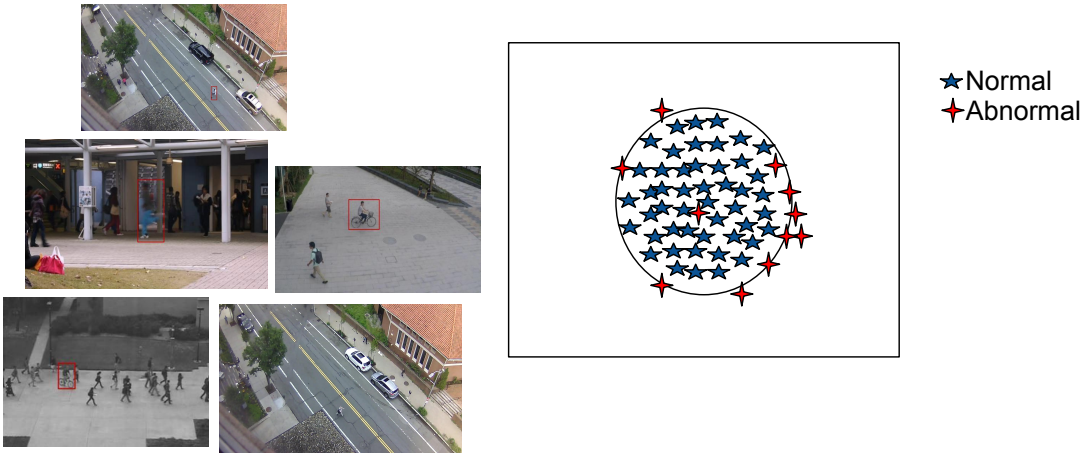


Figure 1.1 The diagram above illustrates the typical architecture of an anomaly detector in video surveillance. In this system, Region Of Interests (ROIs) are extracted to define normal and abnormal behavior. An anomaly detection algorithm is applied to classify between normal and abnormal behavior.

mornings generally see less traffic, while daytime brings a marked increase in activity [38]. Additionally, pedestrian traffic within the same scene fluctuates widely at different times. Also, a pedestrian on the motorway with normal visuals and kinematics, or a red light violation (normal motion at the wrong time) can be anomalous. This variability highlights the need for adaptive knowledge transfer based on local characteristics that shift both spatially and temporally. In this regard, we consider different spatiotemporal regions of the scene (locality) as different domains that can be statistically similar or dissimilar, and the amount of similarity here can quantify the degree of possibility of cross-domains knowledge transfer. By accounting for these spatiotemporal changes, our approach of knowledge transfer with fine tuning aims to achieve a high anomaly detection performance, effectively adapting to the evolving nature of the environment. In our method, we hierarchically partition (via a binary tree) the image space into local regions (domains) sitting at the tree nodes to observe the local (domain) statistics of the video activity. This allows us to observe the similarities between neighboring domains, merging towards upper nodes in the tree, and combining local anomaly detections at the nodes (similar to mixture of experts [39]). Based on the spatial characteristics in each region, and by also accounting for temporal changes, we ensure our model to adapt and deliver peak performance as the data evolves continually. This dynamic approach improves the system's ability to detect anomalies, while also accelerating its overall performance trajectory. Fine-tuning global models to local intermediate models (after possibly merging the lower level nodes/regions) in our approach seeks to continuously optimize the combination of expertise from a set of experts associated with regions.

We accomplish this by assigning a weight to each expert, calculated during training by evaluating the change in each model's expertise and rewarding those with the observed performances. As data evolves, our method dynamically shifts these weights toward the most effective local models, enabling us to achieve and maintain peak performance more quickly and consistently [40]. In addition, local models have the ability to respond differently compared to their global counterparts. While global models lack the inherent ability to localize anomalous events, our approach introduces the capability to localize both the output labels and the actual event locations, adding a crucial localization functionality to anomaly detection.

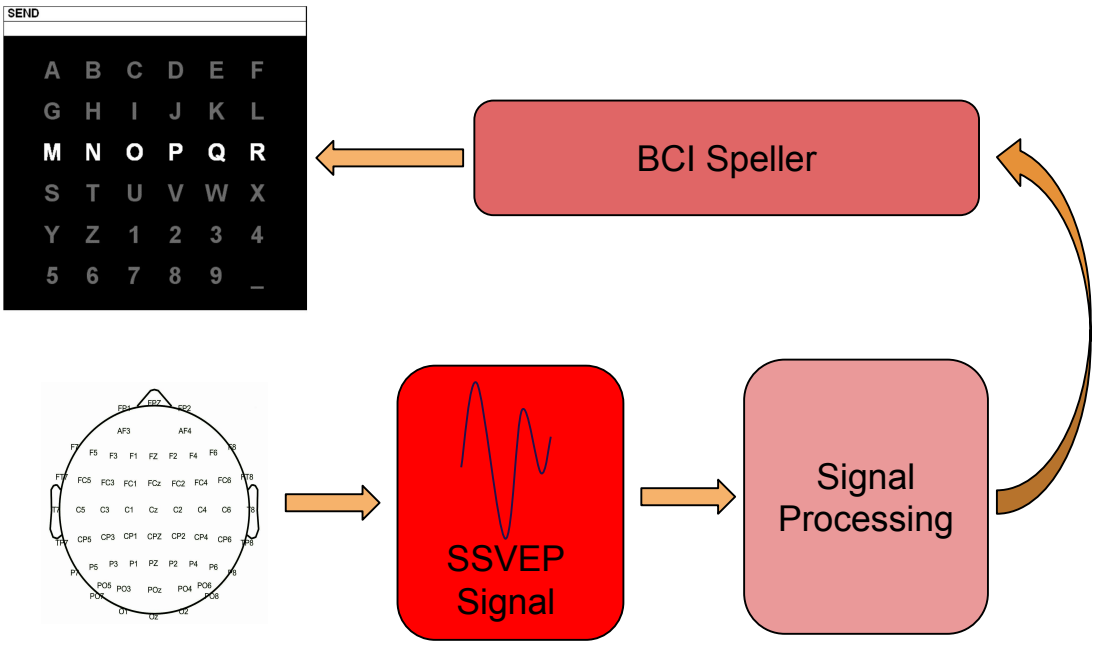


Figure 1.2 The diagram above illustrates the typical architecture of a BCI speller. In this system, the user observes a character matrix and generates SSVEP signals in response. These signals are then processed and translated into input for the BCI speller.

Visually Evoked EEG Signals. In Chapter 4, we focus on an application where it is not straightforward to construct a hierarchy in the space of domains. For this, we chose the problem of character recognition in Steady-State Visual Evoked Potential (SSVEP)-based brain-computer interface (BCI) spellers, as shown in Fig. 1.2.

BCIs enable users to control computer systems using brain signals [41]. Among the various approaches for measuring these signals, non-invasive methods are generally preferred, with electroencephalography (EEG) being one of the most widely used techniques [42]. On the other hand, SSVEPs are brain signals generated when an individual focuses on visual stimuli oscillating at specific constant frequencies [43]. These signals can be captured using EEG. An SSVEP-based BCI speller typically

employs a character matrix where each character flickers at a distinct frequency. By analyzing the SSVEP signals corresponding to the frequency of the focused character, the system can differentiate between characters, thereby enabling the task of character recognition [41]. To validate the efficacy of our approach of fine-tuning, we use the publicly available datasets BENCH [20] and BETA [44]. These datasets consist of SSVEP data collected from various participants/users during speller experiments, in which participants are instructed to focus on different characters in a character matrix. While there are differences (particularly regarding the signal-to-noise ratio level, BETA is noisier as its experiments are outdoor and so more challenging) between the two datasets, the experimental protocols are similar. The BENCH dataset consists of 35 participants/users whereas BETA consists of 70 participants/users. Each and every participant's EEG signals clearly show the SSVEP effect due to the visual stimulation (SSVEP EEG signals clearly show the fundamental stimulation frequency and also the harmonics), hence this allows knowledge transfer. On the contrary, there exist strong statistical variations across the participants which complicates the task of knowledge transfer and so it is not straightforward. In this context, we regard each participant as a different domain.

We consider that this EEG data is non-hierarchical because, unlike video data where a natural distance metric exists between image space partition regions (e.g., temporal or spatial relationships), there is no straightforward way to measure the "distance" between participants/users. Quantifying cross-domain knowledge transfer with fine-tuning in this context cannot be achieved using a context tree based hierarchy as we employ in the case of visual video signals. Instead, here we opt for a combination of correlation-based similarity measures, pseudo-label strategies, and fine-tuning to achieve knowledge transfer between domains (participants/users). To evaluate the efficacy of our approach of knowledge transfer with fine tuning, we conduct experiments on each dataset by selecting one user at a time as the new user, for whom we assume no labeled data is available. This process is repeated for all users in each dataset. The remaining users are treated as previous users, and their associated data is assumed to be fully available. In this setup, the new user is designated as the target domain, while the previous users constitute the source domains. We start with a global neural network model trained with all the available data from all the previous users (source domains), and it is fine tuned to each previous user as well as the target user (target domain). Adapting to the target user through unsupervised fine-tuning allows leveraging their unlabeled data to develop a model that integrates seamlessly into the BCI speller, eliminating the need for calibration.

1.2 Thesis Organization

This current Chapter 1 begins with a discussion, setting the foundation for the thesis. We outline our primary idea underlying fine tuning: controlling variance during the bias reduction phase of knowledge transfer as an effective way of managing the bias-variance trade-off. Following this, we described the hierarchical and non-hierarchical applications addressed in the thesis. Chapter 1 concludes in the following section by presenting the novel contributions and highlights of the thesis.

As we regard fine tuning as a knowledge transfer technique which leverages prior knowledge from source domains, a comprehensive literature survey on knowledge transfer is provided in Chapter 2 to highlight the current state of research in the field from a wide perspective. Note that the prior works related to the specific applications (anomaly detection and BCIs) the thesis considers are given separately in their respective chapters.

Chapter 3 focuses on addressing the cross-domains knowledge transfer in the case of the hierarchical data of visual video signals. In Chapter 3, for supervised setting, we propose a method that uses Neyman-Pearson (NP) classifiers combined with a context tree to detect anomalies while maintaining the false positive rate below a predefined threshold. This method serves as a proof of concept and forms the groundwork for our more advanced approach for unsupervised setting. We build upon this initial work in the supervised setting by introducing a loss function designed to remove the dependency on the labels for obtaining an unsupervised method. Note that our preliminary observations for supervised setting were published as a conference proceeding [45] and our main approach for unsupervised setting was published as a journal article [46].

Chapter 4 studies fine tuning of global models in the absence of hierarchical structures, with a focus on SSVEP-based BCI character recognition. We propose the use of pseudo labels for fine-tuning models as well as coherence scores to assess cross domain (participant/user) similarities.

The thesis concludes with final remarks in Chapter 5.

1.3 Novel Contributions and Highlights

In this thesis, we hypothesize that fine-tuning global models can enhance the performance and capabilities of existing methods by incorporating locality and adaptation to unlabeled data. The novel contributions and highlights of this thesis are presented in two parts: anomaly detection in visual video signals (Chapter 3) and visually evoked EEG signals for BCIs (Chapter 4).

Chapter 3 investigates whether global models can be efficiently fine-tuned by leveraging the inherent hierarchy within the data to localize anomalies in real-time, ensuring their practical applicability in real-world scenarios. To achieve this, we propose using context trees to partition the video space, enabling the computation of the appropriate partition model. This approach is designed to enhance the performance of existing anomaly detection algorithms. The specific contributions of this work are outlined below:

- Our context tree-based image partitioning method for local anomaly detection effectively balances the bias-variance trade-off by smoothly transitioning from coarse to finer granularity within a video stream. This adaptive approach ensures convergence to the best available partitioning model, supported by the theoretical guarantees established in [47; 48]. As a result, the method addresses the slow start problem by selecting the best available partition from the outset, enabling strong performance even with limited initial data.

- Our method effectively detects locational anomalies caused by non-stationary spatial statistics. For example, it can identify a pedestrian walking on a motorway as an anomaly, while recognizing that the same activity on a sidewalk is normal.

- Our unsupervised method uses a novel loss function to evaluate partitioning models in data streams, allowing for a smooth progression to higher complexity. This enables the model to adapt effectively without requiring labeled data.

- We observe the following AUC results across different datasets. In the UCSD dataset [9], where there are no local anomalies, our method with CTBAD achieves an AUC of 0.82, while without CTBAD, the AUC is 0.83. Models trained on the smallest partitions result in an AUC of 0.80. For other datasets, including Avenue [10], Shanghai [11], and Street Scene [12], which contain local anomalies, we observe an improvement in AUC when using CTBAD: Avenue

reaches 0.71 with CTBAD, compared to 0.60 without it, and 0.70 with the smallest partitions; Shanghai shows 0.54 with CTBAD, 0.48 without, and 0.59 with the smallest partitions; Street Scene achieves 0.79 with CTBAD, 0.60 without, and 0.69 with the smallest partitions, as expected.

In Chapter 4, we investigate the potential of unsupervised fine-tuning of global models in SSVEP-based BCI spellers to recognize characters for new users without requiring labeled data, thus eliminating the need for calibration. This is accomplished by generating pseudo-labels using the global model and utilizing data from similar users, identified through similarity measures. Our goal is to show that the existing DNN model [49] can be adapted through unsupervised fine-tuning to effectively replace the calibration process. The key contributions and highlights of this approach are outlined below:

- We introduce a self-regularizing approach for new users that generates pseudolabels, enabling the model to iteratively refine its predictions. To enhance the accuracy of these pseudolabels, we leverage the silhouette score to align the new users' features with those of existing users. This iterative process progressively improves pseudolabel accuracy, guiding the model toward true labels and boosting overall performance.

- Our method eliminates the need for a calibration phase in BCI spellers. Instead, users can immediately perform character recognition using previously calibrated data, enabling a seamless and efficient experience.

- The global models initially achieve mean accuracy percentages of 21.75% 38.01%, 51.01%, 63.94%, and 71.32% for character identification with signal lengths of 0.2, 0.4, 0.6, 0.8, and 1 second, respectively, on the BENCH dataset. Notably, following the first model adaptation, these accuracy rates increase substantially to 28.28%, 62.05%, 76.84%, 83.83%, and 88.34%, respectively. Similarly, on the BETA dataset, the global models initially achieve mean accuracy percentages of 19.44%, 34.28%, 43.78%, and 51.28% for character identification with signal lengths of 0.2, 0.4, 0.6, and 0.8 seconds, respectively. Remarkably, these performances improve significantly to 20.66%, 46.84%, 59.35%, and 66.90%, respectively at the end of the first model adaptation.

- The most recently adapted models reach their best mean identification performances with signal lengths of 0.2, 0.4, 0.6, 0.8, and 1 second at the end of the second and final model adaptation. Specifically, the performances improves to 29.85%, 64.99%, 80.08%, 86.73% and 91.55% on the BENCH dataset and to

19.50%, 47.12%, 61.29%, 69.80% and 75.53% on the BETA dataset, respectively.

In the next chapter (Chapter 2), we provide a general literature survey.

2. Literature Survey: Fine-Tuning Global Models for Comprehensive Knowledge Transfer

As we regard the fine tuning of global models as a technique from the literature of knowledge transfer, in this chapter, we present a general literature survey about knowledge transfer to provide a high level perspective. Note that the prior works related to the specific applications (anomaly detection and BCIs) the thesis considers are given separately in their respective chapters.

Common machine learning (ML) processes consist of various tasks, where systems learn by being trained on a specific set of data tailored for a single task. In such cases, the system is generally optimized to perform only the specific task at hand, without the flexibility to adapt to new or different tasks [24]. In contrast, knowledge transfer in ML refers to the process of leveraging a model trained on one task to improve performance on a different but related task. This approach utilizes the knowledge, patterns, or features acquired during the initial task to enhance learning in the new task, resulting in improved performance, reduced training time, or lower data requirements [50].

The primary motivation behind knowledge transfer is to overcome the limitations of resources by leveraging previously acquired knowledge. This concept mirrors processes inherent in human cognition, where individuals can transfer problem-solving skills gained in one domain and apply them effectively in another. Just as humans use expertise from one area to tackle challenges in a different field, knowledge transfer in ML allows models to apply learned patterns from one task to enhance performance on a related task [40].

A common example of knowledge transfer in humans, as illustrated in Fig. 2.1, is a professional musician learning a new instrument. For instance, a pianist might learn to play the guitar much faster than someone without any musical background. Although the two instruments require different technical skills, the pianist's understanding of music theory, rhythm, and coordination provides a solid foundation. Their mastery of scales, chords, and musical timing can be readily applied, allowing

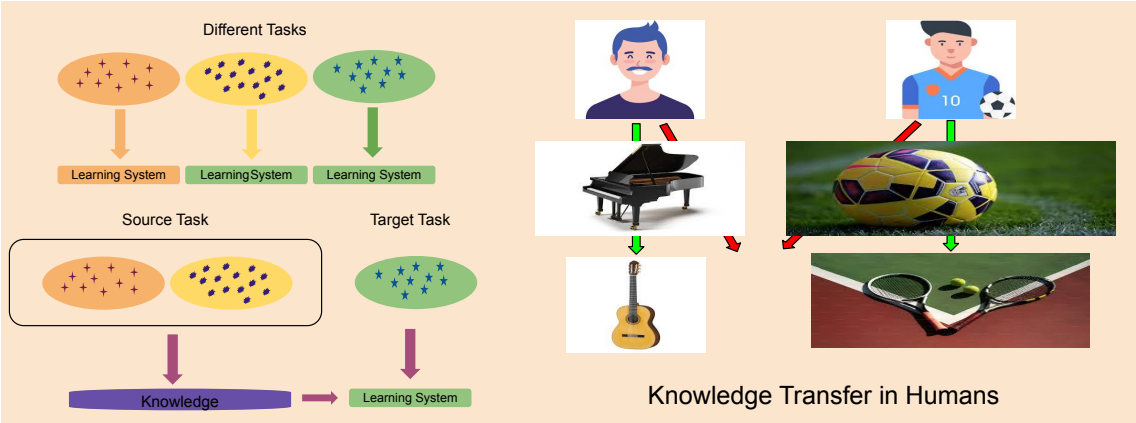


Figure 2.1 In the top left corner, we observe typical ML systems where a separate model is trained for each individual task. While these systems perform well on their respective tasks, they tend to lack robustness when exposed to new data. In contrast, the bottom left illustrates a typical knowledge transfer system. Here, we have a set of source tasks from which general knowledge is extracted and transferred to a target task. This transfer enables better performance on the target task than would be achieved by learning from scratch. An example of effective knowledge transfer can be seen when a musician learns to play a new instrument or a footballer takes up tennis (illustrated with the green arrow). However, if a musician attempts to learn tennis or a footballer tries to learn the guitar (illustrated with the red arrow), these examples do not demonstrate effective transfer, as the tasks are unrelated and no useful knowledge can be transferred.

them to learn the guitar more efficiently than a complete beginner.

Another example, also shown in Fig. 2.1, is a professional athlete transitioning to a different sport. A soccer player learning tennis, despite the two sports being quite distinct, can leverage their agility, stamina, hand-eye coordination, and strategic thinking. These transferable skills enable them to pick up tennis faster than someone with no sports background. This adaptability in applying physical and cognitive skills across different athletic disciplines mirrors how expertise in one domain can accelerate learning in another.

As we observe in the examples for human knowledge transfer, knowledge transfer in ML aims to transfer the expertise of a pre-trained model to a new task where that knowledge can help address the problem. A key principle in designing such systems is selecting related knowledge. The transferred knowledge should align with the target task to be effective. For instance, a professional athlete learning a new instrument would face the same challenges as a complete beginner because their expertise in sports does not directly translate to music. Similarly, in ML, transferring knowledge from unrelated domains may not provide any advantage and could even hinder performance. Therefore, it is crucial to ensure that the source and target tasks are sufficiently related for knowledge transfer to be effective [51].

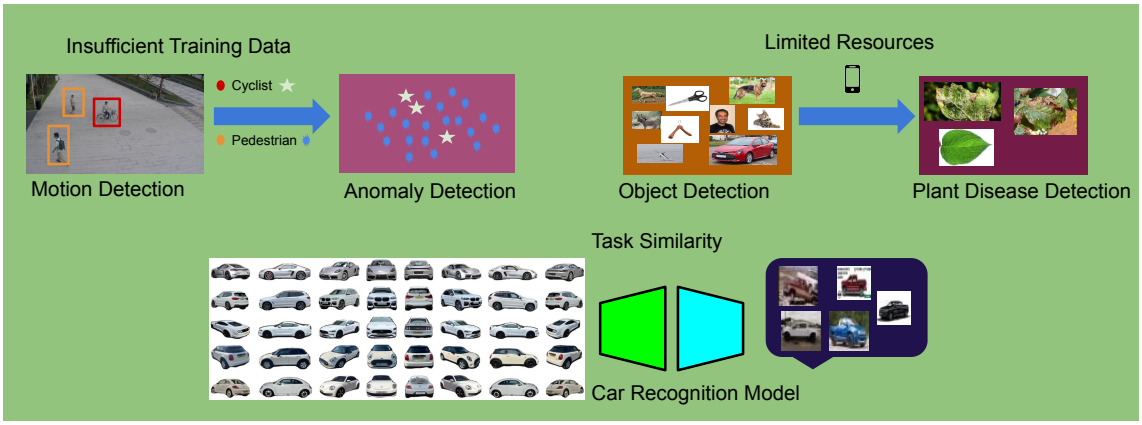


Figure 2.2 The figure above illustrates various applications of knowledge transfer. In the first scenario, we face the challenge of insufficient training data for anomaly detection. Since anomalies are rare and difficult to gather enough data for training, it makes sense to use a model trained on a different but related task which can be leveraged to detect anomalies. By transferring the learned knowledge, we can effectively detect anomalies even with limited data. In the second scenario, the challenge is limited resources. Here, we aim to detect plant diseases using a mobile phone, which has restricted computational capacity. To address this, it makes sense to start with a pre-trained network and leverage its existing knowledge to improve accuracy in this task, despite the device's limitations. In the final scenario, we explore the benefits of applying knowledge across related tasks. For example, as shown in the figure, a model trained to recognize cars can be adapted to differentiate between cars and trucks by utilizing its understanding of the similarities between these vehicle types, thereby improving performance on the new task.

To provide an idea, below are some scenarios, where knowledge transfer is particularly beneficial, as observed in Fig. 2.2:

The first use case for knowledge transfer is when there is not enough data available for the target task. In this case, transferring knowledge from a related task with ample data can significantly improve a model's ability to generalize to the target task. A typical example of a target task with insufficient data is anomaly detection in video surveillance [52]. Anomaly detection involves identifying unusual or suspicious activities—such as theft, accidents, or unauthorized access—against the backdrop of typical daily behaviors [53]. However, since these anomalies are very rare by nature, gathering and labeling sufficient data for these events is a major challenge. Training a robust model from scratch becomes difficult due to the scarcity of labeled anomaly data, as the majority of surveillance footage features normal activities [54].

In such a case, transferring knowledge from a related task can help the model detect anomalies more effectively. By leveraging patterns learned from abundant normal data, the model can better discern what constitutes unusual behavior. This approach not only reduces the dependency on large amounts of labeled anomaly data but also

accelerates the learning process and enhances overall detection performance. [53]. For example we can apply knowledge transfer by utilizing a model pre-trained on a more common tasks like human activity recognition, which has abundant data available. Large datasets such as UCF101 [55] or Kinetics [56] capture general features related to human movements and interactions, which can be valuable for detecting anomalies in surveillance footage. By fine-tuning the pre-trained model with a smaller set of labeled anomaly data (e.g., thefts, accidents, jaywalking), the model can adapt to detect subtle deviations from normal behavior thus detecting anomalies.

This approach works because many anomalies are associated with unusual or unexpected human actions or interactions. The pre-trained model already understands normal behavior patterns, making it easier to recognize irregularities, like someone loitering in a restricted area or behaving erratically. By transferring this knowledge, the model can effectively generalize and detect anomalies with minimal data, improving its performance in real-world surveillance applications [57].

Another case for using knowledge transfer is when computational resources available are limited, and reutilizing an existing model can significantly reduce the need for training from scratch, thereby saving both time and resources. A clear example of this is leveraging pre-trained deep learning models for image classification on devices with constrained processing power, such as smartphones or IoT devices [58].

Consider the case of developing an app for plant disease detection in agriculture, training a deep learning model from scratch to classify plant diseases would demand vast amounts of data, computation, and time which is particularly challenging for a mobile device with limited resources. Instead, we could utilize a pre-trained model like MobileNet [59], which was trained on the ImageNet dataset [60], a large-scale dataset for image classification.

The process would involve fine-tuning MobileNet by reusing its lower layers, which capture general features like edges, shapes, and textures. We would only need to retrain the upper layers using your specific dataset of plant disease images, greatly reducing the computational effort required. By using a pre-trained model in this way, we can create an app that can perform plant disease classification efficiently, even on devices with constrained hardware, without the need to build a model entirely from scratch [61].

Knowledge transfer is most effective when there is a degree of similarity between the source and target tasks, as this facilitates the learning process for the model. For example, consider the tasks of recognizing cars and trucks, which are both

categorized under vehicles. They share numerous visual features, such as four wheels, a rectangular body shape, windows, and common materials like glass, metal, and rubber. Additionally, both types of vehicles possess distinctive attributes, including headlights, tail lights, and grilles, that can be recognized by a ML model [58].

When a model is trained to recognize cars, it learns to identify general features common to both cars and trucks, facilitating knowledge transfer between the two tasks. This transferability allows the model to leverage the information gained from car recognition tasks, significantly minimizing the need for extensive truck-specific training data. By utilizing a pre-trained car recognition model, we can streamline the training process and enhance performance. In this approach, the lower layers of the pre-trained model—having learned to detect general vehicle features—are retained, while the upper layers, which are more specialized for cars, are fine-tuned using a smaller dataset of truck images. This fine-tuning process enables the model to focus on capturing truck-specific characteristics, such as larger body size and more pronounced grilles, rather than relearning basic vehicle features. As a result, the model becomes more efficient and effective, honing its understanding of the subtle differences between cars and trucks while saving computational resources and reducing training time [62].

Research has shown that leveraging similar tasks in transfer learning (TL) not only improves accuracy but also reduces the amount of labeled data required for new tasks. By reusing learned features, models can generalize better across related domains, resulting in more efficient and effective training processes [58].

2.1 Fine Tuning of Global Models in Knowledge Transfer Techniques

In this section, we summarize several techniques employed in knowledge transfer, which enable the transfer of learned knowledge from one task to another. The fundamental concept is to distill knowledge so it can be effectively moved across domains.

Knowledge transfer in ML involves applying insights and patterns acquired from one task or domain to enhance performance on a different, yet related task. This approach is particularly beneficial when labeled data for the target task is scarce, while a substantial amount of data exists for related tasks. By leveraging previously gained knowledge, models can improve their learning efficiency and generalization

capabilities [63].

Feature extraction, as seen in Fig. 2.3, is a crucial technique in ML and deep learning that involves identifying and isolating relevant characteristics or patterns from raw data to improve model performance on specific tasks. In this process, a pre-trained model—often trained on a large dataset—is utilized to extract features that capture essential information while ignoring irrelevant details. For example, in image classification tasks, convolutional neural networks (CNNs) can learn to detect edges, textures, and shapes that are useful for distinguishing between different classes. The extracted features can then be fed into another model, significantly reducing the complexity of the task and enhancing its accuracy, especially when labeled data for the target task is limited. This approach is widely used in various applications, including medical image analysis [64], natural language processing (NLP) [58], and anomaly detection in video surveillance [65].

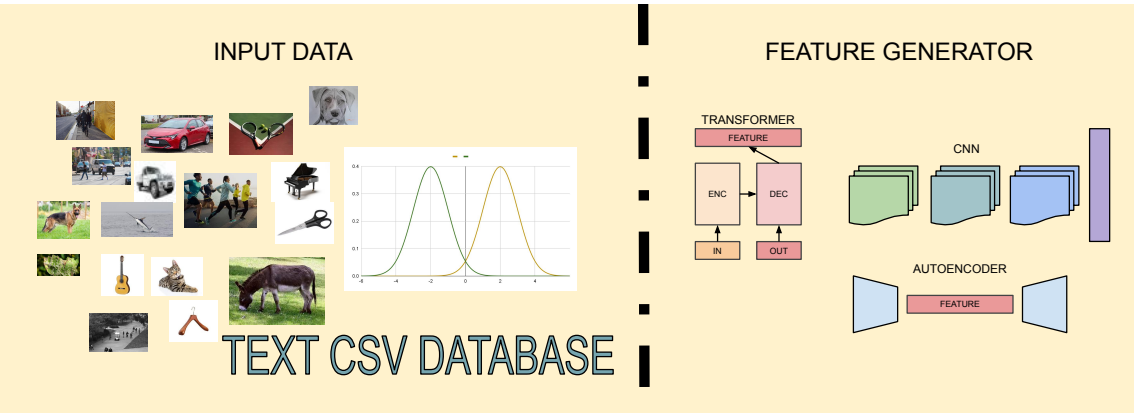


Figure 2.3 The figure above illustrates a typical feature extraction scheme, where data is input into a feature extraction network. This network processes the data to identify and extract relevant features, which can then be utilized for various machine-learning tasks. By effectively isolating key characteristics from the input, the feature extraction network enables more efficient analysis and improved model performance in subsequent stages.

A pre-trained model is taken, and the earlier layers, which have learned generic features such as edges and textures in images are retained whereas later layers, which may have learned more task-specific features such as object categories, are either totally replaced or retrained to fit the new task at hand. The earlier layers are retained by freezing the weights so that the new model can focus on learning task-specific information in the final layers. This approach reduces the amount of training time and data needed, as the early layers already capture meaningful features. This approach is useful when the new task is related to the one the pre-trained model was trained on, so the features learned can be reused [66].

Fine-tuning is a powerful technique in knowledge transfer that involves taking a

pre-trained model and continuing the training process on a new, often related task or domain. Unlike feature extraction, where the model's lower layers are retained for their learned features while only the upper layers are retrained, fine-tuning allows for all or some of the model's layers to be updated during this process. This enables the model to adapt its weights and biases based on the new task, effectively learning to capture nuances specific to the new data while retaining the generalized knowledge gained from the original training [67].

In the process of fine-tuning a pre-trained model, we initiate training using data specific to the new task. Typically, all layers or a selection of layers in the model are unfrozen, allowing the model to adjust its learned weights based on the new task data. Fine-tuning usually begins with a low learning rate to mitigate the risk of catastrophic forgetting, where the model loses useful information acquired during pre-training. This careful approach ensures that the model retains relevant knowledge while adapting to the nuances of the new data.

Fine-tuning is more flexible than feature extraction because it permits adjustments across all layers, enabling the model to learn complex or distinct characteristics of the new task. As a result, fine-tuning allows a pre-trained model to effectively adapt to varying tasks while still benefiting from its initial training. This capability makes fine-tuning a powerful strategy in TL, enhancing performance in scenarios where labeled data for the target task is limited . [23].

Parameter sharing is a crucial technique in multi-task learning that allows different tasks to leverage commonalities in their training processes. In this approach, certain layers or parameters of a model are shared across multiple tasks while other layers remain task-specific. This sharing facilitates the learning of general representations that are beneficial across various tasks, enhancing the model's efficiency and performance [68].

The shared parameters enable tasks to influence each other's learning, leading to improved performance for all involved. For instance, in a model designed for both sentiment analysis and topic classification, the shared layers can learn fundamental linguistic features, such as grammar and word semantics, which are applicable to both tasks. Meanwhile, task-specific layers can fine-tune the model to capture the unique aspects relevant to each individual task [69].

Parameter sharing also reduces the total number of parameters in the model, making it more efficient. This efficiency is particularly beneficial in scenarios with limited data, as the shared knowledge can help improve generalization and reduce the risk of overfitting. In image classification tasks, models like CNNs often share feature

extraction layers for related tasks, such as object detection and segmentation, allowing the model to learn common visual features efficiently. Transformer models, like BERT [70], utilize shared parameters for different language tasks (e.g., text classification, named entity recognition). By sharing parameters in the encoder layers, the model can learn general language representations that benefit multiple NLP tasks simultaneously. Models designed for various languages can share phonetic features while allowing specific layers to adapt to language-specific characteristics. This sharing enhances the model's ability to generalize across languages while still accommodating unique linguistic traits.[71].

Domain adaptation (DA) focuses on the challenge of transferring knowledge between different but related domains. The primary hurdle in DA arises from the differences in data distributions between the source and target domains. For example, a model trained on images of animals in one environment may struggle to generalize to images of the same animals in a different setting due to variations in lighting, background, or perspective [72].

The goal of DA is to minimize the disparity between the data distributions of the source and target domains, enabling a model trained on the source domain to perform well on the target domain. Various techniques can be employed to align the feature representations learned from the two domains [73]. These include modifying the feature space of either the source or target domain to make them more similar. This can involve methods such as domain-invariant feature extraction. Domain-Adversarial training [74] approach introduces a domain classifier into the model, allowing it to learn representations that are difficult for the classifier to distinguish between the source and target domains. The model learns to produce features that confuse the domain classifier, thereby ensuring that the learned features are domain-invariant. This adversarial training encourages the model to focus on the common characteristics of both domains while disregarding domain-specific information. Self-Training technique [75], involves using the model trained on the source domain to make predictions on the unlabeled data from the target domain. These predictions serve as pseudo-labels, allowing the model to fine-tune itself on the target domain's data, enhancing its ability to generalize across domains.

By effectively aligning feature representations, DA enables models to generalize better in scenarios where labeled data is scarce or unavailable for the target domain. This is particularly useful in applications where the source and target domains differ significantly, such as transferring models trained on synthetic data to real-world data. Research in DA has demonstrates its effectiveness in various fields, including computer vision, NLP, and speech recognition, where disparities between training

and deployment environments often pose significant challenge[72].

In computer vision, a variety of techniques leverage the transfer of local data within images to enhance overall performance and accuracy. Local feature extraction methods, such as Scale-Invariant Feature Transform (SIFT) and Speeded Up Robust Features (SURF), play a crucial role in detecting and describing local features. These techniques effectively identify key points and local structures, enabling applications like object recognition and image matching [76; 77]. Additionally, image pyramids, including Laplacian pyramids, facilitate multi-resolution analysis by constructing images at different scales. This approach aids in tasks such as texture synthesis and image blending while preserving essential details [78; 79].

Moreover, techniques such as local contrast normalization enhance visibility by adjusting the contrast of specific regions within an image, which leads to improved segmentation and object detection performance [80; 81]. Patch-based methods like Non-Local Means (NLM) further enhance image quality by averaging similar patches, effectively denoising images while preserving critical features like edges [82; 83]. Region-Based CNN utilize localized regions for object detection, improving performance in complex scenes by transferring information across patches within the network [84; 85].

Collectively, these techniques underscore the significance of local knowledge transfer in optimizing computer vision tasks, showcasing how transferring local data can enhance image processing and analysis. By redistributing local statistics across spatial dimensions, they enhance a model's ability to generalize from limited data, essentially acting as a localized form of knowledge transfer. This process ensures that insights and learned representations from specific regions are shared to inform and improve performance in adjacent areas. Local knowledge transfer is particularly beneficial in scenarios of data scarcity, allowing models to leverage existing knowledge to fill in gaps. By applying features learned from well-represented local regions to those with insufficient data, these methods create a comprehensive understanding of the overall dataset, ultimately leading to improved model performance across various tasks.

In the domain of video surveillance, various studies emphasize the importance of local statistics for enhancing anomaly detection. [86] provide a comprehensive survey that outlines different techniques and highlights the significance of local statistics, such as pixel intensity distributions and spatial patterns, in identifying unusual behaviors. The paper illustrates how models can effectively transfer knowledge from normal activity patterns to recognize deviations, thus facilitating the identification of rare events amidst everyday activities. Similarly, [87] focus on extracting localized

features from video frames, showing that by learning local statistics from normal frames, models can detect significant deviations indicative of anomalies. This approach underscores the value of local statistics as a means of knowledge transfer. [88] contribute by introducing a 3D deep learning framework that captures both spatial and temporal local statistics, enabling the model to learn patterns from local regions and improving the detection of abnormal events across different frames. Finally, [89] present a method for detecting anomalies in crowded scenes by analyzing local motion features, effectively transferring knowledge from well-represented normal motion behaviors to identify unusual actions. Collectively, these studies highlight the critical role of local statistics in anomaly detection, demonstrating how transferring knowledge derived from spatial patterns, motion behaviors, and pixel distributions enhances a model's ability to generalize and identify anomalies in varied scenarios, particularly in the context of data scarcity and variability.

2.2 Challenges

Knowledge transfer is a powerful technique for enhancing model performance across a variety of tasks; however, it presents several challenges that must be carefully addressed to ensure its effectiveness. Key issues include negative transfer, where knowledge from the source task may hinder performance on the target task due to dissimilarities between the domains. Other challenges involve overfitting to the source task, misalignment of feature representations, and the need for substantial fine-tuning, particularly when the source and target domains are not closely related. Lastly, it is essential to recognize that local statistics may be lacking in certain areas of a dataset, while overall statistics can serve to mitigate this data scarcity [50].

Successfully applying knowledge transfer requires practitioners to overcome these challenges by selecting appropriate source models and carefully balancing adaptation and preservation during the fine-tuning process. This involves considering the similarity of tasks, implementing effective fine-tuning strategies, and employing techniques such as gradual layer unfreezing and DA. By proactively addressing these issues, knowledge transfer can lead to significant improvements in model generalization and performance across a wide range of applications, ensuring that the transfer of knowledge genuinely benefits the target task [90].

Negative transfer occurs when knowledge from the source domain or task adversely

impacts the model's performance on the target task. This typically arises when the source and target tasks are too dissimilar, leading the model to apply patterns or representations learned from the source that are irrelevant or misleading in the context of the target task [50].

Such negative transfer is especially problematic when the pre-trained model has learned domain-specific features that fail to generalize. These learned patterns can introduce confusion into the model's decision-making process, particularly when the model's architecture or feature representations are too rigid. In such cases, the model may attempt to force irrelevant knowledge onto the target task, thereby compromising its overall performance [91].

To avoid negative transfer, it's crucial to carefully assess the similarity between the source and target tasks or domains and implement strategies—such as DA or fine-tuning—that can better align the learned knowledge with the specific requirements of the target task [92].

For example, applying a language model trained on legal texts to perform sentiment analysis on movie reviews can lead to negative transfer. The formal, domain-specific language in legal documents may not align with the colloquial, emotionally charged expressions found in movie reviews. This mismatch in language style and context can cause the model to apply irrelevant knowledge, ultimately hindering its performance. Instead of enhancing sentiment analysis, this transfer of knowledge could confuse the model and degrade its ability to accurately interpret the sentiments in reviews, leading to sub-optimal outcomes [93]. Additionally, employing multi-task learning techniques can be beneficial, as these allow the model to be trained on both general and specialized tasks simultaneously, enabling it to capture valuable knowledge from both areas. Furthermore, implementing DA strategies can help tailor the model to the specific distribution and characteristics of the specialized domain [94].

To address the issue of negative transfer, we can take several approaches. First, we can analyze the data distributions and features of both the source and target tasks to ensure that they share sufficient similarity. Another strategy is to selectively transfer only the relevant layers or features from the source model while avoiding the transfer of task-specific components that may not generalize effectively. Lastly, we can employ techniques such as feature alignment or adversarial learning to enhance the generalization of features learned from the source task to the target domain [95].

Another problem in knowledge transfer is choosing the right pre-trained model or source task, which is crucial for its success. A poor choice can lead to feature misalignment between the source and target tasks, resulting in inefficient training or

even negative transfer. In contrast, selecting an appropriate source model can significantly improve performance on the target task [40]. The pre-trained model must have learned features that are relevant to the target task, yet determining the most suitable source task can be challenging. Models trained on highly specific tasks (e.g., medical image classification) might not generalize well to unrelated domains (e.g., everyday object classification). Furthermore, the architectures of the source and target models should be compatible—transferring knowledge between tasks requiring very different model structures (such as from a CNN to an Recurrent Neural Network (RNN)) can introduce difficulties ([96].

For example, if the target task involves classifying satellite images, a model pre-trained on general image datasets like ImageNet [60] may be useful for low-level feature extraction. However, for tasks requiring highly domain-specific knowledge, such as agricultural analysis, a model pre-trained on similar remote sensing data would be a better choice. To choose suitable source models, it is essential to assess task similarity through methods such as comparing data distributions or analyzing the types of features the models have learned. Models pre-trained on large, diverse datasets (like BERT [70] for text tasks, and ResNet [97] for image tasks) tend to capture more general-purpose features, making them valuable for a broad range of TL applications.

Moreover, fine-tuning a pre-trained model requires balancing how much the model adapts to the new task versus how much of the original knowledge is retained. If the model adapts too much, it may suffer from catastrophic forgetting, losing valuable knowledge from the source task. On the other hand, if it adapts too little, the model may fail to capture the nuances of the target task [98]. Therefore, finding the right level of adaptation is key to maximizing the benefits of knowledge transfer.

During fine-tuning, if the model is too flexible (i.e., the learning rate is too high, or too many layers are unfrozen), it may overwrite important features learned from the source task [99]. If too few layers are fine-tuned, or the learning rate is too low, the model may fail to adapt properly to the new task, leading to suboptimal performance [100].

When fine-tuning a pre-trained language model like BERT [70] on a specific sentiment analysis dataset, the model might initially rely on general language understanding. However, if the fine-tuning process aggressively updates the model, it might forget crucial syntactic or semantic patterns learned during pre-training that are beneficial to the target task .

To address the balance problem, one approach is to gradually unfreeze layers during

fine-tuning, beginning with the later layers that are more task-specific and progressively moving to the earlier, more general layers [26]. Another strategy involves using smaller learning rates during fine-tuning to minimize drastic alterations to the model's weights, which helps preserve the pre-trained knowledge. Lastly, we can implement techniques such as Elastic Weight Consolidation (EWC) to prevent significant changes to important parameters that were beneficial in the source task [101].

The availability and quality of data for the target task are crucial to the success of knowledge transfer. If the target dataset is insufficient, it can pose significant challenges:

When fine-tuning a pre-trained model on a small dataset, there's a high risk of overfitting. Overfitting happens when the model learns to memorize the small dataset's details rather than generalize from it, which reduces performance on unseen data [23]. Fine-tuning complex models on small datasets is difficult because the model may not have enough data to learn the target task effectively. With little data, fine-tuning many layers could cause the model to overwrite useful pre-trained knowledge (catastrophic forgetting) [24].

In contrast, if you have access to a large dataset for the target task, the model can more easily adapt to the new task. In such cases, fine-tuning more layers or training a large model can lead to better results. With sufficient data, TL allows faster training and higher performance compared to training from scratch, as the model can build upon its pre-learned representations [102]. For many specialized tasks, data is scarce because labeling can be time-consuming or costly. For example, in medical domains, labeling data requires expert knowledge (e.g., radiologists for medical images). Therefore, TL becomes essential to leverage pre-trained models, but the challenge is how well the model can perform when only limited labeled data is available [103]. In situations with limited data, data augmentation techniques (like generating synthetic data, rotating images, or adding noise) can help by artificially increasing the size of the dataset and introducing more variety to prevent overfitting [104]. Another approach is to fine-tune only the last few layers of the model to prevent overfitting and mitigate the risk of catastrophic forgetting [105].

The size and complexity of the pre-trained model are important considerations. Large models like BERT [70] have millions or even billions of parameters, which makes them powerful but also resource-intensive to work with [106]. Fine-tuning large models requires substantial computational resources. This can be a barrier for smaller organizations or individuals without access to high-performance GPUs or cloud computing. The larger the model, the longer it takes to fine-tune it on a new

task. This increases the cost in terms of both time and computational resources. For example, fine-tuning a model like BERT [70] on a new task can take days or weeks, depending on the size of the dataset and available hardware. These large models require significant memory (RAM or VRAM) for both training and inference. Running models like BERT [70] or large vision models like Vision Transformers on edge devices or less powerful systems can be infeasible.

In some cases, smaller models or model distillation techniques are preferred. Model distillation refers to training a smaller model (a "student" model) to imitate the behavior of a large, complex model (a "teacher" model), reducing the computational cost. Smaller models are easier to fine-tune and can often perform surprisingly well when adapted carefully. To tackle the issue of model complexity, we can consider employing lighter versions of pre-trained models, such as MobileNet [107], which is specifically designed to be less resource-intensive. Additionally, we can apply model compression or distillation techniques to minimize the computational footprint. Lastly, leveraging cloud-based resources or distributed training can help manage the resource demands of larger models.

Task specificity refers to how specialized the target task is compared to the source task. TL often works well when the target and source tasks share similarities. However, when the target task is highly specialized, challenges arise [50].

Pre-trained models (especially those trained on broad, general-purpose datasets like ImageNet [60] or large language models trained on internet text) can capture general representations. These models work well for a wide range of tasks that share common features with the source task. However, for very specialized tasks, such as rare medical conditions in medical imaging, ecological studies, or character recognition in electrocardiogram (ECG) systems, the general features learned from broad datasets may not be sufficient. Specialized tasks often require fine-grained knowledge that general models do not have. In such cases, the TL process might need to be significantly adapted, or the pre-trained model might not provide as much benefit as expected [23].

There can be significant domain gaps between the pre-trained model's task and the target task. For example, a model trained on general news articles might struggle when applied to a highly specific domain like legal document analysis, where the language and structure are quite different. In some cases, transferring knowledge from a general model to a highly specific task requires training more layers of the model from scratch. This is because the model may need to learn entirely new, domain-specific features. Task-specific adaptations may involve fine-tuning more layers or even pre-training the model on a domain-specific dataset (e.g., pre-training

on medical texts for medical NLP tasks) [108].

An illustrative example of task specificity in TL is ECG character recognition. In this domain, identifying shared features across different users poses significant challenges due to the inherent variability in individual ECG signals. Each user's data can exhibit unique characteristics influenced by factors such as physiological differences, noise, and variations in sensor placement. As a result, fine-tuning a pre-trained model to adapt to a specific user's data without succumbing to overfitting becomes a cumbersome process. Moreover, developing a model that effectively accommodates multiple targets—each with its distinct properties—compounds the difficulty. This complexity underscores the need for tailored approaches that can navigate the intricacies of user-specific data while leveraging the strengths of pre-trained models [109].

Local statistics, which capture specific patterns and variations within subsets of the data, might be more readily available in certain regions but absent in others. In such cases, leveraging the overall statistics of the dataset can provide a broader context that informs the relationships and distributions present in the local data. This approach allows us to transfer insights from regions with rich local statistics to those that are underrepresented, effectively enriching the analysis and enhancing the model's ability to generalize across the entire dataset. By utilizing overall statistics to compensate for local deficiencies, we can improve the robustness and performance of our models in areas where data is sparse [110].

2.3 Types of Knowledge Transfer

In this section, we introduce some of the most widely used methods for knowledge transfer in ML. The categorization is somewhat mixed, as certain methods, like DA, can be considered a subcategory of TL. However, due to their broad range of applications, they are often regarded as categories in their own right.

We begin with the most common method, TL, by providing a definition, categorizing its types, and surveying the various techniques used. We then describe DA in detail and conclude the section with an overview of domain generalization (DG).

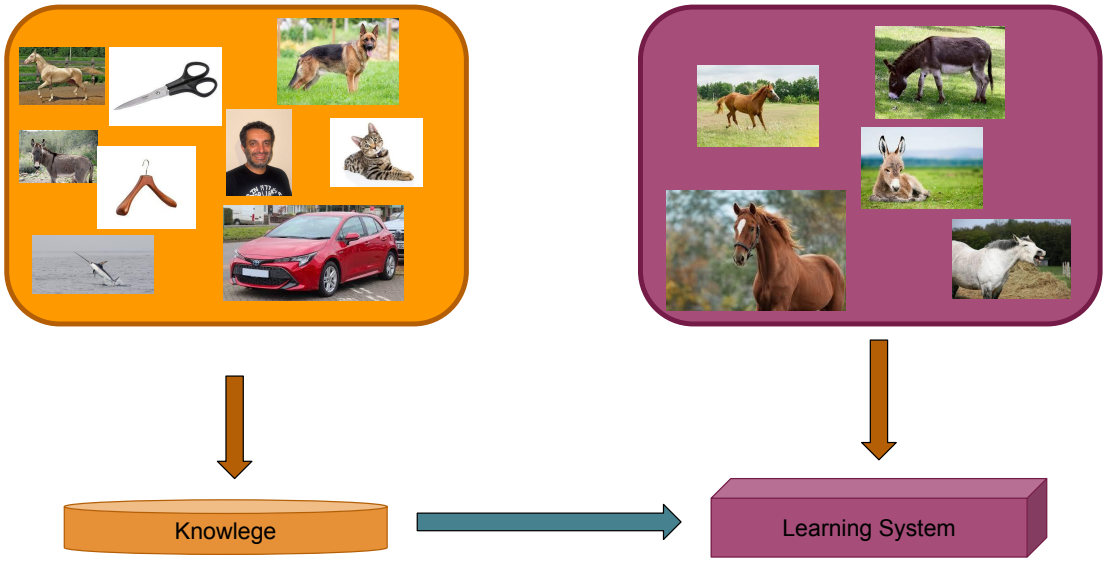


Figure 2.4 An example of the power of TL is using a model trained on a large dataset, like ImageNet, to classify a smaller, more specific task, such as distinguishing between horses and donkeys. By leveraging the knowledge gained from the large dataset and fine-tuning the model on a smaller set of images, the task can be completed with greater accuracy, even with limited training data.

2.3.1 Transfer Learning

Traditional learning algorithms predict future outcomes using models trained on previously collected labeled or unlabeled data [66; 111], as seen in Fig. 2.4. Semi-supervised learning addresses the challenge of limited labeled data by utilizing a small labeled dataset alongside a larger unlabeled one to improve the classifier's performance [112]. TL [50], however, stands out from traditional methods by allowing models to apply knowledge from one domain to another, even when the data distributions differ.

Unlike traditional techniques that learn each task from scratch, TL leverages knowledge from previous tasks to assist in learning a target task, especially when the target task has limited high-quality training data [113–115]. TL reuses a pre-trained model to solve a new, but related problem, allowing the model to generalize knowledge from one task to another. For example, a classifier trained to recognize fruits in images can also help identify vegetables, or a model trained to detect a class of object such as towels can apply that knowledge to recognizing other objects like hangers. By transferring learned model weights from task 1 to task 2, TL improves performance on the new task without starting from scratch with a smaller set of training set [92].

The core idea is to use a model trained on a task with abundant labeled data and apply it to a new task with scarce data. Instead of starting the learning process from

scratch, the model builds upon patterns learned from solving a related task. TL is particularly popular in fields like computer vision and NLP, where tasks like sentiment analysis require significant computational power. Though TL is not strictly a ML technique, it is more of a "design methodology" that spans various fields. It has gained traction, especially when combined with neural networks, which demand large datasets and computational resources [92].

TL can be categorized using different criteria as seen in Fig. 2.5. One way is to classify TL into two categories, homogeneous and heterogeneous based on the similarity of source and target data. Homogeneous TL occurs when the feature spaces of the source and target domains are identical, denoted as $X_S = X_T$. In contrast, heterogeneous TL takes place when the feature spaces differ, represented by $X_S \neq X_T$. Additionally, discrepancies in the marginal probability distributions $P(X_S) \neq P(X_T)$ between the source and target domains can significantly impact model performance. For example, a dataset of X-ray images could be viewed as heterogeneous compared to a dataset of tree species photos in an image-only context. However, it could be seen as homogeneous when compared to the same tree species photo dataset if the comparison also includes audio and text data, illustrating the subjective nature of this classification [116].

In their work, [23] investigate the transferability of features learned by deep neural networks across various tasks, analyzing the impact of different network layers on this transferability. They provide empirical evidence that features from certain layers are more transferable than others, thereby contributing to the understanding of feature reuse in homogeneous TL. Complementing this, [117] explore the use of CNNs for learning mid-level image representations, demonstrating that transferring these representations can significantly enhance performance on new image classification tasks, which further emphasizes the effectiveness of homogeneous TL in visual recognition.

In the realm of heterogeneous TL, as seen in Fig. 2.6, [118] propose a selective TL framework designed to improve cross-domain recommendation systems. Their approach intelligently selects relevant source domains to transfer knowledge to the target domain, showing that selective transfer can enhance recommendation accuracy in heterogeneous settings. Building on this concept, [119] introduces a novel framework that accounts for biases in corresponding instances between heterogeneous feature spaces in the source and target domains. By employing a deep learning approach, their method effectively learns feature mapping and enhances feature representations to mitigate the impact of these biases, thereby improving the TL process in complex scenarios.

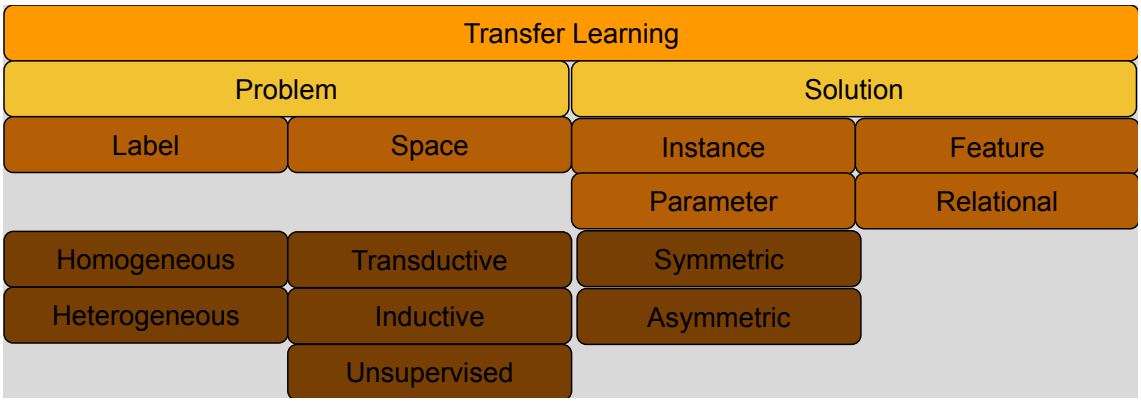


Figure 2.5 This figure categorizes TL into problem-based and solution-based approaches. The problem-based category is divided into two dimensions: label properties, which distinguishes between homogeneous (same label space) and heterogeneous (different label spaces), and the probability space, which is further subdivided into inductive (labeled target data), transductive (unlabeled target data with the same label space), and unsupervised (no labeled data). The solution-based category consists of instance, feature, relational, and parameter transfer methods. Parameter-based methods are classified into asymmetric (partial parameter sharing) and symmetric (complete parameter sharing). This hierarchy provides a comprehensive framework for understanding various TL techniques.

Another categorization of TL methods is based on how they handle labels. The methods can be classified as follows: transductive TL, where only the source data is labeled and the target data remains unlabeled; inductive TL, which involves labeled data for both the source and target domains; and unsupervised TL, where neither the source nor the target data are labeled. Each of these categories addresses different scenarios and challenges in the TL process.

One of the first papers that tackle the problem of transductive TL, [120] introduces transductive support vector machines (TSVMs) for text classification. Unlike traditional SVMs, which focus on learning a model for unseen data, TSVMs aim to optimize the performance on a specific, fixed set of test data. This paper demonstrates that TSVMs can outperform inductive methods when a specific test set is available, making the approach particularly useful for tasks like document classification. In their nominal work [121], the authors propose a framework that blends labeled and unlabeled data for semi-supervised learning, which is closely related to transductive learning. The core idea is that decision boundaries should lie in low-density regions of the data distribution. Their method leverages unlabeled data to create more accurate models, which can significantly improve classification performance when labeled data is scarce.

For inductive TL, [122] introduces deep adaptation networks (DANs), which are designed to transfer knowledge between different domains by minimizing the dis-

crepancy between the source and target domain distributions. It is often cited in inductive TL research for its innovative approach to combining deep learning with TL principles. [74] introduced domain-adversarial training, a powerful technique for inductive TL. The method uses adversarial networks to minimize the discrepancy between source and target domain distributions, effectively learning domain-invariant features. It has become a significant reference in both DA and TL for its ability to improve the performance of ML models on tasks with domain shifts, such as visual recognition and sentiment analysis. In their survey [123], authors provide an extensive overview of self-supervised learning techniques and their relationship to inductive TL, focusing on how self-supervised methods can leverage unlabeled data to enhance learning performance in various tasks.

For unsupervised TL, in their foundational work [124], authors discuss self-taught learning, which enables models to learn from both labeled and unlabeled data through a sparse method, effectively transferring knowledge from a source domain to improve learning in a target domain without supervision. [74] introduces a method for unsupervised DA using a domain-adversarial training approach, allowing a model to learn domain-invariant features by training a classifier alongside a domain discriminator.

Another way to categorize Tls is based on the approaches used, which can be divided into four groups: the first one is instance based approaches where methods use instances or parts of instances from the source data and apply weighting strategies for the target data. [25] presents a method for TL that assigns different weights to instances from the source domain, allowing for more effective knowledge transfer to the target domain. By focusing on the most relevant instances, the authors demonstrate improved performance in various applications. In their work [125], the authors explore the application of SVMs within an instance-based TL framework, proposing a strategy to leverage source instances for better classification in the target domain. The experimental results show that this approach can significantly enhance the learning process when labeled data is scarce. [126] investigates the effectiveness of instance-based TL in knowledge-intensive applications, presenting novel methodologies for transferring learned instances. The authors provide empirical evidence that their approach enhances model adaptability and performance in new environments.

Another approach-based category is feature-based (or mapping-based) approaches where methods map features from the source and target data into more homogeneous data. This can be further divided into two subtypes: Asymmetric feature-based transforms source features to match target ones. [127] proposes a deep learning

framework for DA in sentiment classification, leveraging asymmetric feature transformation to align feature distributions between source and target domains. [128] introduces an asymmetric feature mapping approach that transforms the source domain features to reduce the domain gap while maintaining discriminative information for classification tasks. In their influential paper [128], the authors present DAN, which utilize asymmetric feature transformation techniques to adapt deep learning models to new domains effectively. The approach focuses on minimizing domain divergence while maximizing predictive performance.

Symmetric feature-based methods find a common feature space and transforms both source and target features into this new representation. [127] presents a symmetric feature-based approach for DA that leverages multiple source domains to improve classification performance in a target domain. [129]explores the use of deep symmetric neural networks that are designed to learn a shared feature representation across domains, allowing for effective TL in various applications. Model-based (or parameter-based) approaches: Use knowledge from pre-trained models, with combinations of freezing, fine-tuning, or adding new layers to the network. [130] investigates the effectiveness of fine-tuning deep CNNs that have been pre-trained on large datasets like ImageNet. The authors show that TL can yield substantial improvements in performance across various image classification tasks. The study presents empirical results demonstrating improved performance in diverse applications, including NLP and computer vision.

Another approach-based method category is relational-based (or adversarial-based) approaches, where methods extract transferable features through logical relationships or by applying methods like generative adversarial networks (GANs). In this foundational paper [131], the authors introduce GANs, a novel framework where two neural networks— a generator and a discriminator— are trained simultaneously in a game-theoretic setup. This approach has since been widely adopted in various domains, including TL, for its ability to generate realistic data and improve feature learning. The authors of [132] present a method for adversarial DA that explicitly minimizes the domain discrepancy using a domain discriminator. Their framework enables effective TL by aligning the feature distributions of the source and target domains. [133] explores adversarial learning techniques for semi-supervised DA, combining labeled and unlabeled data to improve classification performance. The proposed method uses adversarial training to align feature distributions, effectively enhancing the model's ability to generalize across domains.

In addition to the aforementioned approaches, using local data as a source of transfer where ample statistics are available can significantly enhance model performance.

One effective method for this is the utilization of context trees, which serve as a powerful tool in TL, particularly for addressing challenges related to out-of-distribution (OOD) generalization. For instance, consider a scenario where a model is being trained to classify images of animals. The training dataset may consist of images from a specific distribution, such as domestic animals like cats and dogs, while the test dataset includes images from a different distribution, such as wild animals like lions and tigers. The primary challenge arises when the model, trained exclusively on domestic animals, fails to generalize effectively to wild animals due to differences in features, backgrounds, and poses that were not represented in the training set. By utilizing context trees, we can create a structured representation of the relevant features for each classification context, thereby bridging this gap [134].

The process begins with creating a context tree where each node corresponds to a context relevant to the classification task. For example, the root node may represent the overall category (animals), with child nodes representing subcategories (domestic vs. wild). As the model processes the data, it learns to capture distinct features associated with each context. When presented with an image of a wild animal, the model leverages the context tree to adjust its predictions based on the specific characteristics of the new data. This enables contextual adaptation, allowing the model to utilize previously learned features—such as color patterns or shapes specific to wild animals—to enhance its classification performance. Overall, by employing context trees, the model improves its ability to generalize to OOD samples, resulting in enhanced accuracy and robustness in classification tasks [110].

A significant challenge in contrast to traditional TL techniques is addressing temporal changes in the statistical properties of both the source and target domains. Unlike standard approaches, which assume static data distributions, the dynamic nature of these domains necessitates models that can adapt to changes over time. This temporal evolution can significantly impact performance and must be carefully managed to ensure accuracy. In [135], the authors tackle this issue by introducing a framework that enables ML models to dynamically adapt to evolving domains. Their method leverages both historical data from source domains and new data from the changing target domain, effectively addressing the challenges posed by temporal shifts.

Building on this work, [136] proposes a meta-optimizer designed to handle temporal changes in statistical properties, applied to both domain-aware and domain-agnostic TL. This approach incorporates dynamic network freezing and domain shift detection to enhance model adaptability. Additionally, [137] introduces a novel unsupervised continual learning strategy that bridges unsupervised TL and continual

Figure 2.6 The above figure presents three images of people running, which should be classified as the same category in tasks like distinguishing between working and running, despite their significant pixel-level differences. Since traditional classifiers may struggle with pixel-level representations, tags are generated by comparing each image with a set of tagged auxiliary images to identify and aggregate relevant descriptors. Analyzing textual data reveals that these images share latent meanings through tags like "road," "track," and "gym," emphasizing their semantic similarity [1].

learning, focusing on adapting to a gradually evolving target domain presented in sequential batches without labeled data. Their proposed method utilizes episodic memory replay with buffer management and incorporates a contrastive loss to improve the alignment of buffer samples with the incoming data stream.

2.3.2 Domain Adaptation

TL has emerged as a powerful approach to address the challenge of acquiring large-scale labeled data by enabling the transfer of knowledge from a labeled source domain to an unlabeled or sparsely labeled target domain, as seen in Fig. 2.7. However, many TL methods assume that the training and test data share the same distribution, an assumption often violated in real-world scenarios due to factors like data collection from different sources or changes in distribution over time. This results in a phenomenon known as domain shift, which can significantly degrade model performance, necessitating costly retraining or additional data collection. DA specifically tackles this issue by enabling models to adapt to new, unseen distributions, allowing them to generalize well to the target domain despite the domain shift [138].

Unsupervised domain adaptation (UDA), a critical subset of DA, focuses on cases where labeled data is only available in the source domain, with the goal of adapting the model to perform well on the unlabeled target domain. In the context of deep learning, UDA has become increasingly important as it addresses the challenges posed by domain shift while leveraging the powerful hierarchical representations

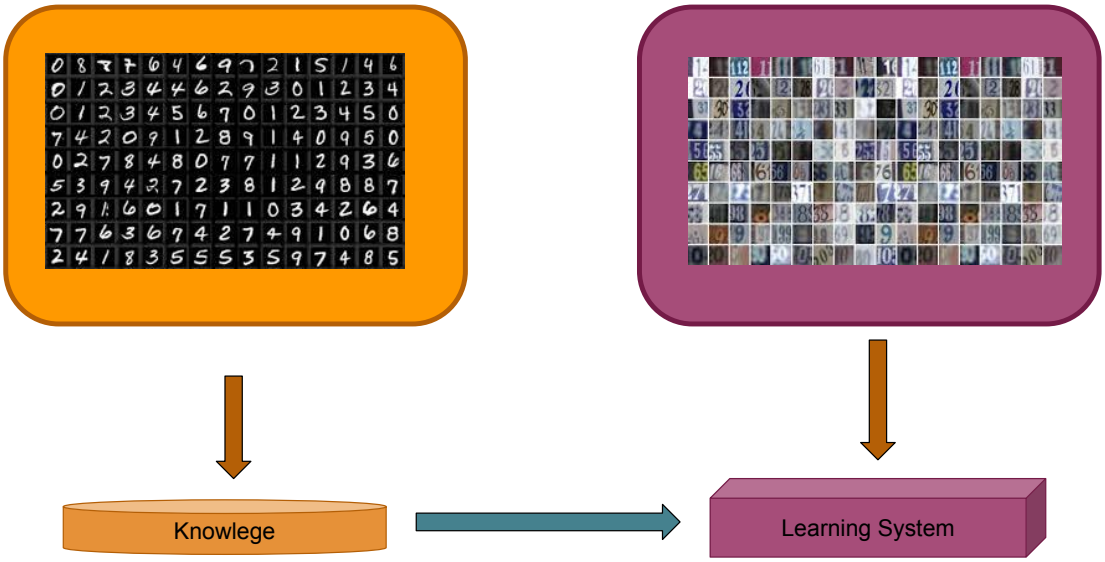


Figure 2.7 The figure above illustrates the straightforward task of learning digit classification using the well-known MNIST dataset [2] and transferring that knowledge to classify digits in the SVHN dataset [3]. In this scenario, both the labels and the tasks are consistent across datasets; however, there is a significant difference between the source domain (MNIST) and the target domain (SVHN), as depicted in the figure. This discrepancy highlights the challenges that arise from domain shift, where the models trained on MNIST may struggle to perform well on SVHN despite the shared task of digit classification.

provided by deep neural networks. This allows models to reduce dependence on labeled target data and maintain performance across domains despite distributional changes [139].

An extension of DA, multi-source domain adaptation (MDA), utilizes labeled data from multiple sources with differing distributions. MDA has gained significant attention in both academia and industry due to the success of DA methods and the increasing availability of diverse multi-source datasets, further improving the model's robustness and adaptability [140].

However, for DA methods, it is assumed that the tasks are the same, i.e., $T_t = T_s$. Generally, these methods are applied to categorization tasks, where both the set of labels and the conditional distributions are assumed to be shared between the two domains, i.e., $Y_S = Y_T$ and $P(Y|X_T) = P(Y|X_S)$. However, this second assumption is quite strong and often does not hold in real-world applications. As a result, the definition of D is relaxed to only require the first assumption, i.e., $Y_s = Y_t = Y$.

Building on the concept of adversarial training, [141] presents an adversarial feature alignment approach that further enhances the alignment of source and target feature distributions. Their proposed method shows considerable improvements in UDAn

tasks, complementing the findings of [142]. In their work, the authors propose a framework that utilizes domain discrepancy to facilitate UDA, achieving better classification accuracy by aligning the feature distributions of the source and target domains.

In a related context, [143] addresses zero-shot learning using visual semantic embeddings to transfer knowledge from seen to unseen classes without requiring labeled data for the latter. This approach effectively improves classification performance in scenarios with limited labeled data, which ties into the broader theme of TL.

The authors of [144] propose a robust and efficient approach to DA for image classification, leveraging a new model architecture that enhances the stability of feature learning across domains. Their method demonstrates superior performance in challenging DA scenarios. Complementing this, [145] proposes a DA approach that matches the source-conditional distributions of the features from the source and target domains, resulting in substantial improvements in adapting models to new domains with minimal labeled data

[146] focuses on adaptive TL techniques to enhance performance in SSVEP-based BCI systems. The authors implement a method that leverages historical data from previous users, thereby reducing the calibration time required for new users and enhancing overall system usability. Building on this theme, [147] explores various DA strategies aimed at improving the performance of SSVEP spellers, specifically focusing on techniques that utilize unlabeled data from new users. The results from this study indicate significant enhancements in classification accuracy and user comfort, showcasing the potential of DA in practical BCI applications.

In a broader context, [148] reviews various DA approaches applied to EEG-based BCIs, including SSVEP systems. This review discusses the effectiveness of different techniques in enhancing the transferability of models across users and tasks while highlighting the challenges and potential solutions in this domain.

Shifting the focus to video surveillance, [149] explores cross-domain learning techniques for detecting anomalies in surveillance videos. The study specifically addresses the challenge of adapting models trained on synthetic data to real-world scenarios, demonstrating that DA can significantly improve detection rates in varied environments. Complementing this work, [150] proposes an unsupervised DA framework that leverages unlabeled data from target domains for anomaly detection. This method aims to bridge the gap between source and target distributions, thereby enhancing the performance of anomaly detection systems in different surveillance environments.

Furthermore, [151] presents a novel framework that integrates DA techniques into existing anomaly detection systems for video surveillance. The authors demonstrate that their proposed approach improves detection accuracy by effectively mitigating the effects of domain shift between training and testing datasets. Collectively, these studies illustrate the potential of DA in both BCI and video surveillance applications, highlighting its importance in improving system performance across varying user and environmental conditions.

2.3.3 Domain Generalization

ML models have achieved remarkable success across a range of applications, including computer vision, speech recognition, NLP, and healthcare. However, their reliance on the assumption that training and testing data are sampled from the same independent and identical distribution (i.i.d.) often limits their real-world applicability due to distribution shifts, where the distribution of data in the training phase differs from that in the test phase [152]. This discrepancy, known as distribution shift, can significantly degrade model performance, making it crucial to develop methods that enhance generalization capabilities [153]. To address this challenge, DG methods, as seen in Fig. 2.8, aim to enable models to perform well on unseen distributions by identifying stable, invariant features or mechanisms across different domains [154].

DG differs from related approaches such as DA and TL, where some access to target domain data is available for model fine-tuning. DG, by contrast, assumes no access to target data during training, making it particularly valuable in scenarios where acquiring new data is impractical or costly. First introduced in [155] in the context of automating cell classification in flow cytometry data, DG addresses cases where shifts in data distribution between patients impair model generalization. In computer vision, the problem of cross-dataset generalization was highlighted by [156], who demonstrates that dataset biases could severely reduce the performance of object recognition models on unseen datasets. To mitigate this issue, [157] proposes learning domain-specific and domain-agnostic components to enhance cross-dataset generalization in classification and detection tasks.

DG methods have been widely applied in various fields, including medical imaging and computer vision, and are typically categorized based on how and when causality is incorporated into the model pipeline. These categories include (i) causal data augmentation applied during pre-processing, (ii) causal representation learning during

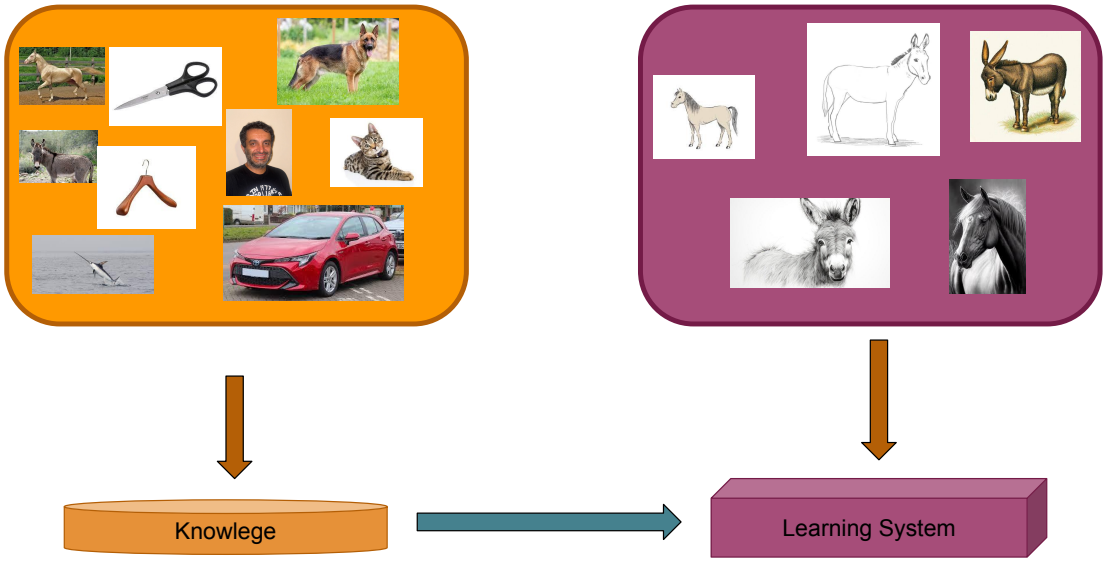


Figure 2.8 In tasks like classifying drawings of horses and donkeys, TL can be used to apply knowledge gained from a large dataset to focus on shape-based features rather than detailed textures. Since drawings often lack the rich details of real images, extracting robust geometric and structural characteristics becomes crucial for accurate classification. This approach helps the model generalize across different visual styles, improving performance in distinguishing abstract representations like sketches.

the feature learning stage, and (iii) transferring causal mechanisms at the classification stage. Causality plays a critical role in capturing invariances, as causal relationships are more stable across different environments than spurious correlations, thus improving OOD generalization [152].

[158] focuses on learning invariant feature representations across different domains to improve generalization capabilities for unseen domains. This foundational work sets the stage for subsequent innovations, such as the approach presented in [159], which involves training models to solve jigsaw puzzles as a method for learning features that generalize well across various domains. Building on the idea of feature extraction, [160] introduces a method that leverages adversarial learning to extract domain-invariant features, enhancing model robustness across diverse domains.

In the realm of data augmentation, [161] discusses strategies for learning data augmentation techniques that promote DG, emphasizing the importance of diverse training data. To improve generalization further, [162] presents a conditional GAN framework that generates aligned samples for unseen domains, effectively bridging the gap between source and target distributions.

Within the context of few-shot learning, [163] introduces Matching Networks, which employ a metric learning approach to classify new examples based on a few la-

beled samples. Their model leverages a novel attention mechanism to match input instances with a support set, achieving state-of-the-art performance in one-shot learning tasks. Complementing this work, [164] proposes a few-shot learning framework that learns a metric space in which classification is performed by computing distances to prototype representations of each class. This method demonstrates superior performance on few-shot classification tasks by effectively capturing class relationships and adapting to new tasks with minimal data.

3. A Hierarchical Approach for Improved Anomaly Detection in Video Surveillance

Anomaly detection for video surveillance gains more attention as the number of deployed cameras constantly increases while the state-of-the-art (SOTA) machine learning methods push the detection performance to its limit. Low complexity methods are relatively straightforward to train (low variance) but suffer from high bias (low performance) whereas, the complex ones can achieve high performance (low bias) with a large sample size to suppress the high variance of estimated parameters. Also, most of the SOTA methods can only detect indigenous anomalies that are spatially stationary, failing at detecting the locational anomalies that are due to nonstationary spatial statistics. To solve these issues, we propose an ensemble technique based on a context tree that generates a hierarchical ensemble of image plane partitions, which we call context tree-based anomaly detection (CTBAD). With CTBAD, partitions yield anomaly detection models of varying complexities, i.e., from coarse to fine details in partitioning with each partition model (which can be any SOTA method) trained separately to allow the detection of locational anomalies, and then we combine them linearly in a weighted manner to achieve a gradual transition from simpler models to more complex ones as more data become available in a video stream. As a result, CTBAD benefits from a low variance of low-complexity methods when the data is sparse and exploits high complexity to achieve low bias when sufficient data is observed. Our experiments show that we significantly reduce the number of training samples to reach the same accuracy as a complex model while successfully detecting the locational anomalies.

3.1 Introduction to Anomaly Detection

With the number of surveillance cameras already reaching 1 billion, real-time manual video analysis becomes impractical and needs to be automated [165]. An important task in the video analysis process is anomaly detection, which aims to identify rare events that do not follow the established normal behavior and so diverge significantly from the majority of the events or observations [166]. Current research in the field concentrates on applying deep learning methods to automate the anomaly detection task mainly in three categories such as generic feature extraction, learning regular representations of normal data, and end-to-end anomaly score learning [167].

Generic feature extraction establishes itself on the basis of transfer learning. These methods [6; 168–172] utilize an existing deep network such as AlexNet [60], VGG [173] and ResNet [174] with the following advantages: i) the availability of existing SOTA (pre-trained) deep models and already established anomaly detectors, ii) deep neural networks' greater strength regarding dimensionality reduction compared to the popular linear methods such as principal component analysis (PCA) [175] and random projections [176], as well as iii) their ease of end-to-end feedforward implementation. Whereas, as a disadvantage, separating anomaly detection from feature extraction usually causes sub-optimal scores.

A better alternative is to develop methods that connect both feature extraction and anomaly detection networks such that the end result is a powerful low-dimensional representation of normal behavior. Instead of employing already trained models, in the second category, both feature extraction and anomaly detection models are trained simultaneously with existing datasets [7; 11; 177–180]. Such methods usually take advantage of AEs/CAEs [181] and GANs to reduce dimensionality, coupled either with well-established one-class classification methods such as one-class support vector machines (SVM) [182], clustering methods such as k-means [183] or distance based measures such as k-neighbours distance [184]. Another approach for learning regular expressions for normal data is to learn the features depending on the anomaly measure. These methods optimize the feature generation task for one particular existing anomaly measure such as distance-based [185; 186], one class classification based [187–190], and clustering based methods [191–195]. Such networks can be introduced to more generalized problems and leveraged but they require abundant data for training.

The third and last category of methods [196–199] utilize deep networks to learn an anomaly score with novel loss functions. In [196], authors propose a self-trained deep

regression model to optimize the anomaly scores for unsupervised video anomaly detection. A Bayesian inverse reinforcement learning-based method is used in [197] which assigns abnormal samples low rewards, whereas normal samples receive high rewards. Another study [198] introduces anomaly score learning from modeling the event likelihood. In [199], the authors study adversarial one-class classification to train two different networks, one to differentiate normal samples from anomalies, and the other to enhance normal samples and generate distorted outliers. To summarize the advantages of this category of methods over the others, the abnormality scores can be individually optimized for the task at hand directly however, the exclusivity of each method to the tasks being considered makes them harder to generalize to other applications.

As we mainly aim, in this study, to provide contributions on the end of anomaly detection rather than learning representations, we opt to stay in the first category to better isolate and demonstrate our technique. The studies in [6; 7; 171], as well as ours here, are from the first category of methods of generic feature extraction with transferring a pre-trained network and they follow a similar process which can be summarized in three steps (cf. Fig. 3.1):

- ROI in the scene are detected by employing an object detection technique such as YoLo [200].
- Features of the available ROIs are extracted either (i) by matching to the receptive field of a set of nodes in a deep neural network that operates on the whole image or (ii) by using a separate network that directly processes the small ROI images.
- A well-established one-class classifier such as one-class SVMs [201] or a thresholding of the reconstruction error (if an AE is used as the ROI feature extractor) [11; 177] is applied to distinguish anomalies from normal behavior.

An important aspect of anomaly detection is *locality*. Decision (step 3 above) in locational anomaly detection methods takes into account the nonstationary spatial statistics [10; 177–180; 202; 203]. Each ROI feature is attached to its location and compared with the learned statistics from only that location, leading to multiple models corresponding to multiple locations. This locality makes the overall method powerful (*low bias*) at the cost of *high variance* since the data is partitioned, i.e., thinned, into locations per model. Hence, overfitting becomes more pronounced as an issue due to thinning in addressing locality. Conversely, if all the ROI features are pooled in the same basket (no partition) by assuming that they are drawn from the same distribution (spatial stationarity) [6; 7; 171], then the method is insensitive to

the locality. Pooling is obviously suboptimal (*high bias*) in case of nonstationarity, but also benefits from *low variance* since all the available data is used for training a single model. The suboptimality in this case can be straightforwardly observed. A local anomaly, for example, is observing a cyclist riding down a sidewalk, whereas observing the same cyclist on a cycle lane is normal and both can be observed in a single scene. Such events are hard to detect for most of the SOTA methods, e.g., [6; 7; 171], which infer the normal behavior from samples arriving over the whole frame and do not consider the local statistics.

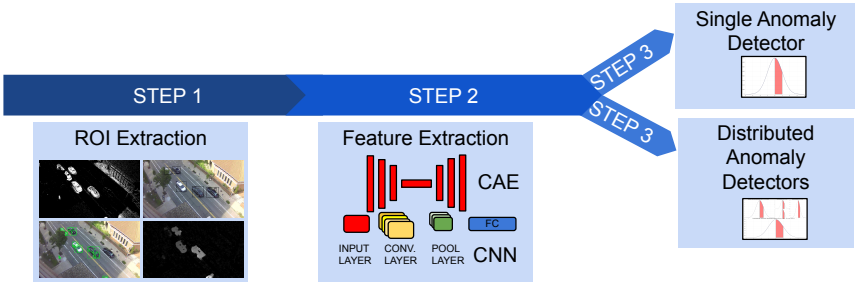


Figure 3.1 Algorithm flow for generic feature extraction methods includes three main steps. In the first step, ROI are extracted from the image using various methods, such as foreground extraction, object detection, motion detection, and optical flow (OF). In the second step, feature extraction is performed on the extracted ROIs using methods such as CAE and CNN. Finally, the last step is split into two paths. In the first path, a single anomaly detector is trained based on the pooling of statistics from all possible locations, but it is unaware of local statistics. In contrast, the second path distributes multiple models to different locations and thus uses distributed anomaly detectors that are location-aware. To summarize, this generic feature extraction algorithm involves ROI extraction in the first step, feature extraction in the second step, and anomaly detection in the last step using either a single anomaly detector or distributed anomaly detectors.

Ensuring a low FPR is crucial for improving the reliability of an anomaly detection model and avoiding unnecessary measures [204]. One effective approach for achieving this is to use the Neyman Pearson (NP) formulation (Fig. 3.2), which introduces asymmetrical class costs to the binary classification problem. The main goal of the NP formulation is to minimize the type II error rate while keeping the type I error rate within a predefined bound [205]. There are several approaches to NP classification in the literature, including thresholding, asymmetric cost learning, constrained optimization, composite models, and online methods.

The thresholding approach calculates a threshold value for a trained classifier based on the desired type I error. For example, in one of the earliest examples of this approach [206], a radial basis function neural network was trained, and various threshold values were tested to achieve the desired type I error. However, since the optimal threshold value is not known beforehand, the best result is obtained

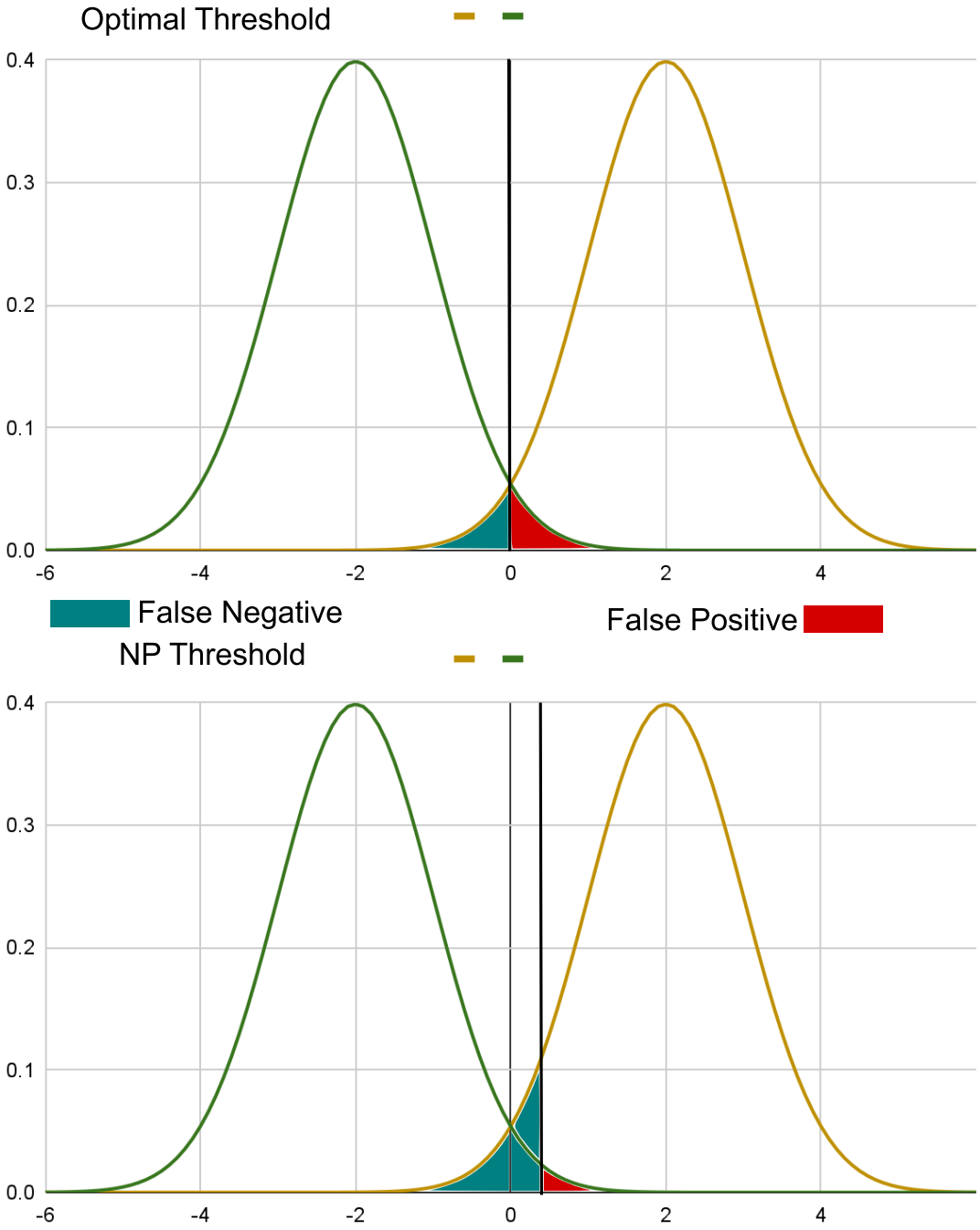


Figure 3.2 NP methods are designed to ensure a pre-specified false positive rate, which is crucial in applications like anomaly detection. These methods allow for the control of false alarms by setting thresholds that correspond to different levels of acceptable false positive rates. In the figure above, we observe that varying the threshold directly impacts the false positive rate. By adjusting the threshold, we can achieve different rates of false positives, providing flexibility based on the specific requirements of the application. For instance, a higher threshold may result in fewer false positives, whereas a lower threshold could lead to more frequent false alarms but might help in detecting more subtle anomalies.

empirically.

In the asymmetric cost learning approach, different costs are assigned to each class through the SVM formulation. In this approach [207; 208], a classifier that targets the desired type I error is obtained by assigning asymmetric costs to different classes. However, the difficulty here is determining the cost values beforehand, as they vary depending on class dependencies and the desired type I error.

In the constrained optimization approach, the asymmetric class costs in the NP classification problem are modeled using the Lagrange multiplier. This approach has both online and holistic implementations [209]. Still, the proposed classifier can only learn linear decision boundaries. Although the non-linear extensions of the same algorithm are mentioned in this study, their implementation is left for future work. Linear and non-linear approaches have been used to tackle online NP classification problems, with online methods being particularly relevant due to their ability to adapt to changing data in real time. However, non-linear approaches have also been proposed to tackle the NP problem. For example, random Fourier features [210] have been used in conjunction with the perceptron algorithm to enable the learning of non-linear decision boundaries online [204].

Composite models have also been proposed to address the NP problem. One such approach is the umbrella algorithm, which creates NP models by combining different algorithms [211] like naive Bayes, SVMs, and decision trees. Although this approach can be used by different models, it may not be scalable to large data sets because it operates as a holistic process. Another composite model presented in the literature is based on the context tree and is designed for online algorithms [204]. This approach combines different classifiers using the context tree, enabling the system to adapt to new data in real-time.

Overall, the literature contains a range of approaches to address the NP classification problem, and the choice of method depends on the specifics of the problem at hand. Linear and non-linear approaches have their own advantages and disadvantages, and composite models can offer additional benefits by combining the strengths of different algorithms. Online methods are particularly relevant in this context due to their ability to adapt to changing data.

In unsupervised anomaly detection, where no examples from the anomaly class ($y = 1$) are available during the training of the classifier, anomalies are often assumed to follow a uniform distribution. Under this assumption, the resulting minimum volume set problem can be addressed effectively [212; 213]. However, this approach may fail if the anomaly class does not adhere to a uniform distribution [214]. The

actual distribution of anomalies depends on the context, data type, and the nature of the anomalies, often deviating significantly from a uniform distribution. In some cases, anomalies exhibit heavy-tailed distributions, such as power-law or Pareto, where extreme values are more common [215]. In other scenarios, anomalies may appear uniformly or randomly distributed [216], form distinct clusters in Gaussian Mixture Models, or occur in low-density regions of the data's probability distribution [217]. Additionally, anomalies in standard distributions are often found in the tail regions, representing extreme values [218], while in multivariate settings, they may manifest as unusual correlations or combinations of features [219]. The choice of a distribution or model for anomalies depends on the characteristics of the data, the anomaly type, and domain-specific insights, which are critical for effective detection and interpretation.

In contrast, in the two-class supervised setting, where both normal and anomalous samples are available during training, the Neyman-Pearson formulation can be employed successfully without assumptions about the underlying anomaly distribution. Moreover, this setting allows the use of 0-1 loss instead of unsupervised loss, simplifying the calculation of anomaly detector performance.

Local anomaly detection method introduced in this thesis, addresses the aforementioned bias-variance trade-off by using a binary tree that partitions the image plane to generate a hierarchical ensemble of partitions (models) of varying complexities (local granularity). The ensemble includes the coarsest single global model at the root of the tree as well as the most powerful model of the finest granularity obtained at the leaves. Based on the weighting scheme in the context tree of [47; 48], our method puts more weight on relatively coarse partition models (high bias but low variance) at the beginning when the data is scarce and gradually switches to more complex models (low bias and variance can still be kept low) as the data size increases. Moreover, we achieve this weighting in a performance-driven online manner during a video stream of ROI features in an unsupervised manner. In this respect, our method also use step 1 and step 2 but replaces step 3 in the above-mentioned process (Fig. 3.1). We aim to create a framework that can be applied to existing methods such as [6; 7; 171] and enhance their performance by taking into account the spatial nonstationarity in the context of bias-variance trade-off. Our proposed algorithm has the advantage of keeping the existing methods as is and is implemented as an additional step which introduces a minimal computational complexity that is used to amalgamate results from different partitioning models. This in return provides the benefits of both local and global models simultaneously.

CTBAD framework possesses remarkable adaptability in accommodating SOTA

methodologies, encompassing a diverse array of feature types and anomaly detection techniques. In our investigation, we rigorously examine an array of feature types from the existing literature, including CAEs, dynamic images, HOF, and motion statistics. It is imperative to note that the framework remains inherently amenable to the seamless integration of any method sourced from the corpus of existing knowledge.

Within the context of our study, the integrated SOTA method assumes the foundational role of the root node model within the CTBAD framework. Correspondingly, the leaf nodes model is built through a piece-wise combination of the aforementioned SOTA methods, deployed across distinct spatial locations. Notably, our algorithmic framework offers an asymptotic performance guarantee, progressively approximating and eventually surpassing the effectiveness of any integrated SOTA method as the scope of observed video activities expands.

Through meticulous experimentation, we consistently ascertain that our CTBAD framework attains a marked and sustained superiority over integrated SOTA methods in terms of performance. This discernible ascendancy is fundamentally rooted in the inherent detection capabilities for locational anomalies. More precisely, upon fusing the CTBAD framework with any chosen SOTA method, the resultant performance aligns with the chosen method or, more frequently, outperforms it. This enhancement hinges on the contextual spatial statistical nonstationarity.

Next, we provide a brief summary of the related work. We continue with the detailed explanation of our proposed method in Section 3.3 and share our results both from simulations and available datasets in Section 3.4. We conclude by summarizing our work and provide how we plan to further expand in the future in Section 3.5.

3.2 Related Work

In our local anomaly detection approach, as in [48], we partition the image plane into small blocks and organize them hierarchically via a binary tree. Without such a hierarchy, features and normality models are defined in [202] for each block separately based on local statistical aggregates. In [203], authors build on that by incorporating the blocks into the histogram of optical flow (HOF) features with local statistics such as size and orientation. In [10], authors resize each video frame into different scales and uniformly partition each scaled image to a set of non-overlapping blocks

of the same size. For a number of consecutive frames, these patches are combined to calculate the features which are then put through a masking process. That ensures a representation for the given training set images from which anomalies are separated based on a distance threshold. Block features are generated, in [180], on the deep layers of a CNN, passing them to an auto-encoder (AE) for obtaining low dimensional local representations. Then, a two-step anomaly detection is employed with the first step of clustering (normal, suspicious, and abnormal). The second step finalizes the decision using a distance threshold for only suspicious and abnormal clusters. In [11] and [177], authors concentrate on using GANs as a pixel-aware model to predict the expected behavior from the video and check the error between actual and expected behavior. The main idea here is that GANs store a statistical model for each input pixel and thus can be thought of as an ensemble abnormality model for the whole scene. Another study [220] proposes to use a student-teacher network scheme to learn pixel-wise anomaly models. These generic models allow the method to detect anomalies with per-pixel statistics. Authors of [221] build up on this idea by concatenating multi-resolution features from separate layers of the network. An anomaly model is introduced in [222] by training the network for different behavioral (motion direction, and motion irregularity) and spatial tasks (reconstruction of object-specific appearance information), simultaneously to create generic local features to detect anomalies. All these methods fail to address the bias-variance trade-off. They tend to utilize atomic abnormality models [202; 203] or an amalgamation of different network levels such as [180; 220; 221] to build features or utilize the whole network as local-aware models [11; 177]. However, all of these methods require thorough training with ample data to be able to detect anomalies, whereas our proposed method achieves a comparable performance with a low number of training samples.

In [47], authors introduce the context tree weighting by applying it to the binary coding problem. Their main idea consists of weighing coding distributions sequentially to realize effective coding for unknown sources. Context trees can also be used for piecewise linear prediction [48]. The authors show that the context tree partitioned predictor achieves the same performance as the best piecewise linear model for every bounded individual sequence. In [32], authors employ the method of nonlinear regression by introducing an online algorithm that mitigates, via a binary context tree, convergence, and lack of training issues of nonlinear regression methods. In [35], the anomaly detection problem in sequential time series data is exploited. In this work, the observation space is divided into disjoint regions, and for each region a new density estimator is trained, and an overall model is built upon the estimators from the disjoint regions. It is shown in [223] that the context

tree weighting algorithm computes the prior predictive likelihood to identify a posteriori most likely models and compute their exact posterior probabilities. In [224], context tree methods are utilized for creating a pattern dictionary for anomaly detection. Others [225; 226] utilize a similar idea, by building up multi-scale features by learning the optimal weights to create refined features.

Deep learning models have been observed to be beneficial for anomaly detection as well [227]. The research related to our work concentrates on 1) utilizing pretrained networks to generate features either by concatenating the network outputs with existing hand-crafted descriptors [6; 171; 172] or by using the network outputs directly [168–170; 178; 228–230], and 2) designing new deep learning networks to learn normal behavior as in [11; 177; 179; 231–233].

Deep networks are employed in [6; 171] for object [200] and motion [5] detection to obtain handcrafted features. Authors assemble the statistical properties (such as skewness and kurtosis [234]) of the motion associated with the object detected and the outputs from the object detection network such as classification scores and the bounding box center. The study in [172] proposes to add the mean squared error between the original and reconstructed image by a GAN for ROI as a new feature dimension. The features are fed, in the last step, to a few shot learning network [235] for anomaly detection, which takes advantage of the persistence of abnormal behavior in time. The authors of [236], propose leveraging self-attention architectures, in particular a spatiotemporal transformer model, for extracting semantic embeddings from videos. In addition authors of [237] propose a new video anomaly detection framework, which utilizes a neural network-based feature extraction module to analyze each new frame. This module identifies the location of objects through object detection, captures appearance information through segmentation, and determines global motion labels using OF analysis. Additionally, it calculates local motion with pose estimation and measures the reconstruction error. These extracted features are then used to create a semantic embedding that represents the detected activity. A deep neural network is trained with metric learning using this semantic embedding to generate an anomaly score for each new frame. Whereas in [238], authors proposed to calculate the nearest neighbor distance for anomaly evidence using a fully connected neural network, which then sequentially decides for anomalous events with a RNN. In their latest work [239], authors present a novel video anomaly detection framework that not only detects anomalous events in surveillance footage but also provides interpretable explanations for the detected anomalies by analyzing object interactions. The method utilizes scene graphs to explain the context of anomalies, offering insights into the root causes while maintaining competitive performance with SOTA approaches. Additionally, the framework supports cross-

domain adaptability, enabling transfer learning in new surveillance environments, and demonstrates strong detection performance both theoretically and empirically on benchmark datasets.

Authors of [170] train a one-class SVM as the anomaly detection method with features from VGG [173] that are fine-tuned to improve the performance. In [168], an anomaly detector is defined based on binary classifiers which are trained with incoming video frames, and the most discriminant frame features are kept in each step of their unmasking process. In the paper, the authors experiment with VGG features and report effectiveness. The masking network is formulated in [169] as a two-sample test. In addition, updating the training pool dynamically has been observed to improve the performance of the framework. The authors in [178] demonstrate the effectiveness of generating deep learning features by applying AEs [181] to both ROIs and the motion data associated with the ROIs, which are fed to one-class SVMs for anomaly detection. A similar method is utilized in [228] but features are obtained from a deep belief network rather than an AE. Unsupervised classification approaches are proposed in [230] for detecting anomalies, by first clustering the CAE [240] features and then regarding each cluster as a class to perform one-vs-rest classification. Similar approaches are also available in graph anomaly detection [229], in which unsupervised clustering-based anomaly detection is employed to detect whether graph vertices or edges are abnormal. The vertices, one-hot encoded, are represented by minimizing AE-based reconstruction loss and pairwise distances of neighboring vertices. In [7], the authors build upon the idea of CAEs. In their work, authors train two CAEs for appearance and motion models, then feed the extracted features to a k-means clustering-based anomaly detector. In addition, [241] couples a denoising AE (to extract representations) with a recursive neural network (RNN) to learn normal patterns from lower dimensional multivariate sequences. A CAE is applied to model regular behavior in frames which is an improvement on AEs by taking advantage of spatial image properties [179]. Both [231] and [233] utilize CNN [242] to learn the spatial properties and combine them with a long short-term memory's (LSTM) [243] temporal data modeling to boost the performance from CNN features. In [244], authors propose to use a sparse coding-based method for feature detection instead of a CNN and feed to an LSTM. In [245], the authors present a video anomaly detection system that combines CNNs for spatial feature extraction and RNNs, such as LSTMs, for temporal sequence modeling to detect anomalies in video streams. The proposed model leverages the complementary strengths of CNNs and RNNs, achieving robust performance in detecting unusual events in surveillance footage.

In [246], authors introduce Anomaly Generative Adversarial Network (AnoGAN)

network, which searches for the latent representation of any input with the minimum distance to the actual input. With the latent space encompassing the sample distribution of the original normal space, anomalies are expected to have a larger distance to their best representation. The problem with this approach is the time it takes to search for the best latent representation. Networks, such as [247] and [248], is proposed to overcome the problem by mapping the best latent representation to the input. In a work similar to [6; 171], authors utilize the mean squared error between the actual and expected frames generated by a GAN as a part of their feature set. [249] introduces a two-stream spatiotemporal generative model (TSSTGM) for real-time abnormal behavior detection in surveillance videos, balancing accuracy and speed. The model uses an end-to-end deep learning framework for video reconstruction and prediction, leveraging reconstruction and prediction errors to detect anomalies, and is designed with a fully convolutional structure to handle input videos of any size. TSSTGM, trained with adversarial learning, efficiently processes appearance, temporal, and motion features.

In [185], the authors employ the random neighbor distance-based anomaly measure to learn low-level representations from high-dimensional data, where the key property is that low-level representations for normal data have smaller distances compared to the abnormal instances. A simpler distance metric between low-dimension representations and randomly projected representations of the same instances is used in [186]. One-class SVM learns the optimal hyperplane maximizing the margin between training data instances and the origin. The main ingredient of deep one-class SVM methods such as [187–189] is to learn the one-class hyperplane from the neural network-enabled low-dimensional representation space instead of the original input space. Authors of [190] take advantage of neural networks to map inputs into the sphere of minimum volume, and then utilize the hinge loss function to guarantee the margin between the sphere center and the projected instances. In addition, training can be combined with feature extraction. Another approach is to extract features tailored for a specific clustering algorithm as in [191–195], developing on this idea with different clustering algorithms as a means to optimize the latent features from different deep networks.

In [250], the authors propose three new algorithms that combine unsupervised deep learning with shallow learning, using Extended Isolation Forest (EIF) for near real-time network traffic anomaly detection. They demonstrate the effectiveness of combining Memory Autoencoder (MemAE) and EIF using SHapley Additive exPlanations (SHAP) for improved result robustness and performance.

In [251], the authors introduce an attention-based residual AE for video anomaly

detection. This model effectively captures both spatial and temporal information by incorporating temporal shifts for efficient temporal modeling and channel attention to exploit channel dependencies. The authors of [252] present a novel AE architecture for video anomaly detection, separating spatial and temporal representations to identify abnormal events in videos. They use an efficient motion AE with consecutive frames and RGB difference for learning regularity in both feature spaces. Additionally, they employ a variance attention module to enhance the motion AE's performance by assigning importance weights to moving parts of video clips. In [253], the authors propose an unsupervised anomaly detection method for video events. They focus on learning temporal correlations by using a dual-stream memory module, integrating high-level semantic information from appearance and motion branches. Feature queues capture historical patterns of normal behavior using momentum-based updates for write operations and OF information for read operations. In [254], a framework based on vision transformers (ViT) [255] is introduced for image anomaly detection and localization. This method combines reconstruction and patch-based learning to effectively detect anomalies in images. Similarly, in [256], the authors propose an encoder-decoder-based method for image anomaly detection and localization. The approach utilizes a ViT-based encoder and a convolutional layer-based decoder with multi-head self-attention from ViT to capture relationships between image patches and learn the distribution of normal data for anomaly detection. In their work [257], the authors introduce a novel video anomaly detection paradigm based on restoring video events from keyframes. This enables more effective mining and learning of higher-level visual features and comprehensive temporal context relationships using ViT.

Further, [257] presents a video anomaly detection system that combines deep CNNs for spatial feature extraction and RNNs, such as LSTMs, for temporal sequence modeling to detect anomalies in video streams. The proposed model leverages the complementary strengths of CNNs and RNNs, achieving robust performance in detecting unusual events in surveillance footage.

[258] addresses the limitations of current anomaly detection approaches in open-world applications by introducing open-vocabulary video anomaly detection (OVVAD). The proposed model decouples OVVAD into two tasks: class-agnostic detection and class-specific classification, using large pre-trained models to detect and categorize both seen and unseen anomalies. It further enhances the model's performance through a semantic knowledge injection module from large language models and an anomaly synthesis module to generate pseudo-unseen anomaly videos. In another approach integrating language models, in [259] the authors introduce Generalist Anomaly Detection (GAD), aiming to train a single detection model that gen-

eralizes across diverse datasets without further training. The proposed in-context residual learning (InCTRL) model leverages few-shot normal images and residual learning to detect anomalies across various domains, significantly outperforming SOTA methods on multiple anomaly detection benchmarks, including industrial, medical, and semantic anomalies.

Bayesian methods are widely used in video surveillance for anomaly detection due to their ability to model uncertainty and incorporate prior knowledge. Dynamic Bayesian Networks (DBNs) are particularly effective for modeling sequential data, enabling real-time anomaly detection in activities such as unattended baggage in public spaces or unusual crowd movements [260]. Additionally, Bayesian nonparametric models, such as Gaussian processes, can identify anomalous events by estimating the likelihood of observations given the learned model of normal behavior [261]. These methods also integrate well with deep learning approaches, like Bayesian convolutional neural networks, which enhance anomaly detection by quantifying prediction uncertainty in complex scenes [262]. The probabilistic nature of Bayesian methods makes them robust in handling noisy or incomplete video data, ensuring reliable detection of subtle anomalies in dynamic environments. Another method utilizes Bayesian nonparametric models to partition videos into temporally consistent and semantically coherent scenes, facilitating robust anomaly detection in real-world surveillance videos with noisy and multimodal scenarios [263]. Lastly, a further approach employs Bayesian feed-forward neural networks to achieve accurate and early anomaly detection and localization in crowded scenes, thereby enhancing surveillance systems' responsiveness to unusual events [264].

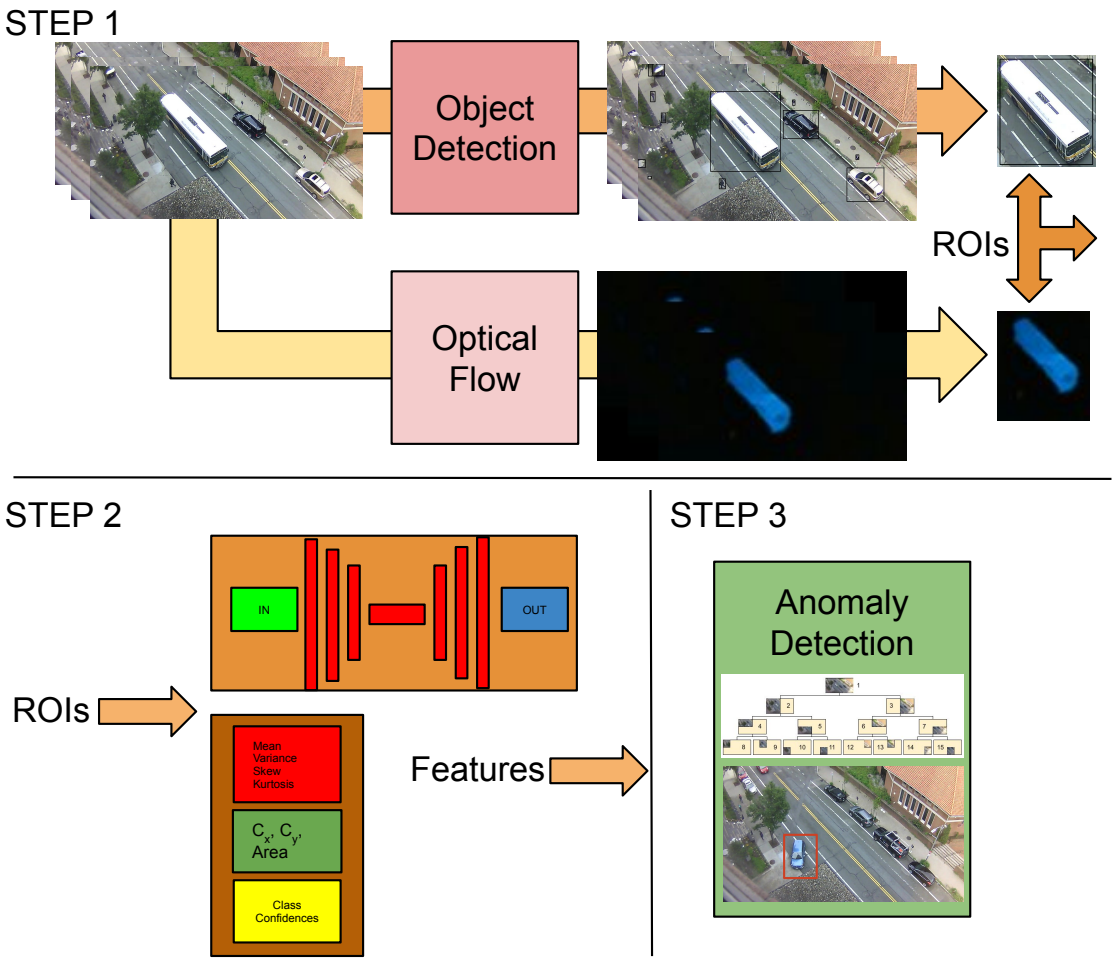


Figure 3.3 Architecture of the proposed anomaly detection framework consists of three main steps. In the first step, the framework uses YoLoV5 [4] for object detection and FlowNet2 [5] for OF estimation to analyze each frame of the video sequence. YoLoV5 identifies objects within each frame, while FlowNet2 estimates the motion between consecutive frames. In the second step, the results from the first step are fed into two feature extractors proposed in [6] and [7]. In [6], the authors propose to extract features related to the object-level appearance and motion, while in [7], the authors propose to extract features related to the pixel-level appearance and motion. Finally, in the third step, the extracted features are used in a context tree-based method to detect anomalies. To summarize, our architecture performs object detection and OF estimation in the first step, feature extraction in the second step using two different feature extractors, and anomaly detection in the third step using a context tree.

3.3 Method

In this section, we first outline our anomaly detection algorithm. This includes a general algorithmic flow and brief explanations. We next continue with the details of our computationally efficient implementation such as the features, context tree

and loss function.

3.3.1 Overview of Our Algorithm

In our algorithm, a video stream is processed frame by frame. We divide each frame F into N rectangular regions. For each region R_i, where $i \in [1, 2, \cdots, N]$, there is an associated local anomaly detector $M_{i,t}$ which is trained with only those video activities that fall inside the region R_i up to and including time t. Video activities are defined by detecting the ROIs inside each frame F with YoLo [4]. It returns the coordinates of the bounding box with the center $C_{bb} = \{x, y\}$ and confidence level for the detected ROI/activity. Based on the bounding box coordinates, a feature vector is extracted for each ROI and assigned to the corresponding region R_i. These regions of activities are designated as "active" for the frame F_t at time t, and their indices are kept in the set of active regions $I_{active}(t)$. In the process of our algorithm, we update the local anomaly detectors, $M_{i,t}$'s, with all the observed ROIs in time (for all $i \in I_{active}(t)$ at each time t).

A partition P_j with $j \in [1, 2, ..., K]$ is defined as the union of disjoint regions. Let I_j be the set of region indices for the partition P_j, and $P_j \ni R_i$ if $i \in I_j$ with $R_k \cap R_l = \emptyset$ and $F = \bigcup_{i \in I_j} R_i$ for all $k, l \in I_j$. Note that there are K different partitions. For each partition P_j, we append the local models from its regions, $M_{i,t}$'s where $i \in I_j$, to cover the whole frame and, thus, obtain an anomaly detection partition model $\overline{M}_{j,t}$ corresponding to the partition P_j.

When the frame F_t is received at time t and YoLo activities are obtained, we calculate/update the current performance of each partition, which is an amalgamation of the performances of the corresponding local anomaly models $M_{i,t}$'s. Then, we assign a weight (proportional to the performance) to each partition and obtain a weighted average of the anomaly decisions of partition models. This weighted average is the final anomaly detection taken by our proposed algorithm. As the number of observed ROIs increases in time, a single partition begins to outperform the others becoming the optimal partition $P_{optimal}$. As time progresses, the performance of our algorithm approaches $P_{optimal}$, thanks to the theoretical guarantees established in [47; 48].

This flow we described is summarized in Fig. 3.3. There are 3 main steps:

- Objects are detected from the input frame.

- Features are extracted from the bounding boxes (ROIs) describing the objects detected. Note that these features define our video activity representations.
- Anomalies are detected using our algorithm: Partitions are generated via a binary partitioning tree and the weighting scheme is based on context tree weighting.

In this framework, given a feature vector x_t for a detected YoLo activity at time t, a partition model $\overline{M}_{j,t}(\cdot) \in \{0,1\}$ performs anomaly detection by accepting the decision of its corresponding local model $M_{i,t}(\cdot)$, i.e.,

$$\overline{M}_{j,t}(x_t) = M_{i,t-1}(x_t),$$

$i \in I_j$ and $x_t \in R_i$. Then, the final anomaly detection in our method is a weighted combination,

$$f(x_t) = \sum_{j=1}^{K} w_{j,t-1} \overline{M}_{j,t}(x_t), \tag{3.1}$$

where the time-varying weights are updated in time proportionally to the partition model performances, and always nonnegative and sum to 1. One of these weights converges to 1, which describes the best-performing partition that our algorithm is asymptotically tuned to.

We describe our method in this order and start with the video activity (or object) features in the following.

3.3.2 Features

To extract meaningful and processable data from each video frame F_t at time t, the first step is to extract set of features ($x_t \in \mathcal{X}_t$) of the scene, which reduces the input dimensions from the whole frame to mere 1D vectors representing the important properties of the activity inside. To achieve this dimensionality reduction, we choose to employ the features utilized in [6] and [7]. However, we want to emphasize here that, for our algorithm, the choice of features is not critical and we can exchange the current features with any others.

Both of the feature extraction methods explained in [6] and [7] employ an object detector as the first step. The object detection method utilized in both of the methods is YoLo, which we also deploy with the implementation provided in [4]. In both

Figure 3.4 Above, we observe the objects detected by YoLoV5 [4] over a video frame from UCSD Pedestrian dataset.

methods, objects[1] are extracted from the unprocessed images and the coordinates of bounding boxes for each detected object are returned, as seen in Fig. 3.4. Features of a detected object are assigned to the frame region (R_i) that includes the center of the bounding box i.e. ROI. Regions of the detected objects become active and their local anomaly detection models ($M_{i,t}$'s) and parameters are updated with the corresponding features. Next, we provide a summary of each feature extraction method, [6] and [7], that is applied to the active regions $R_i \in I_{active}(t)$.

3.3.2.1 Convolutional Autoencoder (CAE) Features

A CAE is utilized in [7] for feature extraction, cf. Fig. 3.5, as summarized in the following list.

- Based on the detected bounding box coordinates, small snippets from the frame are extracted, and processed by the CAE. The processing steps include:
  - Snippets are resized to fit the input of the CAE.

[1] In this work, we decided to use a subset of the object types since they are the most interesting to us. The object types we utilize: $\{person, bicycle, car, motorcycle, bus, truck, cat, dog\}$.

– A batch normalization is applied to the resized snippets. Random Gaussian noise $N(0,1)$ is also added.

- The feature vector is extracted from the bottleneck layer after training the CAE.

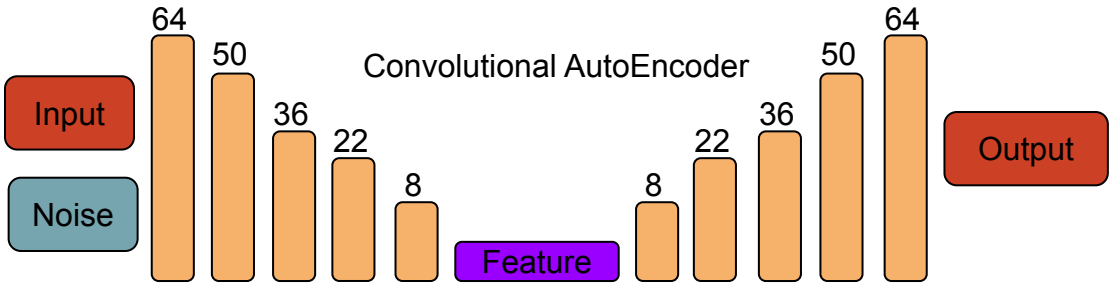


Figure 3.5 The authors of [7] propose the above architecture to extract features from ROIs. In this network, the ROIs are first resized, and random noise with normal distribution is added. Then, the bottleneck layer in the middle of the network provides the feature, and the mean squared error (MSE) between the input and output is added as an additional dimension to the feature. This approach aims to capture the most important attributes of the ROIs while preserving the spatial and temporal information. By including the MSE as an additional dimension, the model can differentiate between normal and anomalous ROIs more effectively.

To use these features in this thesis, we train two different networks. One of the two networks is for the detected objects from unprocessed images, and the other is for dynamic motion images extracted from video. Dynamic motion is obtained via Laplace transforms as explained in [7] for each frame F_t.

3.3.2.2 Flow Features

In [6], authors build a feature vector using both flow features from [5] and [200]. OF, as seen in Fig. 3.6, is calculated in [5] between frames F_t and F_{t-1} for the generation of flow frame F_t^{flow}. Flow features are then extracted by superimposing the bounding box coordinates detected by YoLo on the flow frame F_t^{flow}. The final feature vector includes the following from both frames (F_t and F_t^{flow}):

- Mean, variance, skewness, and kurtosis for the detected bounding boxes from the flow frame F_t^{flow}.
- Center point coordinates and total area of detected bounding boxes.

- Calculated class probabilities from the result of YoLoV5 for each set of classes that we employ [200].

Figure 3.6 The results of the OF algorithm, specifically from [8], show the computed motion between two consecutive frames. The figures on the right illustrate the OF, depicting the direction and magnitude of movement between the frames, as well as the changes in flow over the observed period. This visual representation highlights dynamic elements in the scene and provides insight into motion patterns detected by the algorithm.

In addition to the above, we also utilize the well-known HOF features [265]. The idea is to model the movement of each detected object with HOFs, instead of certain statistics (mean, variance, skewness, and kurtosis) from the flow frame F_t^{flow}. For this purpose, we replace the statistics dimensions of the flow features with the 4-direction HOF.

3.3.3 Context Tree

In our anomaly detection algorithm, we use the context tree weighting method [47; 48] to efficiently (i) determine the set of partitions ($\{P_j\}_{j=1}^{K}$) and (ii) weight and combine them proportionally with respect to their performances. Our use of the context tree weighting method is similar to the feature dimension partitioning described in [48]. However, the applications and motivations in that study [48],

and in this thesis here are completely different. One particular difference is that, instead of dividing feature dimensions using the context tree, we apply it to divide the spatial coordinates of each frame F_t. This allows the context tree to group observations that are from regions with similar statistical backgrounds and speed up the learning process, as the grouped observations are used to train a single model instead of training a different one for each region without checking their statistical similarities. Whereas observations with dissimilar statistics are not grouped to not underfit. To this end, our binary context tree stores multiple anomaly detection models with growing complexity as the depth increases. The increase in complexity enables the combination of all such models (which we propose as the final detection) to evolve in time from simpler (the simplest one corresponds to the whole space) models to more complex ones (the most complex one assigns a different model to each small space region) to overcome the bias-variance trade-off. For this purpose, we utilize the well-known binary tree structure (context tree) to partition a given video coordinate space into non-intersecting regions R_i. By introducing a priori structure to the partitioning problem [266] instead of a random splitting, we reduce the computational complexity to find the appropriate partition model.

We begin by introducing the essential terminology related to context tree implementation. Each member of the context tree is defined as a node, and there are three types of nodes:

- Parent Node: A predecessor of any node.
- Child Node: A descendant of any node.
- Root Node: The highest node in the tree without a parent. (Level 1 in Fig. 3.7, N_1 as the node that has the corresponding region R_1 which covers the whole coordinate space).
- In-Between Nodes: A parent and two children nodes. (Level 2 and level 3 in Fig. 3.7, N_i for $i \in [2,3,4,5,6,7]$).
- Leaf Nodes: A node with no children nodes. (Level 4 in Fig. 3.7, N_i for $i \in [8,9,10,11,12,13,14,15]$).

We build a full binary tree for our algorithm with all the parent nodes having both of the children nodes. Each tree node N_i represents a region R_i from the whole frame, which is divided into two other equal regions assigned to the corresponding children nodes. This division, as shown in Fig. 3.7, can be achieved horizontally or vertically. In our algorithm, we change the division rule at each level, so if a parent node is divided vertically, its children nodes are divided horizontally. The root node

R_0 covers the whole coordinate space. Within this division scheme, regions of the leaves of each pruning yield a different partition of the whole video frame. From now on we use the node N_i interchangeably with the region R_i associated with it.

For a full binary tree with depth D, there are in total $N_{total} = 2^D - 1$ nodes. At each level ℓ, for which $\ell = 1, 2, \dots D$, the union of regions of the nodes at the same level spans the whole space, $F = R_{all} = \cup_{k=1}^{2^{\ell-1}} R_{l,k}$ where $k = 1, 2, \cdots 2^{\ell-1}$ is the node number at level ℓ, and $\emptyset = R_k \cap R_l, \forall k, l$ at the same level of the binary tree, as shown in Fig. 3.7.

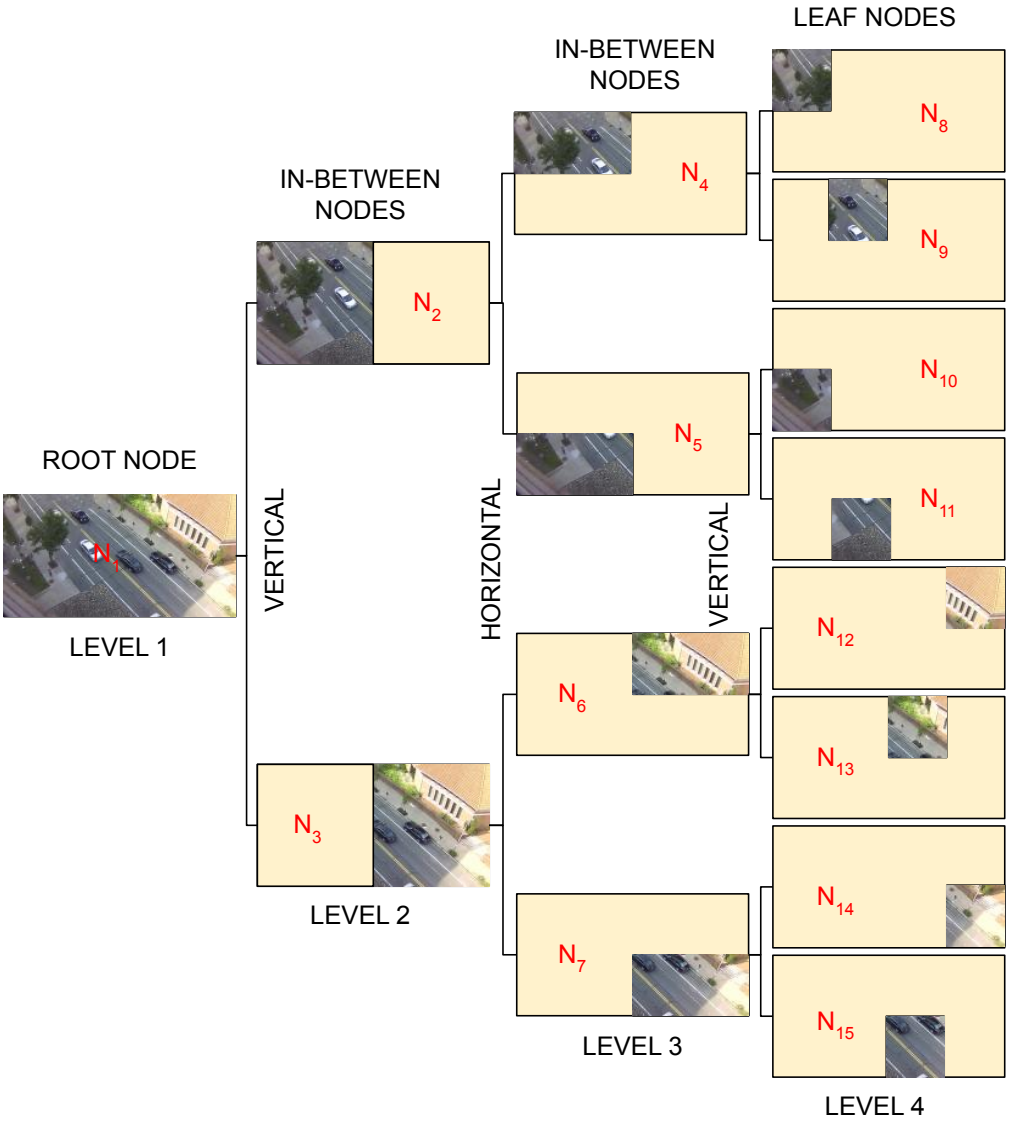
The introduced binary partitioning tree allows our algorithm to generate in total approximately $K = c^{2^D}$ different video frame (coordinate space) partitions (prunings), where $c = 1.50283801...$ [48]. We define the partition P_j of a pruned tree as the collection of the regions of the leaf nodes in that pruning, $P_j = \{R_i : i \in I_j\}$ where I_j is the set of the leaf nodes. Note that for those leaf nodes, $R_i \cap R_m = \emptyset$ for any $i, m \in I_j$ and $\cup_{i=1}^{n} R_i = R_{all}$ and R_{all} is the whole coordinate space, and so each pruning provides a partition spanning R_{all}. The complexity of a partition can be measured by the number of regions, i.e., leaf nodes, it includes. The simplest partition contains only the root node, whereas the most complex partition contains all the leaf nodes of the full tree.

As explained in Section 3.3.1, a partition corresponds to a different piecewise anomaly detection model $\overline{M}_{j,t} = \{M_{i,t-1} : i \in I_j\}$ each of which is the collection/union of the local anomaly detection models ($M_{i,t}$'s) running in its regions, i.e., at the leaf nodes in its pruning.

We point out that our method defined in (3.1) requires $O(K)$ doubly exponential computational complexity for a given arbitrary set of partitions ($K = c^{2^D}$). This can be significantly reduced to only linear $O(D)$ when a structured set of partitions is employed as in the case of context tree [48], which we exploit here for anomaly detection. Although there are K different partitions in the ensemble of pruned trees, one can have only $D << K$ different decisions since partitions share regions. Hence, the result $f(x_t)$ in (3.1) can be obtained by combining only those D different decisions as

$$f_{t-1}(x_t) = \sum_{k \in I_{active(t)}} \tilde{w}_{k,t-1} M_{k,t-1}(x_t), \tag{3.2}$$

where $I_{active}(t)$ contains the id of D nodes that the observation x_t (feature vector of the detected YoLo activity) visits (depending on the activity's position in the frame) from the root $I_{active}(t, 0)$ to the leaf $I_{active}(t, D)$, i.e., $x_t \in R_i, \forall i$, where $i \in I_{active}(t)$.



(a) Tree structure visualized

(b) Vertical/horizontal splits at each level are illustrated.

Figure 3.7 Our partitioning algorithm is visualized above as a binary tree structure, where each node corresponds to a video frame that is split into two parts of equal size either horizontally or vertically at each level of the tree. As we traverse deeper into the tree, smaller patches are obtained, which serve as inputs for the anomaly detection model. By partitioning the frames in this manner, the model can be trained on a more diverse set of training samples, resulting in better precision and accuracy in anomaly detection. The hierarchical structure of the binary tree allows the model to learn and detect anomalies at various levels of granularity, from coarse to fine-grained details.

On the other hand,

$$\tilde{w}_{k,t} = \sum_{j=1}^{K} 1_{\{R_k \in I_j\}} w_{j,t} \tag{3.3}$$

is the accumulation of the weights of the partitions which include R_k as a leaf. Here, $1_{\{.\}}$ is the indicator function which returns 1 if its argument holds, and returns 0 otherwise.

3.3.4 Tree Recursions

The accumulated weights in (3.3) are also computed efficiently based on certain recursions over the tree and proportionally to the partition model performances. We next continue with explaining these recursions and then finally provide the utilized performance metric.

A local performance $\Phi_i(t)$ measurement is first defined for each local anomaly detection model $M_{i,t-1}$ at previous time $t-1$ after making a prediction for the observation x_{t-1}. This local performance can be obtained by accumulating (through multiplication after lifting with an exponential) the instantaneous losses over time via

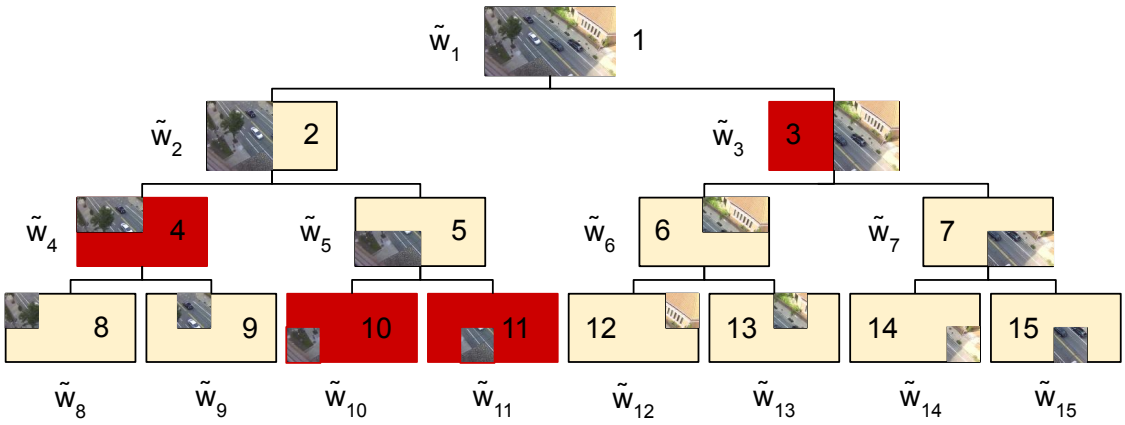
$$\Phi_i(t) = \Phi_i(t-1) \times \exp(-h \times \mathcal{L}(x_t, M_{i,t-1})), \tag{3.4}$$

where h is a constant, and $\mathcal{L}(x_t, M_{i,t-1})$ is the instantaneous loss of the model $M_{i,t-1}$ after predicting for x_{t-1}. The loss we use in this thesis is explained in Section 3.3.7.

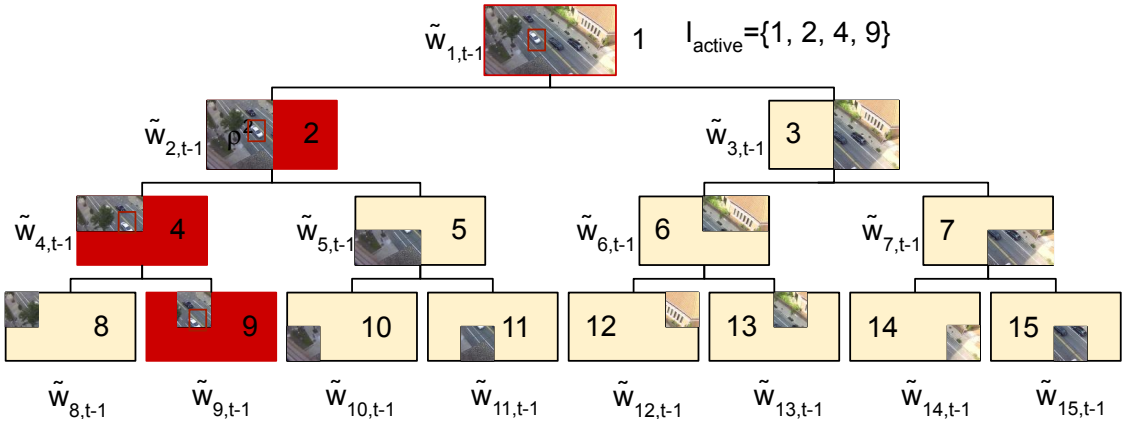
In addition to the local performance, we also define a node performance variable $\mathcal{P}_i(t)$ (illustrated in Fig. 3.8) which can be calculated recursively from leaves to the root as explained in [48]:

$$\begin{aligned} &\mathcal{P}_i(t) = \Phi_i(t), \text{ if node } N_i \text{ is a leaf,} \\ &\mathcal{P}_i(t) = \beta \times \mathcal{P}_{i,left} \times \mathcal{P}_{i,right} \\ &+ (1-\beta) \times \Phi_i(t), \text{ otherwise,} \end{aligned} \tag{3.5}$$

where $i \in I_{active}(t)$, $\mathcal{P}_{i,left}$ is the left child's node performance and $\mathcal{P}_{i,right}$ is the right child's node performance. Here, β is a parameter (split probability) that controls the initial weighting of the partition models concerning their complexities.



(a) Nodes in the calculated pruning are colored in red.



(b) Decision calculated for the new sample x_t (i.e. for the white car inside the red bounding box): $f_{t-1}(x_t) = \sum_{k \in I_{active(t)}} \tilde{w}_{k,t-1} M_{k,t-1}(x_t)$, where $i \in I_{active}$.

Figure 3.8 Whole process for the pruning and obtaining the decision through the active nodes for sample x_t is illustrated above.

Note that when an observation x_t is received, all of the local models ($M_{i,t-1}$) first make their predictions, the instantaneous losses ($\mathcal{L}(x_t, M_{i,t-1})$) are calculated and then all of the local models, and local as well as node performances are updated bottom-up recursively, yielding $M_{i,t}$, $\Phi_i(t+1)$, $\mathcal{P}_i(t+1)$. Once these recursions are completed, one final (this time top-down) pass over the tree is required to obtain the accumulated weights in (3.3) as desired. For this purpose, an auxiliary variable $\mu_{i,t}$ is recursively defined as explained in [48]:

$$\mu_{i,t} = 1 - \beta, \text{ if node } N_i \text{ is the root,}$$
$$\mu_{i,t} = \beta \times \mathcal{P}_{i,sibling}(t) \times \mu_{p,t}, \text{ otherwise,}$$

where $i \in I_{active}(t)$, $\mathcal{P}_{i,sibling}$ is node performance of the sibling node of N_i and N_p is the parent node.

These top-down and bottom-up recursions can be completed with computational complexity $O(D)$ since the number of nodes visited by x_t, i.e., the cardinality of $I_{active}(t)$, is only depth D. With these results, the accumulated weight in (3.3) can be simply calculated as (cd. [48])

$$\tilde{w}_{i,t} = \mu_{i,t} \times (\mathcal{P}_i(t)/\mathcal{P}_{N_1}), \tag{3.6}$$

where N_1 is the root node.

3.3.5 Computational Complexity

Context tree-based methods, like ours, (as explained in [48]) benefit a computational efficiency advantage for calculating the performance of each possible binary tree partition in the context tree. When processing a new feature vector x_t, the algorithm only evaluates the active local anomaly detection models, which are members of $I_{\text{active}(t)}$ - the nodes visited by x_t. This entails that, the algorithm updates D different local anomaly detection models in each iteration, where D is the depth of the context tree, i.e., the cardinality of $I_{\text{active}(t)}$.

The key advantage of this approach is that it drastically reduces the number of local models that need to be evaluated when calculating the performance of each partition. Instead of evaluating all possible context tree partitions, which would lead to a total of $K = 1.50283801...^{2^D}$ partitions, the algorithm only considers the

active nodes (D local anomaly detection models) in the context tree. This reduction in the number of models to evaluate results in significant computational efficiency, making the method suitable for handling complex models.

As a result, while introducing locational anomaly detection and enhancing the overall performance of existing anomaly models, we only need to run D different local anomaly detection models, each corresponding to a node at a different depth in the context tree. The independence of each model allows us to calculate their results in parallel, further enhancing the efficiency of the process.

Our proposed local anomaly detection method (introduced in Section 3.3.6) computes the congruency with the anomaly model for each new feature vector and updates the anomaly model parameters using Welford's online algorithm [267]. Both of these operations depend solely on the feature dimensionality. Consequently, the computational complexity for each model in our anomaly detection method is $O(\text{dim})$ per single feature vector.

As emphasized before, we need to calculate the local anomaly model results D times for each new feature vector. Therefore, the overall complexity of our local anomaly detection model is $O(D \times \text{dim})$ for processing a single feature vector. Considering N new feature vectors, the overall complexity of our anomaly detection method becomes $O(N \times D \times \text{dim})$. This linear (w.r.t. the incorporated local model) complexity allows our anomaly detection algorithm to achieve real-time performance.

Furthermore, it's worth noting that our algorithm does not introduce any additional computational complexity for certain phases, such as object and motion detection networks that run on the whole image and need to be executed only once for the integrated methods. As mentioned in the computational complexity analysis in references [171; 172], the integrated approaches, with the use of capable graphical processing units and real-time motion (Flownet2 [5]) and object detection (YoLo [4]), can efficiently process data and generate new features in real-time.

By introducing location-aware anomaly detection and utilizing a smaller number of features, we achieve higher performance while effectively reducing the computational burden. This allows us to maintain accuracy while running the algorithm more efficiently.

3.3.6 Local Anomaly Detection Model

In this section, we explain our local anomaly detection model $M_{i,t}$ that we use to produce (by combining those of an active set of nodes) our final anomaly detection decision as stated in (3.2).

In the first step, we calculate the Mahalanobis distance [268] of the new sample x_t to the mean $\theta_{i,t-1}$ of the previous samples in the node N_i as

$$D_i(t) = \sqrt{(x_t - \theta_{i,t-1})^T \Sigma_{i,t-1}^{-1} (x_t - \theta_{i,t-1})}, \tag{3.7}$$

where $\Sigma_{i,t-1}$ is the covariance and $i \in I_{active}(t)$. In the second step, we calculate the corresponding p-value as

$$p_{value,i,t} = \mathcal{F}(D_i(t)^2, dim), \tag{3.8}$$

where $\mathcal{F}$ is the cumulative distribution function of the χ^2 (chi-squared) distribution ("dim" stands for the feature dimension). In the last step, the node decision $M_{i,t}(x_t)$ is calculated as:

$$M_{i,t-1}(x_t) = 1, \text{ if } p_{value,i,t} > 1 - \tau,$$
$$M_{i,t-1}(x_t) = 0, \text{ if } p_{value,i,t} \leq \tau,$$

where $\tau \in \{0,1\}$ is a desired false alarm rate threshold, and $\{0,1\}$ are labels for normal and anomalous samples respectively. Note that this describes an optimal (in the NP detection sense) anomaly detection model that operates at a given desired false alarm rate τ if the distribution is Gaussian. It can be straightforwardly extended by using a nonparametric density estimation approach to target more complex non-Gaussian situations. In addition, one can also make a soft decision by directly applying the continuous $p_{\text{value},i,t}$ as the local decision $M_{i,t-1}(x_t)$. During our experiments, we observed that soft decisions produce better results. From this point on, we use $M_{i,t-1}(x_t) = p_{\text{value},i,t}$.

Below, we summarize the execution cycle of our context tree-based anomaly detection approach for each new incoming sample x_t as seen in Fig. 3.8.

- A new sample (YoLo activity) $x_t = [x_{t,1}, \cdots, x_{t,dim}]$ arrives on the tree, where dim is the number of features.
- For each node N_i that the sample visits, i.e., $i \in I_{active}(t) = \{1,2,4,9\}$ in Fig.

3.8, we calculate the local decisions $M_{i,t-1}(x_t)$.

- Then we calculate our overall soft decision $f_{t-1}(x_t)$ as given in (3.2). Based on that, the label prediction can be made after thresholding and checking the sign, i.e., $\hat{y}_t = \text{sign}(f_{t-1}(x_t) - \eta)$. Here, η is a threshold used for obtaining the ROC (receiver operating characteristics) curve in our experiments.

- The local models (means and covariance matrices) are updated online with the sample x_t using Welford's online algorithm [267]. Then, we calculate our loss (defined in the section below) for each node $i \in I_{\text{active}}(t)$. The recursive updates of means and covariances:

$$\theta_{i,t} = \frac{\mathcal{N}_{i,t-1} \times \theta_{i,t-1} + x_t}{\mathcal{N}_{i,t-1} + 1},$$

$$\hat{x}_t = (x_t - \theta_{i,t})(x_t - \theta_{i,t})^T,$$

  where $\hat{x}_t$ is a temporary attribute to be used in the covariance update as

$$\Sigma_{i,t} = \frac{\mathcal{N}_{i,t-1} \times \Sigma_{i,t-1} + \hat{x}_t}{\mathcal{N}_{i,t-1} + 1}$$

  and

$$\mathcal{N}_{i,t} = \mathcal{N}_{i,t-1} + 1.$$

  Above, $\mathcal{N}_{i,t-1}$ is the number of observations made until time $t-1$.

- In the last step, the tree variables are updated online with the sample x_t using the described recursions in Section 3.3.4.

Note that as there might be multiple YoLo activities in each frame F_t, we observe a corresponding set $\mathcal{X}_t$ of new samples at each time t. Hence, we run our algorithm for every sample $x_t \in \mathcal{X}_t$ separately.

3.3.7 Local Anomaly Detection Loss

One important aspect of our algorithm is that it is an unsupervised technique since we do not use label information. Namely, in the training phase, we assume that all of the samples are normal as anomalies are typically extremely rare. This prevents the use of $0-1$ loss for measuring the performance of our local models, which has a critical importance in our tree recursions and weight assignments. Therefore, we

are required to measure the performance in an unsupervised manner without using any label information.

To that end, we propose a novel loss function for our local anomaly detection models. Given a sample x_t, the loss of any non-leaf node N_i in $I_{\text{active}}(t)$ is derived from the bias, i.e., difference, between its (non-leaf) true mean $\bar{\theta}_i$ and the true mean of the corresponding leaf node $\bar{\theta}_{i_1}$ in $I_{\text{active}}(t)$. This bias $\bar{\theta}_i - \bar{\theta}_{i_1}$ is unknown but could be straightforwardly estimated by $B_{i,t} = \theta_{i,t} - \theta_{i_1,t} \sim N(\bar{\theta}_i - \bar{\theta}_{i_1}, V_{i,t})$ for non-leaf nodes and $V_{i,t} = \frac{\Sigma_{i,t}}{\mathcal{N}_{i,t}} + \frac{\Sigma_{i_1,t}}{\mathcal{N}_{i_l,t}}$, if the two sample (data of the non-leaf and leaf) were independent. However, since the two sample certainly overlap, we need to take care of the intersection and subtract it from the sample of the non-leaf in the calculations. Therefore, letting

$$\hat{\theta}_{i,t} = \frac{\mathcal{N}_{i,t}\theta_{i,t} - \mathcal{N}_{i_l,t}\theta_{i_l,t}}{\mathcal{N}_{i,t} - \mathcal{N}_{i_l,t}},$$

then $B_{i,t} = \hat{\theta}_{i,t} - \theta_{i_1,t}$. Similarly, letting

$$\hat{V}_{i,t} = \frac{\mathcal{N}_{i,t}(\Sigma_{i,t} + \theta_{i,t}\theta_{i,t}^T) - \mathcal{N}_{i_l,t}(\Sigma_{i_l,t} + \theta_{i_l,t}\theta_{i_l,t}^T)}{N_{i,t} - \mathcal{N}_{i_l,t}}$$

yields

$$V_{i,t} = \frac{\hat{V}_{i,t} - \hat{\theta}_{i,t}\hat{\theta}_{i,t}^T}{\mathcal{N}_{i,t} - \mathcal{N}_{i_l,t}} + \frac{\Sigma_{i_l,t}}{\mathcal{N}_{i_l,t}}.$$

Based on the above derivations, we define the instantaneous loss $\mathcal{L}(x_t, M_{i,t-1})$ in (3.4) as

$$\mathcal{L}(x_t, M_{i,t-1}) = \mathcal{F}(B_{i,t}^T V_{i,t}^{-1} B_{i,t})$$

for a non-leaf node N_i, which computes the Mahalanobis distance between 0 (no bias) and the observed bias $B_{i,t}$ under $V_{i,t}^{-1}$ and normalizes it to $[0,1]$ by using its quantile through the chi-squared cumulative distribution function $\mathcal{F}$. This distance is uniformly distributed in the long run if $B_{i,t} \to 0$. In this case of no bias, $B_{i,t} = \theta_{i,t} - \theta_{i_1,t} \to 0$, we infer that the local statistics of N_i is similar to the corresponding leaf node, and we desire no loss discrimination between this non-leaf N_i and N_{i_1}. To incorporate that, we define a randomized loss for the leaf node as

$$\mathcal{L}(x_t, M_{i,t-1}) = U$$

for a leaf node N_i and U is uniformly distributed in $[0,1]$.

If we observe non-zero bias, $B_{i,t} \neq 0$ in the long run, the loss is convergent to 1, i.e., $\mathcal{L}(x_t, M_{i,t-1}) \to 1$ since $B_{i,t}$ is bounded away from 0 and so we infer that the local

statistics is different than that of the leaf.

Consequently, if there is non-zero bias, i.e., $B_{i,t} \neq 0$, compared to the leaf (dissimilar statistics), then the weight $\tilde{w}_{i,t}$ of $M_{i,t}$ in (3.2) will converge to 0 since the loss is convergent to 1. Hence, our final anomaly detection asymptotically benefits more from the descendent nodes, particularly the leaf N_{i_1} as it compensates for the bias (by having no bias) in the ascendant node N_i. The situation is reverse in the transient phase, meaning our final anomaly detection benefits more from the ascendant node N_i. This is because, despite the non-zero bias, the ascendant node N_i has higher precision (lower variance) since it typically observes more samples, i.e., $\mathcal{N}_{i,t} > \mathcal{N}_{i_1,t}$. This leads to better anomaly localization in the long run while exploiting the global perspective in the beginning. On the other hand, if the bias is zero, i.e., $B_{i,t} = 0$ (similar statistics between the non-leaf N_i and N_{i_1}), then both of the local models $M_{i,t}$ and $M_{i_1,t}$ are expected to perform asymptotically equally well. However, the local model $M_{i,t}$ of the non-leaf will always have a better precision. Hence our final anomaly detection benefits more from it since we assign a larger weight to it a priori with $\beta < 1$ (cf. (3.5), in which a typical choice is $\beta = 0.5$), grouping regions of similar statistics as desired. Finally, by inspecting the weights $w_{j,t}$'s in (3.1), one can obtain the best statistical grouping (owing to the theoretical guarantees established in [47; 48]), i.e., the best context tree partitioning of the scene, as a result of the introduced loss function here that manages the bias-variance trade-off and measures the performance in an unsupervised manner.

3.4 Experiments

In the following, we start with our simulations and then continue with the real data experiments where we also compare two different feature sets, i.e., descriptors compressed with AE [7] and hand-crafted HOF features as well as motion statistics [6]. We close this section by presenting results for supervised anomaly detection, where the local anomaly detection model in CTBAD is replaced with a Neyman-Pearson (NP) classifier. Since this is a supervised setting, we use a simple 0-1 loss function instead of the novel loss employed in the unsupervised approach.

In order to demonstrate the effectiveness of CTBAD, we conduct experiments using both simulated and real datasets. To provide a clear understanding of our experimental approach, we outline a simple flow:

- For each new feature, whether obtained from real or simulated scenario data, we employ CTBAD with different split probabilities, $\beta \in [0.125, 0.25, 0.375, 0.5, 0.675, 0.75, 0.875]$. Additionally, we compare the decision outcomes with those obtained from both the root node model and the leaf node models.
- To observe the gradual improvement of CTBAD performance, we analyze its performance with increasing numbers of training samples. We periodically halt the training process and evaluate the performance of the method at that particular point in time. To do this, we utilize test samples and calculate the AUC at each point. This analysis allows us to understand the impact of adjusting split probabilities and observe the progressive performance improvement of CTBAD over time.

3.4.1 Simulations

We have observed that the datasets used for anomaly detection (UCSD [9], ShangaiTech [11], Avenue [10], ...) usually cover the trivial case where anomalies are defined as anomalies for the whole scene and normal behavior in one part of the scene is normal behavior throughout the scene. For this reason, we are not able to show one of the key features of our algorithm, which is its ability to apply different models to different locations in the scene, so can be utilized for differentiating between different anomalies at different places, thus labeling the same object as normal in one part of the scene and an anomaly in the other. This means we are not able to show the full potential of our algorithm with the current datasets available. For this purpose, as an initial step as a proof of concept, we decide to work with simulated data as a proof of concept.

In order to achieve that, we developed a simulation framework, in which we can define different actors (normal and abnormal) in different locations, in a frame with different statistics representing them. As this is a proof of concept, and we want to show the effectiveness of our algorithm, we define a simple test scenario that divides the scene into two separate parts each representing two main actors, vehicles, and pedestrians. Before explaining the results, we want to explain some of the rules for the scenario and its actors in detail:

- The Motorway is out of limits for pedestrians, so any actor with pedestrian statistics shall not occur in the motorway.

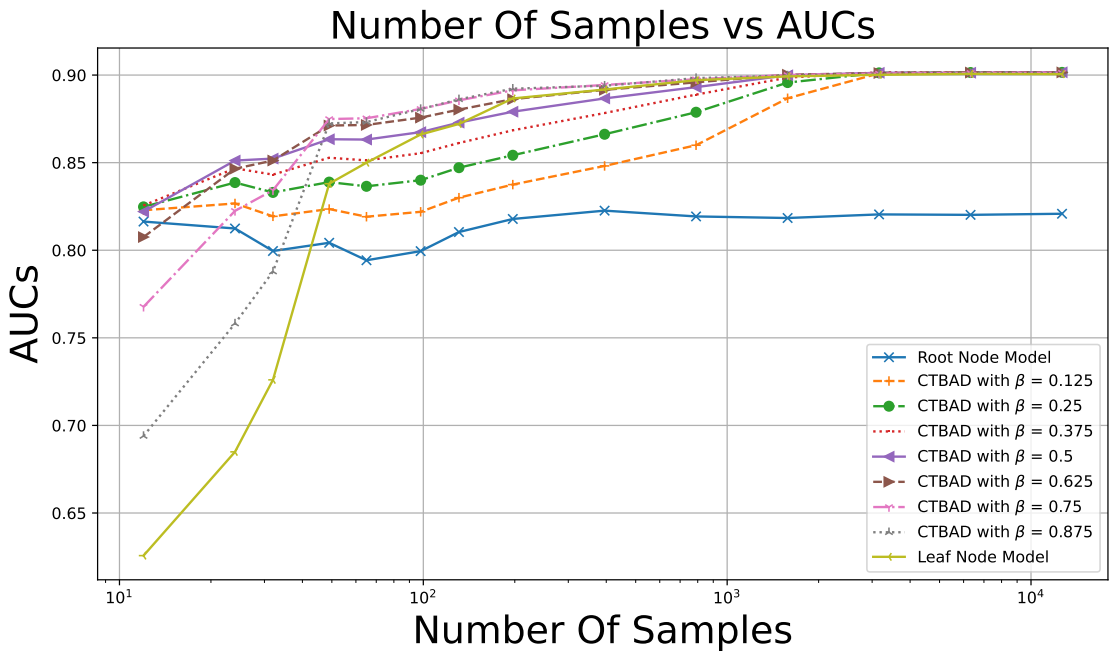


Figure 3.9 AUC vs number of training samples for the simulation data

- There shall be no vehicles on the sidewalk.
- Cyclists, equestrians, etc.. are modeled as abnormal pedestrians and vehicles to show that in addition to separate abnormalities, shared abnormalities can be detected as well.

It is worth noting that while the scenario we created for this proof of concept is relatively simple, real-world data may present additional challenges, such as complex and dynamic scenes, varying environmental conditions, and unpredictable anomalies. Therefore, our framework can be extended to create more complex scenarios that better represent real-world anomalies and evaluate the performance of our algorithm in more challenging conditions.

We assumed for our scenario that each actor (pedestrian, vehicle, cyclists, equestrians, etc...) in each part of the frame is modeled by using separate multivariate Gaussian distributions (MGD). By this, we mean that our simulation creates the result of a feature generation network and feeds them to the abnormality detection framework. Both a vehicle and a person have been modeled as a feature having a fixed dimension d sampled from a fixed MGD such as $x\ N(\underline{0}, \sigma^2)$.

For separate actors in our tests, we have used the following distributions :

$\hat{x} \sim \mathbf{N}(0, \sigma^2 I)$, for normal pedestrians

$\hat{x} \sim \mathbf{N}(0, z^2\sigma^2 I)$

, for abnormal pedestrians (cyclists, equestrians, etc...)

where $z \geq 1$

$\hat{x} \sim \mathbf{N}(\alpha, \sigma^2 I)$

, for normal vehicles

$\hat{x} \sim \mathbf{N}(\alpha, z^2\sigma^2 I)$

, for abnormal vehicles (cyclists, equestrians, etc...)

where $z \geq 1$

The reason we select to use different covariance matrices for normal and abnormal behavior is to put on the real-life behavior expected of feature generation networks. It will not be possible to have a hundred percent detection rate and we want to simulate that behavior as well. In the next section, we share the results and explain them in detail for simulation data.

We compare the results of our algorithm to the two extreme cases. The first one is where no partitioning to the sample space is applied, which we name the root node model. The second one is where the sample space is partitioned into the smallest partition (leave nodes in our case) available, and it is called the leaf node model. Both these cases provide the basic partitioning options that can be computed and provide a basis to show the advantages of our algorithm. The reason for comparing these two cases is to show the advantages of using a root node model that learns the overall statistics faster since all incoming samples affect the performance, and in the case of the leaf node model learning precise models as the training continues. In addition, we want to prove how our algorithm takes advantage of both partitions.

We want to explain the figures we utilize to represent our results in detail. We argued that using our algorithm decreases the total number of training samples to reach top performance compared to piece-wise models (instead of using a tree structure, use the models from the leaf nodes only), and another argument we claim is that for a small number of training samples, the performance is at least comparable to model using the whole sample space or as we call it the root node model. To prove this, we take snapshots during the training process of our models and compare the effects of increasing the number of training samples on the performance of both

piece-wise and root node models. At each snapshot, we calculate the ROC [269] for changing $\beta \in \{0.125, 0.25, 0.375, 0.5, 0.525, 0.555, 0.875\}$, for increasing number of training samples that are a proportion to the total number of training samples available. From these ROCs, we compute the area under curve (AUC) [269] for each β for each ROC calculated at each training sample. We illustrate the AUC progress as the number of training samples increases. As you can see, the aim is to show that for a smaller number of training samples, the AUC for models except the leaf node model, shall be higher whereas, for models with a larger number of training samples, all tree models shall converge to the piece-wise model except the root node model. One additional observation is that the performance of the root node model shall deteriorate as the number of training samples increases because the statistics for the trained model become the added statistics from two different MGD and its model shall diverge from the actual normal model for both parts of the scene.

In Fig. 3.10, we provide the AUC results of our simulations. As explained, the tests are executed with different numbers of training samples from the total training sample set with around 12K samples. As observed in Fig. 3.10, for models with a lower number of training samples, the tree algorithm performs much better than the piece-wise model, whereas up from a certain number of training samples the performance of the piece-wise model catches up with the tree model and they all converge to the same performance. This is the expected behavior from our algorithm and the simulation proves that our algorithm outperforms every simple model with the correct distribution of actors. However, as explained before the uniform distribution of expected behavior throughout the scene in the datasets such as UCSD causes us not to observe the full potential of our algorithm since the root node model already covers one of the most important aspects of our algorithm, to overcome the slow start problem for training models since all incoming samples become a part of the model.

Another important aspect that we want to emphasize is that we ensure enough training samples are observed for each leaf node throughout the simulation scenario. With this, we establish a working model for each node and the results from that node become dependable. However, we need to posit that this sometimes is not the case in real-life data since no sample can be observed for some parts of the frame. To overcome no sample problem, as explained we decide to assign a random score and we can observe the results of this decision for the piece-wise model.

For a detailed look, at the performance of the tree algorithm for different β values, which controls the rate of tree saturation to the leaf nodes. Here a higher value for β, means a faster saturation to the leaf, and as expected, this reflects on the

performance of our algorithm. With a small training sample set, β values smaller than 0.5 performs better, and for larger training sample sets, the opposite is true as seen in Fig. 3.10.

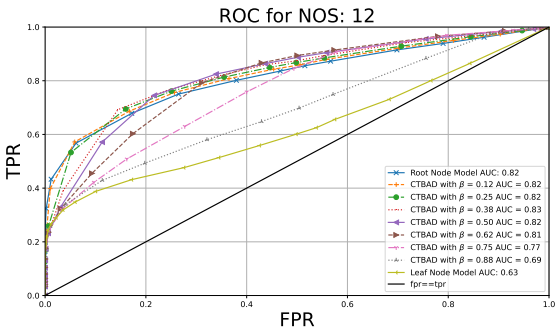
We have observed that commonly used datasets for anomaly detection, such as UCSD [9], ShanghaiTech [11], and Avenue [10], typically focus on detecting anomalies that affect the entire scene uniformly. These datasets do not allow us to showcase one of the key features of our algorithm, which is its ability to apply different models to different locations in a scene. Our algorithm can differentiate between different anomalies at different places, even labeling the same object as normal in one part of the scene and an anomaly in another.

Unfortunately, the current datasets available have limitations, and we are unable to demonstrate the full potential of our algorithm. Therefore, as an initial proof of concept, we decided to generate and work with simulated data. This approach allows us to create a range of scenarios where anomalies can occur in different parts of the scene, enabling us to highlight the strengths of our algorithm in a controlled environment. Simulated data allows us to define various actors, including both normal and abnormal, at different locations within the scene with different statistical representations.

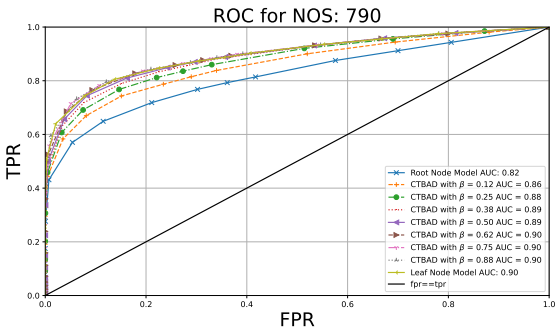
As a proof of concept, we define a simple simulation scenario that divides the scene into two distinct parts, each representing two primary actors/regions, vehicles/motorways and pedestrians/sidewalks. Before presenting the results, we explain two rules we employ for this scenario and its actors.

- Pedestrians (vehicles) are not allowed on the motorway (sidewalk), hence any pedestrian (vehicle) on a motorway (sidewalk) is a locational anomaly.
- Any pedestrian (vehicle) whose statistics are deviant regardless of its location is an indigenous anomaly.

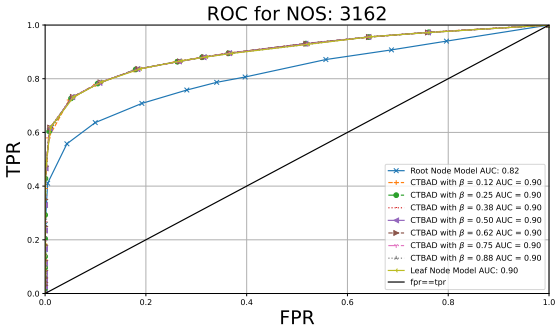
With these rules, we consider a scenario that mimics real-world situations where anomalies can occur in different parts of a scene, and evaluate our algorithm's performance in detecting both locational and indigenous anomalies. Each actor (pedestrian or vehicle) in the scene is modeled using a separate multivariate Gaussian distribution (MGD). Specifically, we generate d-dimensional features for each actor and feed them into our anomaly detection framework. Assumed MGDs for feature vectors of the normal pedestrians: $x \sim N(0, \sigma^2 I)$, anomalous pedestrians (such as cyclists): $x \sim N(0, z^2\sigma^2 I)$, normal vehicles: $x \sim N(\alpha, \sigma^2 I)$, and anomalous vehicles: $x \sim N(\alpha, z^2\sigma^2 I)$, where $z > 1$. These are indigenous anomalies, and note that any pedestrian (vehicle) on the motorway (sidewalk) is another type (locational) of



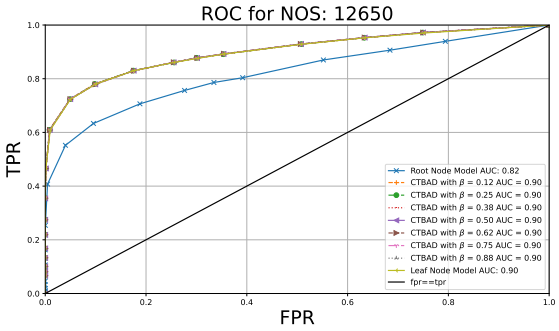
(a) ROC with 12 samples for training the overall anomaly detection models



(b) ROC with 790 samples for training the overall anomaly detection models



(c) ROC with 3162 samples for training the overall anomaly detection models



(d) ROC with 12650 samples for training the overall anomaly detection models

Figure 3.10 ROCs with increasing number of training samples for the simulation data

anomaly.

To simulate separate actors in our tests, we used different distributions. We modeled normal pedestrians using $\hat{x} \sim N(\overline{0}, \sigma^2 I)$, while abnormal pedestrians, such as cyclists and equestrians, were modeled using $\hat{x} \sim N(\overline{0}, z^2\sigma^2 I)$, where $z \geq 1$. Similarly, we modeled normal vehicles using $\hat{x} \sim N(\overline{\alpha}, \sigma^2 I)$ and abnormal vehicles using $\hat{x} \sim N(\overline{\alpha}, z^2\sigma^2 I)$, where $z \geq 1$. We used different covariance matrices for normal and abnormal behaviors to simulate the expected behavior of feature generation networks in real-life scenarios. We acknowledge that achieving a 100% detection rate is not possible, and we aimed to simulate this behavior as well.

In this framework, we compare our CTBAD algorithm (with changing split probability β values) to two extreme partitioning methods, namely the root node model and the leaf node model. We aim to highlight the advantages of using a root node model (observing more samples) for faster learning of overall statistics (higher precision in the beginning at the cost of a higher bias) and a leaf node model (observing fewer samples) for better localization and higher performance (lower bias) in the long run (at the cost of higher variance in the beginning) models as training pro-

gresses. Fig. 3.9 depicts the AUC results of our simulations, where we observe the effect of increasing the number of training samples on the performance by using snapshots taken during the training process and testing repeatedly on the separate test set. At each snapshot, we calculate the AUC of the ROC for increasing numbers of training samples as well as for changing β values (split probability) which applies a priori weighting on the partition models. Note that higher β puts higher weights on deeper nodes and $\beta = 0$ corresponds to the root model whereas $\beta = 1$ corresponds to the leaf node model, cf. (3.5). Fig. 3.9 shows that the root node ($\beta = 0$) model performs well in the beginning but suffers from bias in the long run. The leaf node model ($\beta = 1$) suffers from high variance in the beginning but performs well in the long run. Whereas our tree algorithm (with varying β's) outperforms both the root node and leaf node models in terms of accuracy, demonstrating the efficacy of combining different partitions. We benefit from the root model's higher precision in the transient phase and from the leaf node model's lower bias in the long run. Note that, as expected, with sufficiently large data, our tree algorithm and the leaf node model converge to the same performance level, which shows that our algorithm has no bias and achieves appropriate anomaly localization in the long run. Fig. 3.10 presents the corresponding ROC curves to describe the behavior of detection power against the false alarm rates. Note the convergence among all the models (except for the root with $\beta = 0$) as data size increases, and also note our superiority in the course of convergence. Experimentally, $\beta = 0.5$ appears to provide the best version of our algorithm, yielding a highly superior performance in both low and high data regimes thanks to the introduced bias-variance trade-off.

3.4.2 Datasets

To test the effectiveness of our methods, we utilize four distinct datasets: UCSD Pedestrian [9], Avenue [10], Shanghai [12], and Street Scene [12].

The UCSD Pedestrian dataset consists of footage from a camera observing a pedestrian path where traffic flows both left-to-right and right-to-left. This dataset captures two types of abnormal behavior: (1) vehicles such as bicycles or small golf carts on the pedestrian path, and (2) pedestrians running.

The Avenue dataset captures scenes in front of a train station on a campus. In this dataset, anomalies include unusual activities such as throwing papers, which are challenging to detect using conventional features.

Figure 3.11 A cyclist is detected as an anomaly in the UCSD Pedestrian dataset. This anomaly is not location-specific because there is no designated cycling lane in the scene.

The Shanghai dataset includes footage from multiple camera angles capturing complex scenes involving pedestrian and vehicle traffic on a campus. The StreetScene dataset offers a bird's-eye view of a complex local road scene with both vehicle and pedestrian traffic. In these datasets, anomalies include activities like jaywalking and unauthorized vehicle traffic.

We use two types of feature sets for evaluation: (1) descriptors compressed with an AE [7] and (2) flow features (HOF) [6]. Table 3.1 provides an overview of these datasets, which can be categorized based on the nature of the anomalies.

The first category consists of datasets like UCSD Pedestrian, as seen in Fig. 3.11, and Avenue, as seen in Fig. 3.12, where anomalies are spatially stationary. In these datasets, anomalies can occur anywhere in the scene with similar statistical characteristics, independent of location. Therefore, anomaly labeling does not rely on specific locations.

The second category includes datasets like Shanghai, as seen in Fig. 3.13, and StreetScene , as seen in Fig. 3.14, which exhibit locational anomalies. In these datasets, anomalies are defined by location-specific statistics. Therefore, an activity may be labeled as normal or abnormal depending on its location, requiring algorithms to be location-aware for accurate anomaly detection.

As anticipated, utilizing global statistics (root node model, $\beta = 0$) for the entire scene proves to be a viable approach for detecting anomalies in datasets with only indigenous anomalies, such as UCSD Pedestrian dataset (cf. Fig. 3.15). This approach yields decent performance without experiencing any issues in the steady

Figure 3.12 An example from the Avenue dataset depicting a person running in a train station. This behavior is considered anomalous, as individuals typically walk or wait in this environment.

Figure 3.13 A cyclist detected on the pedestrian path in the Shanghai dataset. Since this path is designated for pedestrians, the cyclist's presence constitutes a locational anomaly.

phase. Consequently, we can observe that the performance of our context tree approach is not significantly better than that of the root node model. In fact, as the parameter β decreases (weight of partitions with higher depth), the performance of our context tree consistently improves and approaches to that of the root model. This can be attributed to that generating multiple partition models with varying complexities and combining them only provides minimal gains, as the simplest root

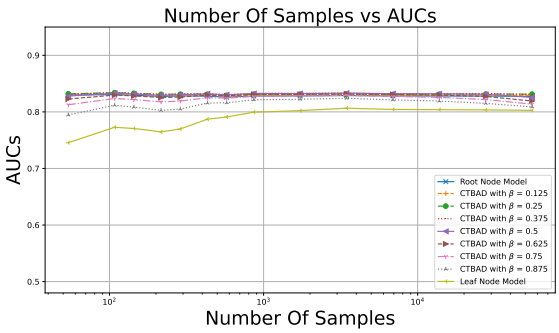
Figure 3.14 A jaywalker crossing the road in the StreetScene dataset is considered a locational anomaly. In contrast, walking on the sidewalk is classified as normal behavior.

Table 3.1 Number of training and test frames in the datasets

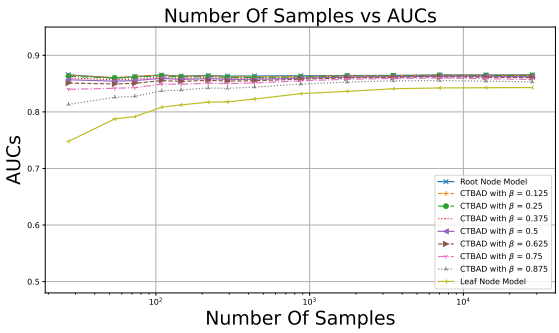
Dataset	Train	Test	Total
UCSD [9]	2,550	2,010	4,560
Avenue [10]	15,324	15,328	30,652
Shanghai [11]	274,515	42,883	317,398
StreetScene [12]	56,893	146,482	203375

model is already suitable due to the non-stationary statistics present in the dataset. On the other hand, it is clear that the AE features are outperformed by the hand-crafted flow features (upper row vs bottom row in Fig. 3.15), meaning that the simple motion attributes (HOF or motion statistics) are sufficient for the detections and more important compared to texture. This is probably because the UCSD Pedestrian dataset is one of the earlier datasets and it does not require complex features. We present an example of an anomaly detected using our algorithm in the UCSD dataset in Fig. 3.11.

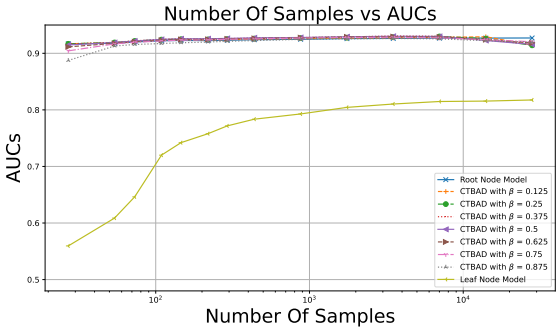
In our experiments with UCSD Pedestrian dataset, our algorithm is generally observed to produce a detection performance that is on par with the steady-state results reported in the literature. However, here we introduce a novel capability of bias-variance trade-off that enables us to outperform in the low data regime. We also emphasize that our proposal is a framework that can operate with any anomaly detection technique, since we generate a context tree-based hierarchical ensemble consisting of instances of the same technique at varying complexities and then combine them for superior performance.



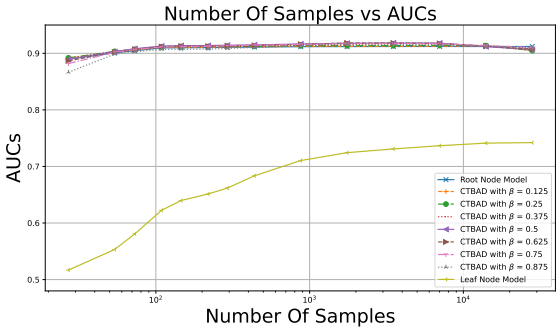
(a) AUC vs number of training samples with regular images [7]



(b) AUC vs number of training samples with dynamic images [7]



(c) AUC vs number of training samples with motion statistics [6]

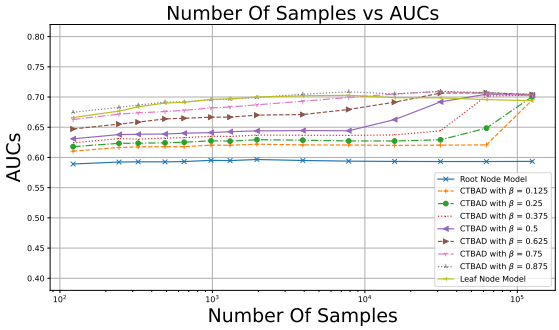


(d) AUC vs number of training samples with HOF features [6]

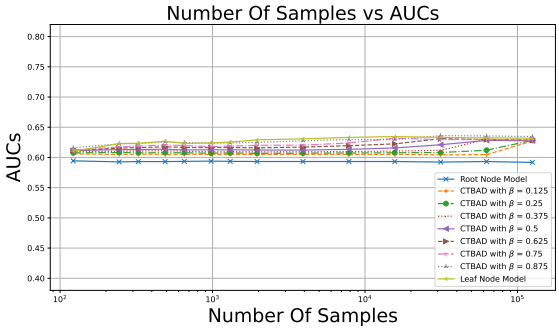
Figure 3.15 AUC vs number of training samples with different feature descriptors for UCSD Pedestrian dataset [9]

When evaluating the Avenue dataset (Fig. 3.16), we encounter challenges in performance for both feature sets, primarily due to the limitations of the employed object detection methods. Specifically, these methods struggle to detect certain objects defined in this dataset, such as papers being thrown in the middle of a scene. Consequently, the performance of all features falls below other datasets. Although the feature extraction methods exhibit sub-optimal performance, we present a successful example of an anomaly detected in the Avenue dataset in Fig. 3.12.

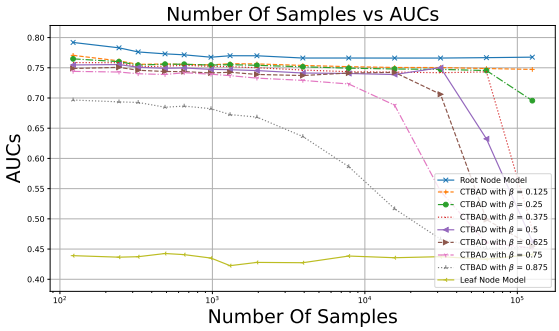
Remark: One important observation in certain cases of both UCSD Pedestrian and Avenue datasets stems from the steady state performance of the leaf node model that is lower than the root node model. This is contradictory to our initial expectations since the leaf node model is the most complex with the lowest bias and it should outperform all the others in the long run. When closely investigated, we figure out that in the training phase, the regions of our tree are not fully populated and some observe no samples. Thus, some of the local models are not well-trained, and then in the test, if a sample drops there, the detections turn out poor. If the training size was large enough, then this certainly would not be the case and results would meet our expectations. Indeed, in our simulations, every node is populated and the



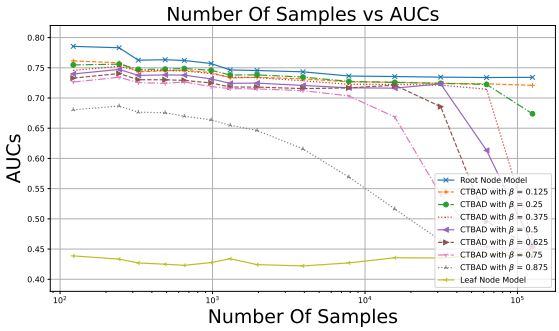
(a) AUC vs number of training samples with regular images [7]



(b) AUC vs number of training samples with dynamic images [7]

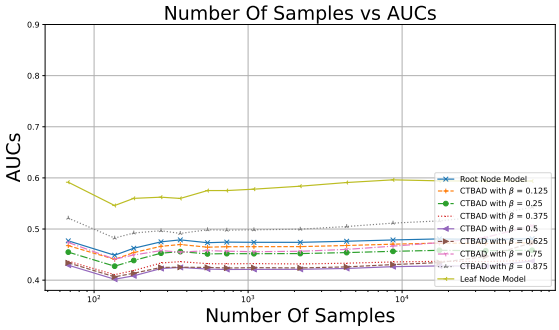


(c) AUC vs number of training samples with motion statistics [6]

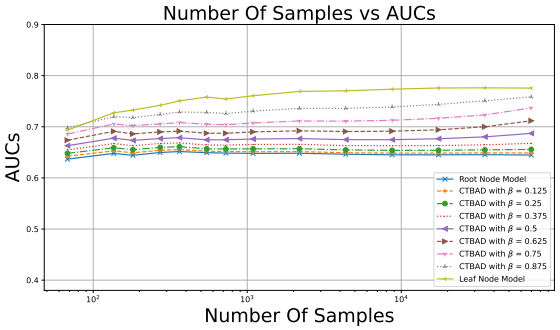


(d) AUC vs number of training samples with HOF features [6]

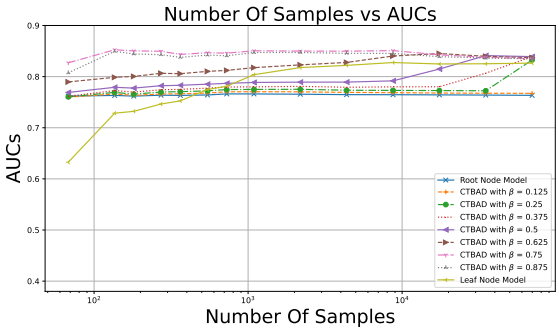
Figure 3.16 AUC vs number of training samples with different feature descriptors for Avenue dataset [10]



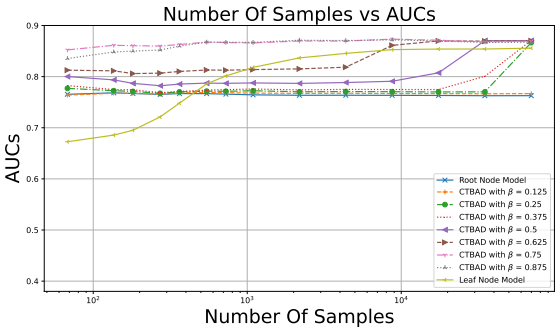
(a) AUC vs number of training samples with regular images [7]



(b) AUC vs number of training samples with dynamic images [7]

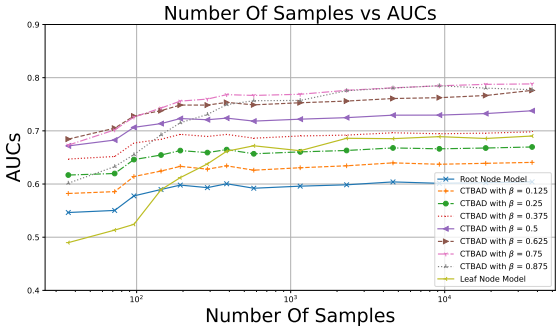


(c) AUC vs number of training samples using motion statistics [6]

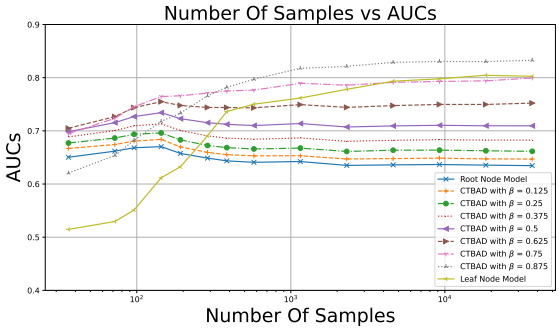


(d) AUC vs number of training samples with HOF features [6]

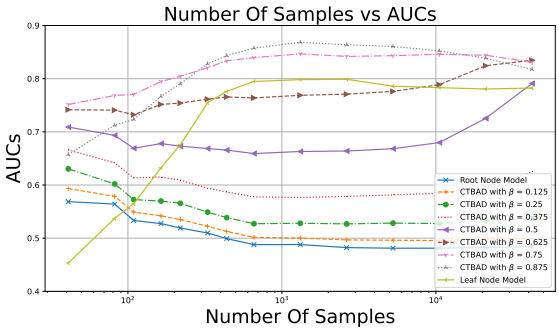
Figure 3.17 AUC vs number of training samples with different feature descriptors for Shanghai dataset [11]



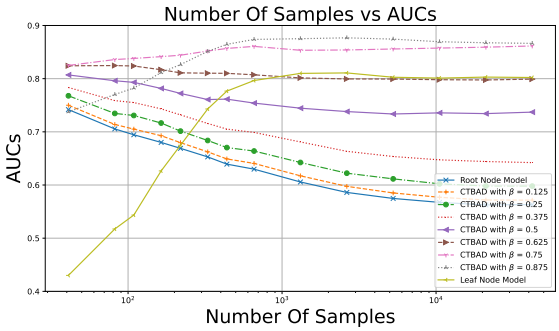
(a) AUC vs number of training samples with regular images [7]



(b) AUC vs number of training samples with dynamic images [7]



(c) AUC vs number of training samples with motion statistics [6]



(d) AUC vs number of training samples with HOF features [6]

Figure 3.18 AUC vs number of training samples with different feature sets for StreetScene dataset [12]

leaf node model is the best performing in the long run. This issue can be mitigated by using the nearest populated node model for samples dropping in unpopulated nodes, which we leave as a future work. Alternatively, one can just choose a small enough β that is readily available in our framework.

Particularly, our context tree-based ensemble of partitions always converges, in all datasets, to the leaf node model. This only reinforces our loss function design which puts more and more weight on the deeper nodes (hence the leaf node model) as the corresponding models get more and more precise, explaining this expected behavior.

In our further experiments with Shanghai and StreetScene datasets (Fig. 3.17 and 3.18), we specifically focus on detecting anomalies with nonstationary spatial statistics. For a demonstration, we identify video segments that are aligned with our use case, e.g., videos containing cyclists (Fig. 3.13) or jaywalkers (Fig. 3.14). Our findings are similar to these datasets, and so we concentrate on the Shanghai dataset in the following as a showcase.

In the Shanghai dataset (Fig. 3.17), the training set predominantly covers only one part of the image, with only a few samples observed on other parts. However, during testing, anomalous activities occur on the less well-covered side. Hence, the above remark is again in effect and the leaf node model again does not reach its maximum potential. However, unlike UCSD Pedestrian and Avenue datasets, our algorithm now fully demonstrates its capability of detecting anomalies even in areas of the image with limited or no training data. The reason behind this contrast is that the anomalies in UCSD Pedestrian and Avenue are typically spatially stationary whereas in Shangai and StreetScene, they are non-stationary. Hence, the leaf node model is the best performing in the steady state and our algorithm successfully tunes to it as time progresses. Notice how the leaf node model starts performing poorly and gradually outperforms the root node model. This highlights the effectiveness of our tree algorithm (particularly in the cases of locational anomalies) which exploits the root in the beginning and the leaf in the long run, achieving the best of both.

Our comprehensive partitioning strategy enables robust performance and generalizability, even in situations where training data is limited in certain areas. By leveraging the capabilities of the tree algorithm, our approach demonstrates its effectiveness in handling such challenging scenarios, making it a valuable tool in video surveillance applications. As depicted in Fig. 3.18, our algorithm showcases significant advantages over the root node model, consistently outperforming it even with a limited number of training samples. One of the notable strengths of our algorithm lies in its ability to address the challenge of having parts of the frame with no training samples. By incorporating these untrained parts into models that are connected

to sampled regions, our algorithm effectively integrates the spatial information and improves detection performance.

Accordingly, we observe that the performance of the root node model deteriorates relative to the other models as the number of samples increases. This can be attributed to that the non-stationary spatial statistics introduce biases into the root node model's performance. In contrast, the leaf node model demonstrates an inherent capability to detect locational anomalies, resulting in improved performance with each additional sample. Notably, our context tree-based partitioning algorithm facilitates a gradual transition from more general models to localized models, leading to substantial gains in the transient phase while remaining competitive with the leaf node model in the long run.

In Table 3.2, we present the highest achieved AUC scores for each of the models utilized in our experiments, including our proposed algorithm, root node model, and leaf node model. The results indicate that our algorithm outperforms the others, even in the simplest datasets with unique anomalies. Moreover, our algorithm demonstrates significant improvements over the leaf node model, particularly in scenarios where there are insufficient training samples for certain parts of the scene. These findings provide strong support for our argument that intelligent partitioning of the entire sample space can lead to the development of a more robust and capable video surveillance system.

3.4.3 Supervised Anomaly Detection

Up to this point, we have focused on the problem of unsupervised anomaly detection, where the training data consists solely of samples representing normal behavior. In this section, we introduce a new approach for supervised anomaly detection in real-time video streams. This method employs a NP formulation to balance the trade-off between false alarms and missed detections in local anomaly detection models within a context tree, enabling precise control over the FPR while maintaining high detection power. Similar to CTBAD, the context tree-based NP classifier partitions the video scene into disjoint regions and trains individual NP classifiers for each partition, effectively capturing varying levels of scene complexity. The simplest partition contains a single NP classifier, while the finest partition consists of 2^D NP classifiers, where D is the depth of the context tree. Unlike the unsupervised approach, the training dataset in this supervised setting includes samples from both normal and anomalous events, eliminating the need to make assumptions about the

anomaly distribution for NP optimization. This inclusion also facilitates the use of a simple 0-1 loss function to compute both the node loss and the overall performance of the context tree.

Our method for supervised anomaly detection, denoted as f_t, calculates the prediction as $f_t(x_t) = 2 \times \text{sign}(M_{i,t-1}(x_t)) - 1$ with probability q_i for $1 \leq i \leq N_q$. Here, $\boldsymbol{x}_t \in R^d$ represents the feature extracted from the video stream, $M_{i,t-1}$'s are piece-wise linear NP classifiers (experts) trained on different partitions, N_q is the total number of experts, and q_i is the probability of selecting the prediction of expert i based on its local performance as the final outcome of f_t. In contrast, as stated in [205], NP classification aims to maximize the detection power while upper bounding the false positive rate by a user-defined value τ. For supervised anomaly detection method, each expert $M_{i,t-1}$ is a piece-wise linear NP classifier that solves $f^* = \arg\max_{M_{i,t-1}} P_d(f)$ subject to $P_{fa}(M_{i,t-1}) \leq \tau$, where P_d, P_{fa}, and τ represent the detection, false alarm, and target FPR, respectively. As explained in [204], we can use a piece-wise linear perceptron for $M_{i,t-1}$ and estimate the no detection rate ($P_{nd} = 1 - P_d$) and the FPR to define the overall NP loss of the classifier with the Lagrange objective, as follows:

$$L(f, \gamma) = \frac{\lambda}{2}||\boldsymbol{w}||^2 + \hat{P_{nd}}(f) + \gamma(\hat{P_{fa}}(f) - \tau), \tag{3.9}$$

Here, λ is the regularization parameter, and γ is the class weight, which ensures convergence to the target FPR τ. We train the parameters of f using stochastic gradient descent with the loss function in Eq. (3.9), which yields the following updates:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta_t\Big(\lambda \boldsymbol{w}_t + \mu_t \nabla_{\boldsymbol{w}} l\big(y_t f_t(\boldsymbol{x}_t)\big)\Big) \tag{3.10}$$

$$b_{t+1} = b_t - \eta_t\Big(\mu_t \nabla_b l\big(y_t f_t(\boldsymbol{x}_t)\big)\Big) \tag{3.11}$$

$$\gamma_{t+1} = \gamma_t + \beta_t\Big((1_{\{y_t=-1\}} t/n_{t\_}) l\big(y_t f_t(\boldsymbol{x}_t)\big) - \tau\Big), \tag{3.12}$$

where η, β are learning rates for perceptron and class weight and $n_{t\_}$ is the total number of negative (normal) samples.

For supervised anomaly detection, we use the UCSD Pedestrian, ShanghaiTech, and Street Scene datasets in our experiments. To construct training sets containing both normal and anomalous samples for the supervised anomaly detection model during

both the training and testing phases, we ensure that the causality between features collected from consecutive video frames is preserved.

To achieve this, we divide the original training and testing sets at the midpoint in time, keeping the video frame order intact. The first half of the video sequence is assigned to the new training set, while the second half is assigned to the new testing set.

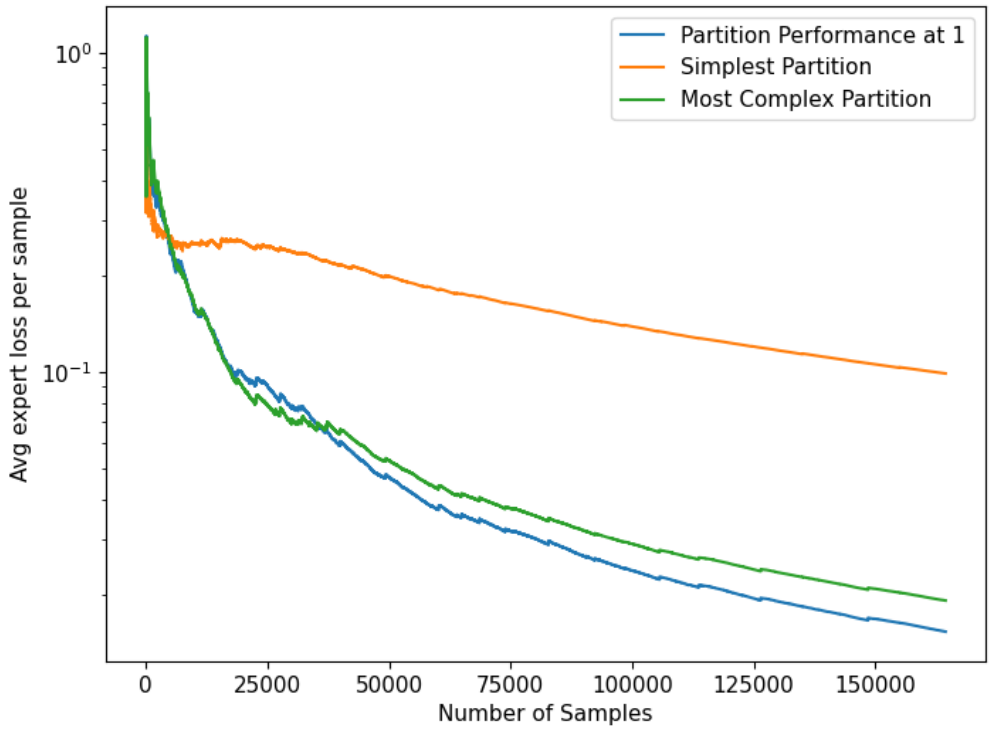


Figure 3.19 In the above figure, we observe the average loss per node in the partition as a function of the number of training samples. As expected, the graph shows that the loss decreases as the number of training samples increases, indicating an improvement in the algorithm's performance.

Additionally, we analyzed the performance of different partitions of the context tree for each new sample ($x(t)$). Fig. 3.19 shows the comparison between the best-performing pruning model at the end of training and the least complicated partitioning that spans the whole space ($P_{least} = f_{root}$) and the most complicated partitioning, ($P_{most} = \forall f_i i \in I_{leaves}$, where I_{leaves} is the set of leaf nodes of the context tree), which is the combination of all the leaf node models of the context tree.

We observed that as the number of training samples spanned by a model increases (for a context tree node), the model becomes more performant at detecting local

anomalies, resulting in an overall increase in performance. This indicates that in areas with high sample counts (such as sidewalks in pedestrian traffic), the best-performing node is located closer to the leaf nodes. In contrast, for parts of the image with a low sample count (such as buildings in the scene), the algorithm assigns models spanning larger neighborhoods as the best available model, which is expected.

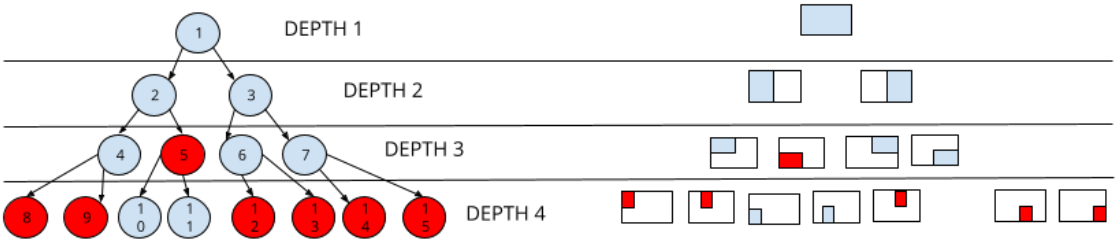


Figure 3.20 In the above figure, we observe the context tree algorithm and its corresponding partitioning. The nodes of the partition are highlighted in red, which indicates that these nodes contain the most significant information for detecting anomalies in the data. Overall, the figure demonstrates the effectiveness of the proposed context tree algorithm in achieving appropriate partitioning and improving the accuracy of anomaly detection in video scenes.

Fig. 3.19 also illustrates that a simple partitioning model (single region covering the whole scene) has better performance at the beginning of the training process because global statistics result in a better model since more samples are utilized. However, as the number of samples increases, more complex partitions (i.e., partitions with lower-level nodes as their members) become more dominant and better at distinguishing local and global anomalies. The best partitioning model, which our algorithm approaches, follows the same trend. For parts of the image with fewer samples (5 in Fig. 3.20 and Fig. 3.21), the node on the upper branches has the highest performance, whereas, for disjoint or image parts with ample samples, leaf nodes exhibit the same behavior.

We illustrate the performance of the context tree with a depth of 4 for different false positive rates set at $fpr \in 5e-3, 1e-2, 5e-2, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9$ to calculate the receiver operator characteristics (ROC) for different features and datasets. For this purpose, we execute experiments for the dynamic motion and image features from CAE bottleneck proposed in [14] for the UCSD Pedestrian dataset only, and the statistics-based feature defined in detail in [13] for all datasets. At each fpr, we compute a new tree model based on NP classifiers set at the corresponding fpr and calculate its achieved false and true positive rates for each feature set. In Fig. 3.22, we show the overall performance of the system for features and datasets, and we provide the AUC values in Table 3.3.

Figure 3.21 Above, we observe the partitioning of the model space for the appropriate partition at a particular depth. This partitioning allows for the identification of groups of statistically similar regions, which can be used to train a separate classifier for each combination of regions. This approach effectively manages the bias-variance trade-off and can improve generalization performance.

As shown in Fig. 3.22 and Table 3.3, the performance of [13] is superior to [14] for UCSD Pedestrian, which is also inline with the results in the original papers. In addition, we observe that by capitalizing on the local statistics, we are able to achieve higher performance compared to both more complex and simple methods, as seen in Fig. 3.19. We also perform better for both Shanghai and Street Scene datasets.

3.5 Discussion

We proposed CTBAD, a method that effectively partitioned video frames with a context tree for anomaly detection. This approach was particularly effective for locational anomalies exhibiting spatially nonstationary statistics, even with limited training samples. By combining models with increasing complexity and locality, CTBAD detected anomalies in specific regions, surpassing the limitations of relying solely on global scene statistics. This represented a significant improvement over existing methods. Moreover, the approach demonstrated robustness across di-

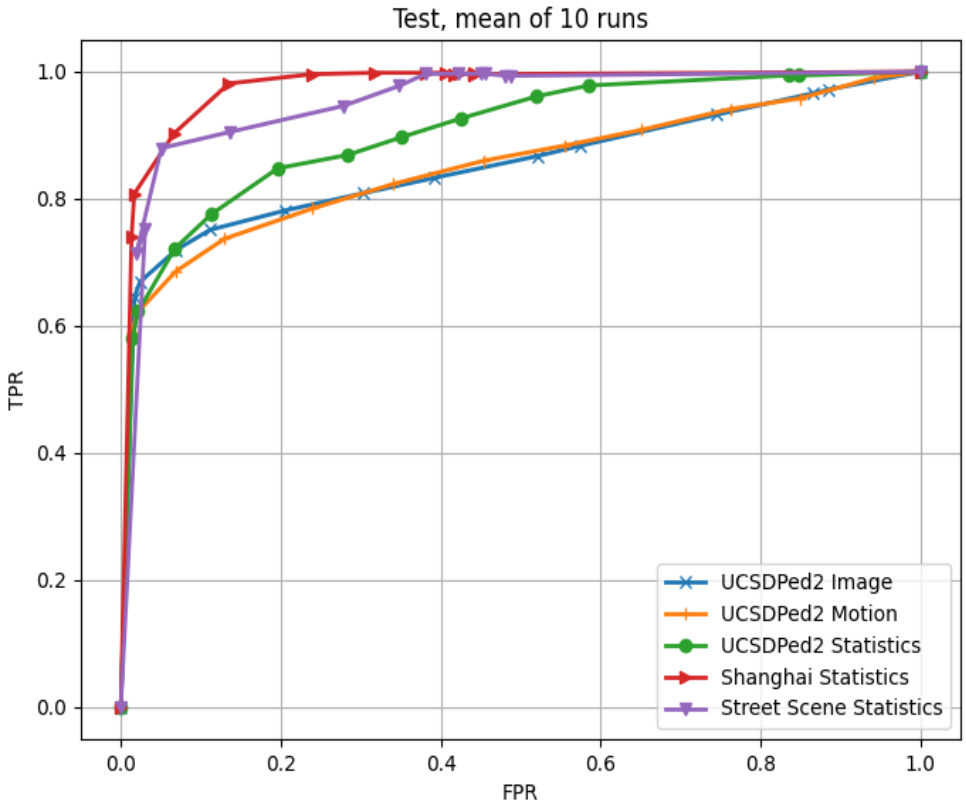


Figure 3.22 ROC curves for [13] (image and motion) and [14] (statistics) features for UCSD Pedestrian dataset, [13] features for Shanghai and Street Scene datasets.

verse scenes and actors, emphasizing its potential for real-world video surveillance applications. CTBAD was also easily integrable with current SOTA techniques.

However, we identified a specific failure mode of the algorithm. This issue arose when anomalies were spatially stationary (where global statistics were sufficient), and the training samples were limited such that certain image regions lacked activity during training but exhibited activity during testing. This situation led to a typical train-test mismatch. Although this was a common issue for data-driven approaches, it could be mitigated in our framework by assigning more weight to simpler models (choosing $\beta \sim 0$). However, this adjustment required additional parameter tuning for β.

The underlying causes of this failure were threefold:

- The most granular and complex partition model suffered from undertraining.
- No single partitioning scheme provided optimal performance due to spatially stationary statistics.

- The loss function was designed to converge to the most granular partition model, assuming sufficient data was observed over time.

To address this issue, one could switch to the second-most granular model that was sufficiently trained when statistics were insufficient.

We also conducted experiments for supervised anomaly detection using a context tree-based NP classifier, which replaced the local anomaly detection models in CT-BAD with NP classifiers. The results of our experiments demonstrated that the model consistently converged to the appropriate partition rather than the finest one, achieving high AUC scores across various feature extraction methods. Moreover, our pipeline was designed to seamlessly integrate with existing or future feature extraction techniques and online models, making it well-suited for sequential anomaly detection tasks.

Table 3.2 AUC Results of different partitioning algorithms for different datasets.

Anomaly Detection Method	Feature Type	UCSD	Avenue	Shanghai	Street Scene
CTBAD with $\beta = 0.875$	CAE (Image)	0.82	0.71	0.54	0.79
CTBAD with $\beta = 0.875$	CAE (Dynamic Image)	0.85	0.64	0.76	0.83
CTBAD with $\beta = 0.875$	Motion Statistics	0.93	0.77	0.85	0.85
CTBAD with $\beta = 0.875$	HOF	0.92	0.73	0.87	0.87
Root Node Model	CAE (Image)	0.83	0.60	0.48	0.60
Root Node Model	CAE (Dynamic Image)	0.87	0.59	0.65	0.63
Root Node Model	Motion Statistics	0.93	0.77	0.77	0.48
Root Node Model	HOF	0.91	0.75	0.77	0.56
Leaf Node Model	CAE (Image)	0.80	0.70	0.59	0.69
Leaf Node Model	CAE (Dynamic Image)	0.84	0.63	0.78	0.80
Leaf Node Model	Motion Statistics	0.82	0.44	0.83	0.80
Leaf Node Model	HOF	0.74	0.44	0.86	0.80

Dataset and Feature Set Name	**AUC**
UCSD Pedestrian Image Features	0.853
UCSD Pedestrian Dynamic Features	0.851
UCSD Pedestrian Statistics Features	0.907
Street Scene Statistics	0.953
Shanghai Statistics	0.977

Table 3.3 AUC Values for the ROC curves in Fig. 3.22

4. SSVEP-based BCI Speller character identification with Domain Adaptation

SSVEP-based BCI spellers provide a vital communication tool for individuals with disabilities, allowing them to spell words using electroencephalograph (EEG) signals. However, the complex process of collecting and calibrating EEG signals for each new user presents significant practical challenges. To address these challenges, we propose a novel character identification approach that leverages both labeled data from previous users and unlabeled data from new users. We treat these data sets as distinct domains and frame the integration of unlabeled data as a DA problem. Our method is specifically designed to maximize the utilization of existing user data, thereby improving accuracy and maximizing the ITR across various signal lengths. We introduce a self-improving iterative DA system that capitalizes on pseudo labels generated by the system itself, utilizing the similarity between previous and new users' data. This similarity metric enables our model to enhance previous user data and continuously self-improve over time. This iterative process leads to continuous improvement, resulting in a more robust and adaptive system for SSVEP-based BCI spellers.

4.1 Introduction to SSVEP-based BCI Spellers

BCIs are technologies that set-up a direct link between the brain and a computer through brain signals [41]. Because of its noninvasive nature, EEG is usually preferred to capture brain signals in BCIs. [42]. SSVEP are generated in the brain in response to a visual stimulus flickering at a fixed frequency, which provides a high signal-to-noise ratio [43] when measured with EEG and used as a control signal in BCIs. SSVEP-based BCI systems have many applications from robotic control to gaming [270]. One of the prominent applications is the speller system that enables

individuals suffering from serious motor neuron diseases to communicate with their environments [41].

In SSVEP-based BCI speller systems, a character matrix, where each character flickers with a distinct frequency, is presented to the user. The main goal is to correctly identify the attended character by the user as successfully as possible, solely based on the measured brain EEG signal (SSVEP). The success of the character identification is quantified by the information transfer rate (ITR), a combined objective of accuracy and time [271]. To achieve high ITRs, a separate calibration session is typically conducted in these systems before a new user starts using the system, because the EEG signal statistics are known to be highly nonstationary [272]. This calibration session is a period of supervised data collection and algorithm training using the collected data [41], which is obviously inconvenient for the new users as it prevents the product-ready use. Efforts to mitigate this issue include developing DG-based algorithms [154] that aim to transfer information existing in previous users' data to new users with no data/calibration at all. However, the ITR drop is generally significant when the calibration is removed. In other words, the ITR difference between the calibration-based algorithms and the DG-based ones remains to be large [270], leaving rooms for improvement.

Calibration plays a crucial role in SSVEP-based BCI speller systems. It involves collecting and using calibration data from the user to train the algorithms and customize them for individual users. Calibration helps establish a reliable mapping between the user's brain signals and the corresponding characters, thereby enhancing accuracy and ITR [270].

The scarcity of published SSVEP data poses another challenge, impeding the progress of experiments and algorithm development. Collecting reliable and high-quality SSVEP data is a challenging and time-consuming task, limiting available resources for research and experimentation. Furthermore, gaining a comprehensive understanding of SSVEP-based BCIs and their associated challenges requires expertise in the field. The complexity of brain signals, signal processing techniques, and statistical analysis demands specialized knowledge to effectively address the issues and develop innovative solutions [41].

Data collection from SSVEP-based BCI speller systems face challenges related to stimulus design, electrode placement, target identification methods, parameter optimization, data scarcity, and the need for expertise for each new user. Overcoming these challenges requires continuous research, collaboration, and the application of advanced techniques to improve the field of SSVEP-based BCI spellers [20].

Data collection from SSVEP-based BCI speller systems, as discussed in [20], is a challenging and demanding task, especially for new users. This poses a significant barrier for individuals who are newcomers to these systems and wish to utilize them. The data collection process for user calibration often requires users to focus on multiple visual stimuli with different flickering frequencies, which can be mentally and visually taxing. This can lead to user fatigue and reduced performance during actual data acquisition. Therefore, it becomes imperative to address these challenges and devise strategies to alleviate the burden on users, ultimately enhancing the accessibility and user-friendliness of SSVEP-based BCI speller systems by eliminating the entire calibration process.

The objective of this study is to address the challenges associated with low accuracy and low ITR in DG methods for character identification algorithms in SSVEP-based BCI speller systems without any calibration data. The proposed solution leverages unsupervised calibration data from new users to enhance the significance of existing user data. This is achieved by employing similarity methods between already calibrated expert user data and new user data. By incorporating this additional information from new users, we adopt an alternative approach to DG, which allows us to narrow the character identification margin with incoming new user data compared to traditional methods. This strategy not only enables us to improve our results but also facilitates continuous performance enhancement. The influx of additional data enhances our method's accuracy and ITR, even when dealing with low signal rates, thereby ensuring robust and reliable performance over time.

Additionally, we can argue that this brings additional information compared to DG methods, thus helping us close the margin mentioned earlier. On the other hand, when using the user system, it already generates this data, therefore suggesting continuous improvement without causing much difficulty to the user.

Instead, the algorithm is trained on SSVEP-based BCI speller data from already calibrated data of different users. The key concept in this thesis is to treat each user as a separate domain, which is part of a general domain shared by all users. The goal is to discover the common characteristics and patterns within this shared domain, enabling the algorithm to generalize across users. Successful DG for each user mitigates calibration issues in the SSVEP-based BCI speller and eliminates the need for user-specific algorithms. Ultimately, a single global model can be developed, simplifying the process compared to dealing with multiple user-specific models.

The primary aim of this study is to address the challenges associated with the target character identification (EEG signal classification) algorithms within SSVEP-based BCI speller systems, without requiring additional calibration steps for new

users. Our proposed character identification method is trained using a combination of SSVEP-based BCI speller data from previous users (such large scale data are already publicly available in the literature, e.g., [20; 44]) and a new user's unlabeled data with pseudo labels created during model training. Note that the new user's unlabeled data do naturally accumulate as s/he uses the speller interface. The central concept of our method revolves around leveraging the similarities between new and previous users to develop a self-improving iterative system that effectively utilizes all available data. In essence, our method is a DA technique [72] which eliminates the necessity for user-specific calibration processes while outperforming the DG approach. Our strategy also supports continuous performance enhancement. The influx of unlabeled data from the new user (as s/he uses the system) enhances the accuracy and ITR, thereby ensuring consistent and dependable performance over time.

The proposed solution leverages unsupervised labeled data from new users to augment the significance of previous user data by utilizing models trained on previous user data. This augmentation is achieved through similarity methods that capitalize on the data of previous users who exhibit similar characteristics to the new user. By integrating this additional information from new users, we adopt an alternative approach that enables us to narrow the character identification margin with incoming new user data compared to traditional methods. This strategy not only improves our results but also supports continuous performance enhancement. The influx of additional data enhances the accuracy and ITR of our method, even when dealing with low signal rates, thereby ensuring consistent and dependable performance over time.

Furthermore, it can be argued that this approach provides valuable additional information compared to DG methods, which solely rely on previous user data to establish shared domain characteristics among users for enhancing performance for new users. In contrast, when utilizing the DA system which utilizes new user data, there is also a continuous improvement without imposing significant difficulties on the user. This highlights the advantage of our approach in leveraging unsupervised labeled data from new users, enabling a more nuanced and effective adaptation process that can lead to improved overall performance.

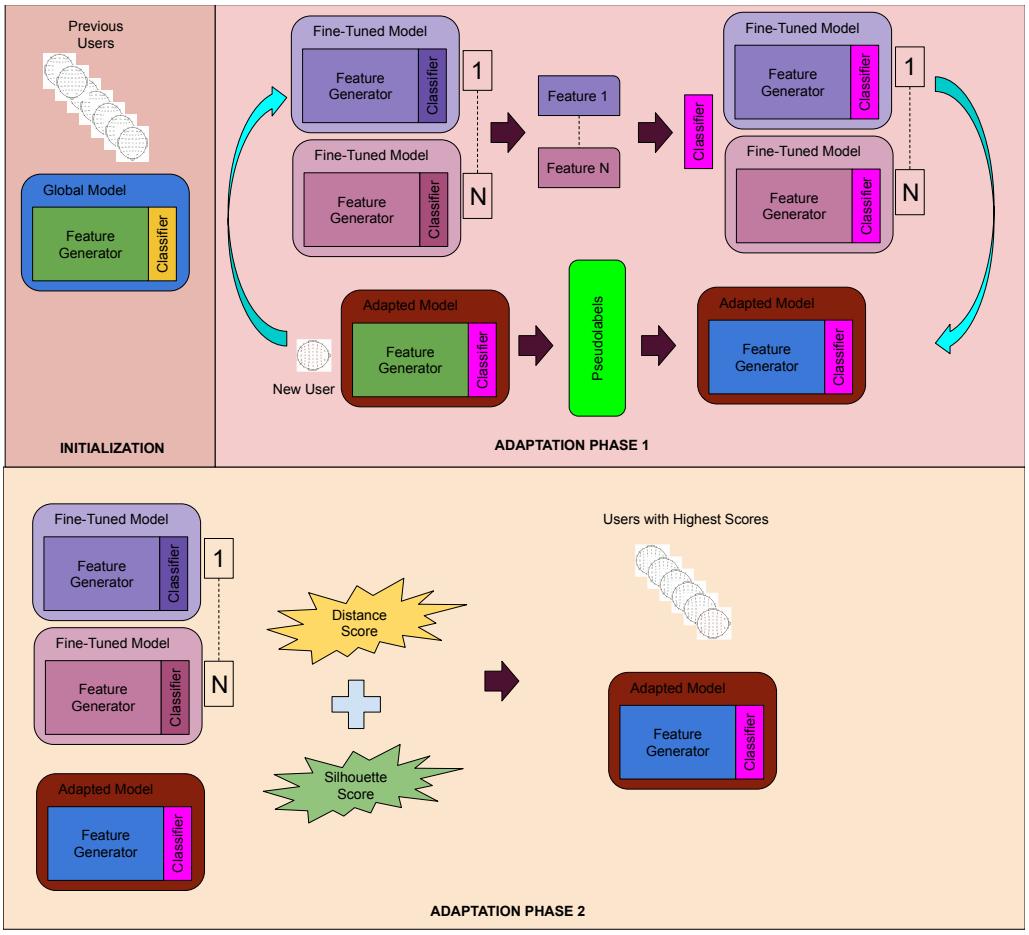


Figure 4.1 The proposed SSVEP-based BCI speller system architecture for target character identification comprises three key steps: global model generation, model fine tuning, and model adaptation. Initially, a global network (Γ_{global}) is created using labeled training data from all previous users, serving as the foundation in the subsequent steps for adapting to the new user. The model fine tuning step refines the feature generator (f_{global}) while keeping classifiers unchanged, tailoring feature extraction to each user's characteristics for improved performance. Model adaptation enhances the global model's performance by training the classifier using fine-tuned features and generating pseudo labels for the new user's unlabeled data iteratively, enabling semi-supervised learning and continuous adaptation until saturation is achieved. Overall, this structured approach builds a robust classification system capable of adapting to varying data characteristics and maximizing information utilization.

4.2 Related Work

The target character identification methods of SSVEP BCI speller systems can be broadly classified into two categories: calibration-based and calibration-free. Calibration-based methods, [49; 273], require an initial calibration phase where new users provide labeled training data through a separate EEG session to enable the

system to recognize characters accurately during subsequent uses. On the other hand, calibration-free methods, such as [274] and [275], eliminate this calibration step for new users, allowing them to utilize the system without any prior EEG experimentation.

Calibration-free methods can further be categorized into three groups: completely training-free methods [274; 275], DG [18; 19], and DA [276; 277] based methods. Completely training-free methods provide plug-and-play usability with no training at all (neither with previous users' data nor new users' data), whereas both DG and DA employ training with labeled data from previous users. DG does not adapt to new user data, whereas DA adapts by using the unlabeled data of new user accumulated during the system use. By using this terminology from machine learning, we aim to provide fresh insights into SSVEP character identification, drawing inspiration from the concepts of DG and DA found in the literature.

Calibration-based methods consistently outperform calibration-free methods, as evidenced by [49] which showcased deep neural networks (DNNs) achieving superior performance with calibration data. They demonstrate the effectiveness of their DNN [49] (which set the SOTA at the time) by utilizing the publicly available large scale BENCH [20] and BETA [44] datasets. Subsequent studies have delved into CNNs to address character identification challenges. For instance, [278] introduce task-related component analysis net (TRCA-Net), a pioneering algorithm merging TRCA's [279] spatial filters with CNN models to elevate signal-to-noise ratio and achieve precise identification. TRCA-Net showcases improved performance and adaptability across varied CNN architectures with BENCH and BETA datasets. Additionally, [280] presents the parallel multi-band fusion CNN (PMF-CNN) method, which integrates spatial and temporal self-attention modules alongside a squeeze-excitation module to capture correlation information within SSVEP signals, augmenting character identification accuracy. Employing a dual-stage training regimen and a brain functional connectivity analysis bolsters algorithm robustness and confirmed the approach's efficacy. Furthermore, in their work, [281] introduce EEGformer, a pioneering model for EEG analysis. This innovative approach combines a depth-wise convolution-based 1D CNN with an EEGformer encoder featuring temporal, synchronous, and regional transformers. Notably, EEGformer also includes a decoder component with a comprehensive architecture tailored for efficient EEG signal processing. Demonstrating effectiveness across SSVEP-based BCI, emotion analysis, and depression discrimination tasks (across diverse applications), the model's performance is validated on the BENCH and also additional EEG datasets.

In another study [282], they introduce a novel joint frequency-phase modulation

method and a user-specific decoding algorithm, resulting in an increased ITR of 60 characters per minute. Their approach demonstrates notable improvements in character identification. Similarly, authors of [283] address the issue of low SNR in SSVEP data by effectively decoding the SSVEPs within a short data length. They accomplish this by reducing background EEG activities using TRCA. As a result, they achieved significantly higher ITRs compared to previously considered levels.

In their paper [19], the authors propose a novel training-free framework for frequency-phase coding SSVEP BCI spellers, with a focus on target detection. The framework is centered around transferring SSVEP signals from source users to a new user (referred to as the target user) to capture critical frequency and phase information. They introduce the transfer template-based Canonical Correlation Analysis (tt-CCA) method, which extends upon Canonical Correlation Analysis (CCA) [284]. CCA is a statistical technique that explores the relationship between two sets of variables by identifying highly correlated linear combinations known as canonical variates.

In tt-CCA, the authors generate transferred EEG signal templates for the target user at the single-channel level using data from the source users. These templates are derived by grand averaging the corresponding channel's EEG signal across the source users. To identify targets, a combination of Pearson correlation coefficient and CCA is computed to measure the similarity between the SSVEP signals and transferred templates. Moreover, they propose an online transfer template method (ott-CCA) that enables real-time adaptation of the templates by gradually updating them based on the SSVEP signals. By leveraging inter-user information embedded in the SSVEP signals, these methods provide effective means for target detection.

The effectiveness of the proposed methods is validated through an offline frequency-phase coding SSVEP BCI speller experiment, where classification accuracy is significantly improved, highlighting the advantages of exploring and utilizing inter-user information in SSVEP signals for BCI implementation. The results demonstrate that incorporating inter-user information into the BCI system enhances classification accuracy by up to 7.5% compared to traditional methods. Additionally, the proposed approach reduces the training time needed to calibrate the BCI system.

The Combined-CCA method is an extension of CCA for target detection in SSVEP BCI spellers, as proposed in [285]. It integrates reference signals from traditional CCA with prototype responses obtained from averaged SSVEP training trial signals. Instead of averaging the SSVEP trial signal solely from the same user's calibration session, a pooled transfer approach is introduced where SSVEP trial signals from other users are also averaged.

The method involves computing correlation coefficients between projections using spatial filters derived from the CCA between the test set and the averaged SSVEP signals. This computation results in a correlation vector. To obtain the SSVEP detection score, a weighted sum of these correlations is calculated. The template with the highest weighted correlation value is selected as the SSVEP target.

The Combined-CCA method improves target detection in SSVEP BCI spellers by incorporating inter-user information through the pooled transfer approach. By combining reference signals from traditional CCA with prototype responses derived from averaged SSVEP training trials, it enhances the accuracy of target selection. This approach allows for more robust and reliable SSVEP-based BCI systems.

The Adaptive-C3A method, presented in [18], builds upon the Combined-tCCA approach to enable unsupervised adaptation of SSVEP templates. It operates in a simulated online scenario, where trials are classified using Combined-tCCA, and the resulting predictions are used as pseudo labels. The method identifies high-confidence trials based on the best vs. second-best (BvSB) confidence ratio and selects them for template adaptation.

The adaptation process in Adaptive-C3A involves updating the existing SSVEP templates using a weighted averaging scheme. Eligible trials for adaptation are determined using a threshold, allowing only high-confidence trials to contribute to the template update. By incorporating this unsupervised adaptation mechanism, Adaptive-C3A achieves a significant performance improvement compared to the Combined-tCCA and standard CCA methods.

Experimental results demonstrate that Adaptive-C3A outperforms the Combined-tCCA method by 20% in terms of classification accuracy. Moreover, it surpasses the standard CCA method by up to 40% in terms of performance. The ability to adapt SSVEP templates in an unsupervised manner allows the method to continually improve its target detection capabilities over time, enhancing the reliability and effectiveness of SSVEP-based BCI systems.

In their work [17], the authors propose an online adaptive method by combining and enhancing existing approaches. They first utilize the method introduced in [276], called Prototype Spatial Filter (PSF). In [276], the authors propose learning a spatial filter, referred to as the PSF, from multiple Canonical Correlation Analysis Spatial Filters (CCA-SFs) associated with different stimulus frequencies or users. The objective is to find a spatial filter that maximizes the similarity to all CCA-SFs. This is achieved by iteratively updating a covariance matrix with each new CCA-SF, and the resulting PSF is obtained as the eigenvector of the updated covariance

matrix.

Another approach employed by the authors of [17] is based on the work presented in [277]. In that paper, the authors introduce a modified version of conventional CCA called multi-stimulus CCA (msCCA) for learning a common spatial filter from a user's multi-stimulus SSVEP templates. They further extend this approach to an online learning mode, referred to as online msCCA (OMSCCA). OMSCCA adaptively learns an online spatial filter by updating covariance matrices trial by trial, and the spatial filters for the next trial are computed based on the updated covariance matrices.

In the proposed online adaptive CCA method (OACCA) in [17], the authors incorporate the PSF, OMSCCA-SF, and correlation coefficients, which are learned online. The spatial filters are updated based on previous trials, with zero initialization for the first trial. Correlation coefficients are computed using CCA and the spatial filters to measure the similarity between the input data and the stimulus frequencies. These coefficients are summed to obtain the detection score. The OACCA algorithm utilizes the filter bank technique to decompose the input data into subband data for feature extraction, classification, and online adaptation.

Overall, the OACCA method enhances the online performance compared to standard methods by incorporating the PSF and OMSCCA-SF spatial filters and updating them adaptively based on previous trials. This online adaptive approach enables the system to continually improve its performance over time and adapt to changes in the SSVEP signals, making it a promising method for real-time SSVEP-based BCI applications. Their offline learning approach achieved an ITR of 158.87 bits/min on the BENCH dataset and 123.91 bits/min on the BETA dataset. In online learning, the ITR reached approximately 95.73 bits/min.

To overcome these disadvantages with an acceptable performance researchers propose to use calibration-free methods. These methods introduced with [19], in which the authors propose a training-free framework for SSVEP BCI spellers using the tt-CCA method, which transfers SSVEP signals by averaging EEG signals from source users to generate transferred templates. The similarity between SSVEP signals and templates is measured using Pearson correlation coefficient and CCA [284]. They also introduce the ott-CCA method for real-time template adaptation, demonstrating improved classification accuracy in offline experiments. The Combined-CCA method introduced in [285] enhances target detection by integrating traditional CCA reference signals with prototype responses obtained from averaged SSVEP training trials. This method uses a pooled transfer approach and computes correlation coefficients between projections using spatial filters derived from CCA. By combining

inter-user information and traditional CCA, the Combined-CCA method improves target detection and enhances the robustness of SSVEP-based BCI systems.

Combined-tCCA proposed in [18] extends the approach to develop the Adaptive-C3A method for unsupervised adaptation of SSVEP templates. Predictions from Combined-tCCA are used as pseudo labels for template adaptation, and high-confidence trials are selected based on confidence ratios. The existing templates are updated through weighted averaging, resulting in significantly improved performance compared to Combined-tCCA and standard CCA methods. In a different study, [17] propose the online adaptive CCA (OACCA) method by integrating the Prototype Spatial Filter (PSF) [276] and online multi-stimulus CCA (OMSCCA-SF) [17] approaches. OACCA utilizes filter bank decomposition for feature extraction, classification, and online adaptation. It achieves improved online performance compared to standard methods, showing promise for real-time SSVEP-based BCI applications with a high ITR.

Despite their advantages, calibration-based algorithms come with certain limitations. They typically demand a substantial volume of SSVEP data per participant, as attempts to enhance accuracy through the utilization of SSVEP data from only other users often result in heightened misclassification rates compared to using an individual's own SSVEP data. Moreover, the calibration process necessitates participants to dedicate considerable time, potentially inducing fatigue and consequently impacting the quality of the acquired data. To address these limitations while maintaining acceptable performance, researchers have proposed calibration-free approaches (completely training-free, DG and DA).

In [16], the DNN architecture introduced in [49] is initially trained for each source user. Subsequently, the resulting ensemble of DNNs is transferred to the new user, utilizing the most representative user DNNs for predicting spelled characters. Notably, their ensemble of DNNs demonstrates superior performance compared to other DG methods [16]. While DG approaches are calibration-free and practical, they also suffer from not adapting to the new user, resulting in performances that may not be as satisfactory as those achieved by user-dependent DA approaches.

In a study by [15] on DNN-based SSVEP target identification, a significant step is the inclusion of a local regularity term in the loss function. This term ensures that neighboring instances have similar labels, setting their approach apart from earlier methods. Additionally, their method dynamically adjusts the weights of various components based on clustering performance, thereby eliminating the need for predefined parameter settings.

In this thesis, building on the approach of [15] in applying DA methods to SSVEP character identification, we additionally draw inspiration from methods such as SHOT [286]. In SHOT, the classifier module (hypothesis) of the source model is frozen, while the target-specific feature extraction module is learned by leveraging both information maximization and self-supervised pseudolabeling. We benefit from this approach, which implicitly aligns representations from the target domains with the source hypothesis. Furthermore, unlike the label noise encountered in conventional LLN scenarios [287], we consider that the label noise in utilizing pseudo labels in our context of DA follows a different distribution assumption. This distinction renders existing LLN methods, which rely on traditional distribution assumptions, ineffective in addressing the label noise in pseudo labels within DA contexts. Inspired by their work [286; 287], we propose the use of pseudo labels to enhance our method's performance in a way that is specific to SSVEP signals. In summary, here we propose a novel DA method for target character identification that not only better suits the unique characteristics of SSVEP signals but also maintains practicality for new users by eliminating the need for calibration. Next, we detail our novel contributions and highlight the important technical aspects of our method.

We briefly discuss several source-free DA methods from the machine/deep learning literature. For instance, SHOT, as proposed in [286], adapts a network to the target domain by freezing the classifier layer and fine-tuning the remaining parts (feature extractor) through pseudolabeling. In [288], authors introduce neighborhood reciprocity clustering (NRC), utilizing intrinsic data structure to cluster similar instances. Another study [287] addresses source-free DA from a label noise perspective, employing a regularizer to mitigate label noise memorization.

4.3 Problem Description

An SSVEP-based BCI speller system is a type of BCI that leverages SSVEP EEG signals to enable users to spell out characters / words or make selections on a screen composed of M flickering boxes. To choose one of the available options, participants position themselves in front of the screen and focus their attention on the $y^{th} \in \{1, 2, 3, ..., M\}$ target box. Brain responses (SSVEPs resulting from flickering stimuli) are measured in the form of multi-channel EEGs, denoted as $x \in \mathbb{R}^{C \times N}$ where C represents the number of channels used to record these brainwave signals and N represents the number of time samples collected from each channel.

These SSVEP EEG signals are generated based on the specific frequency and phase associated with the selected y^{th} target.

The challenge at hand is to establish the relationship, represented as $R(x) = \hat{y} \simeq y$, between the SSVEPs x and the user's intended target selection y in order to achieve the highest level of accuracy and speed in recognizing characters. A key metric for evaluating the effectiveness of this relationship is ITR, which takes into account both the accuracy of target identification P and the duration T of the stimulation (interaction period in seconds):

$$\mathrm{ITR}(P,T) = (\log_2 M + P\log_2 P + (1-P)\log_2\left[\frac{1-P}{M-1}\right])\frac{60}{T}.$$

Since ITR reflects the efficiency of the SSVEP-based BCI speller in conveying the user's intended selections, the goal of the presented research is to maximize it. From an analytical perspective, it is evident that a shorter T leads to a higher ITR if the accuracy P does not degrade, as this relationship is readily observable. Likewise, a higher level of accuracy P for the same T corresponds to an elevated ITR, as the first derivative of ITR with respect to P is positive within the interval $(0,1)$. However, the variables P and T are inherently interdependent, and reducing T adversely affects P. Therefore, we aim to maximize P for each predetermined T, and pick the (P,T) pair of the maximum ITR observation.

The process of collecting data from SSVEP-based BCI speller systems, as outlined in [20; 44], can indeed pose significant challenges. This challenge acts as a barrier for newcomers to these systems. Finding methods to alleviate this barrier can increase the overall usability of such systems and facilitate the integration of newcomers into the BCI speller environment. Traditional DNN architectures tend to tightly adapt to the specific statistical properties of the training data (previous users or source domains), and since such properties may not be present in the test set [289; 290] (new user or target domain), these traditional models often struggle to adapt effectively to unseen domains. To that end, one must certainly address the distributional changes (regarding EEG signal statistics) between different users which can actually be modeled as a domain shift problem [50].

In this context, we tackle the domain shift problem for SSVEP target character identification, with specific focus on the seamless integration of a new user (target domain) while leveraging labeled data from previous users (source domains). We also acknowledge the presence of unlabeled data from the new user generated during system usage. Consequently, the need for a separate calibration session for labeled data collection is obviated, thus removing a significant barrier for new users. Our technical objective is to distill essential and transferable knowledge from the source

domains, enabling effective adaptation to previously unseen target domains. This poses a challenging task, as it necessitates models to capture the underlying essence of the data and adapt across different data statistics. Specifically, we introduce a method that harnesses previously labeled user data to achieve target identification with the maximum possible ITR for new users with unlabeled data.

To achieve our goal, we capitalize on similarity scores between new user with unlabeled data and previous users with existing labeled data. These scores allow us to rank the k nearest previous users to the new user, facilitating the utilization of models fine-tuned on their labeled data. Subsequently, we generate pseudo labels from these models and employ them to fine-tune the developing model of the new user based on the created pseudo labels. While these pseudo labels may not perfectly align with the true labels, they provide a robust initial foundation for adapting the new user model. This methodology also enables continuous adaptation with incoming data of the new user during the system usage, thereby augmenting the overall performance of the model.

4.4 Proposed Method

In this section, we begin with a summary of our proposed method, providing an initial understanding and an overview of the workflow. We then delve into each component, explaining their goals and operational principles. Finally, we conclude by discussing the nuances of the proposed loss functions and similarity metrics.

Our method aims to enhance SSVEP character identification performance through a structured approach. We begin with a preprocessing step that filters and samples the raw EEG data. This involves extracting selected EEG channels and applying a bandpass filter to isolate relevant signals from each user's data. Once the data is preprocessed, it is split into two parts: new user and previous users data. The goal is to demonstrate the effectiveness of our model by adapting it to and testing with the new user data, while leveraging the already calibrated and established data from previous users. This systematic process ensures improved adaptation and identification performance for new users.

After preprocessing the data, we train a global model based on the DNN architecture described in [49], using data from previous users to establish a baseline performance. This training captures shared characteristics from previous users, which is also ben-

eficial for the new user. The global feature generator (consisting of all layers up to the fully connected (FC) layer) is then fine-tuned individually for each previous user using their specific labeled data. This fine-tuning process generates user-specific feature generators while keeping the global classifier's (last FC layer's) weights and biases fixed. The fine-tuned feature generators are then used to generate features, which serve as inputs for training the global classifier (last FC layer). By training the global classifier with these features, it benefits from the combined knowledge of previous users' fine-tuned models and better defines the boundaries of each label in the feature hyperspace. The global classifier replaces the classifiers of the fine-tuned models, and the fine-tuned feature generators are updated by training them a second time (second iteration), keeping the global classifier's weights and biases fixed. These updated feature generators are saved for subsequent iterations, integrating the knowledge acquired from the global classifier. By retraining (second iteration) the global classifier with these fine-tuned features, the model's ability to generalize and adapt to new user is enhanced. This approach leverages the unique characteristics of previous users' data to optimize performance for new user, enabling the model to achieve more accurate predictions and better handle variations across different users.

Up to this point we described only two iterations; in general, the fined-tuned models continue improving during many iterations. During the same iterations, we also employ a separate inner loop of adaptation process for the new user that utilize pseudo labels. This adaptation process involves two iterative loops: an outer (main) loop for for the fine-tuning operations described so far, and an inner loop for adapting the new user's network with pseudo labels. In essence, the parameters from a pre-recorded adapted model (from the previous outer loop iteration) are transferred to a new model that retains the same structure but incorporates the updated global classifier of the current (outer) iteration. The feature extractor of the transferred model is then refined recursively (inner loop) in an unsupervised manner, using pseudo labels generated from the adapted network of previous iteration as targets for training. This process continues until saturation is reached, where further iterations yield minimal performance improvements. For initialization, the last FC classification layer of the very first global network is updated during model fine-tuning, then this updated global network is transferred for new user and adapted (only feature extractor, otherwise the FC is kept fixed) in an unsupervised fashion with the unlabeled new user's data (but utilizing pseudo labels). This marks the end of the first iteration in which, recall that, we also obtained the fined-tuned networks of previous users that we later use in a separate other adaptation process. In the next iteration, the adapted network's FC is first updated, then transferred to new

user, its feature extractor is updated, and so iterations continue similarly.

Once the aforementioned outer and inner loops are completed, we obtain a set of fine-tuned networks and one adapted network. Then, we start one more and final separate other adaptation phase that again works in unsupervised fashion. In this final unsupervised adaptation phase, a new loss function tailored for SSVEP signal classification and target identification is also introduced. The introduced loss function is centered around three concepts: 1) A silhouette score [15] that measures how well the new user data is finally clustered into classes is used for parameter selection and for assessing pseudo-label quality (now the pseudo labels are generated from the nearest users' fined-tuned networks in addition to the new user's self adapted network), 2) a local regularity loss term (leveraging the silhouette score) that enforces nearby new user instances to be classified similarly, and 3) a metric that measures user-to-user similarity that picks the most similar previous users to new user whose responses (in addition to the adapted network's responses) to new user data are obtained as pseudo labels (balancing the contributions of neighboring previous users). We describe these three components of our introduced loss function in great detail later in this section.

After giving this summary, we next describe the datasets used in this study. Afterwards, we continue with the notations as well as all the details of our method.

4.4.1 Datasets

To evaluate the performance of our method and compare it with the SOTA (SOTA) approaches later, we focused on two widely used SSVEP-based BCI speller datasets: BENCH [20] and BETA [44].

The BENCH dataset consists of EEG data from a 40-target BCI speller, including 64-channel recordings obtained from 35 users (8 previous, 27 naive) performing a cue-guided task. The virtual keyboard consisted of 40 flickers coded with joint frequency and phase modulation. Stimulation frequencies ranged from $8Hz$ to $15.8Hz$ with a step size of $0.2Hz$, and adjacent frequencies had a phase difference of 0.5π. Each user completed 6 blocks of 40 trials, with randomized flicker presentations and visual cues. This dataset allows for comparing different methods for stimulus coding and target identification, conducting offline simulations, designing BCI systems, and evaluating performance. Additionally, it provides high-quality EEG data suitable for computational modeling of SSVEPs.

The BETA dataset comprises EEG data collected from 70 users who have performed a 40-target cued-spelling task. This dataset is designed and acquired with the aim of meeting the requirements of real-world applications, making it suitable for testing in practical scenarios. Thus, EEG signals of BETA are relatively noisier. The study involved 70 healthy volunteers, with an average age of 25, including 42 males and 28 females. A sinusoidal stimulation method employing joint frequency and phase modulation was used to present visual flickers on the screen. The frequency and phase values for the 40 targets were determined using a frequency interval of $0.2Hz$ and a phase interval of 0.5π, similar to the BENCH dataset.

4.4.2 Notations

In this subsection, we introduce a set of notations. These notations are designed to provide the reader with a clearer understanding of the methods that we will subsequently elaborate on in greater detail. Also, the rest of the article and all of our explanations will be based on the scenario in which the n^{th} user is the designated as the new user.

N_{ch}	Number of EEG channels
F_s	EEG sampling frequency (in this study $F_s = 250$ Hz)
t	Duration of the (EEG) SSVEP signals in seconds
n	New user index
j, k	The outer (inner) loop index used in the fine tuning (and first model adaptation)
X	The entire dataset containing SSVEPs
$X_i \subset X$	SSVEP signals of the i^{th} user
U	Set of all users in the dataset
U^n_{train}	Set of previous users in the dataset, except the new user
$x_{n,m} \in X_n$	The m^{th} new user instance
$x_{i,m} \in X_i$	The m^{th} instance of the i^{th} previous user ($i \in U$ such that $i \neq n$)
Γ_{global}	Global network
$\Gamma^{i,j}$	Fine-tuned network belongs to the i^{th} user at the beginning of the j^{th} iteration step
$\overline{\Gamma}^j_{global}$	Adapted network at the beginning of the j^{th} iteration step ($\overline{\Gamma}^0_{global} = \Gamma_{global}$)
f_{global}	Feature extractor of the global network: $\mathbb{R}^{N_{ch} \times (F_s t) \times N_{band}} \rightarrow \mathbb{R}^{4 \times (125t)}$

$f^{i,j}$	Feature extractor of the i^{th} fine-tuned network at the beginning of the j^{th} iteration step: $\mathbb{R}^{N_{ch}\times(F_s t)\times N_{band}} \rightarrow \mathbb{R}^{4\times(125t)}$ (here, $i \in U$ such that $i \neq n$)
$\overline{f}^{j}_{global}$	Feature extractor of the adapted network at the beginning of the j^{th} iteration step: $\mathbb{R}^{N_{ch}\times(F_s t)\times N_{band}} \rightarrow \mathbb{R}^{4\times(125t)}$ ($\overline{f}^{0}_{global} = f_{global}$)
h_{global}	Single layer perceptron of the global network: $\mathbb{R}^{4\times(125t)} \rightarrow \mathbb{R}^{M}$
$h^{i,j}$	Single layer perceptron of the i^{th} fine-tuned network at the beginning of the j^{th} iteration step: $\mathbb{R}^{4\times(125t)} \rightarrow \mathbb{R}^{M}$, $i \in U$ such that $i \neq n$
$\overline{h}^{j}_{global}$	Single layer perceptron of the adapted network at the beginning of the j^{th} iteration step: $\mathbb{R}^{4\times(125t)} \rightarrow \mathbb{R}^{M}$
$v_{n,m}$	The m^{th} feature for the new user instance
$l_{n,m}$	The true unknown label of the m^{th} feature of the new user
$\tilde{l}_{n,m}$	The predicted label of the m^{th} feature of the new user
$\mathcal{L}^{(r)}_{total}(\lambda)$	Total loss calculated at the iteration r of the second (separate and final) adaptation phase. Note that the first adaptation phase has the outer / inner loop iterations j / k
$\overline{\Gamma}^{w_r}_{global}$	Adapted network with parameters w_r during the second adaptation phase
Λ	Set of self-adaptation loss weights (in this study $\Lambda = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$)
$\lambda \in \Lambda$	Self-adaptation loss weight
$w_r(\lambda)$	The parameters of the adapted network $\overline{\Gamma}^{w_r}_{global}$ that is trained in unsupervised fashion with the self-adaptation loss weight λ
$\tilde{l}_{n,m}(\lambda)$	The predicted label for the m^{th} new user feature obtained from the adapted network of the self-adaptation loss weight λ
s^{r}_{m}	The softmax outcome of $\overline{\Gamma}^{w_r}_{global}$ for $x_{n,m}$

S^u_{silh}, S^u_{dist}, S^u_{ovr} Silhouette, Distance and Overall scores for a given previous user $u \in U^n_{train}$

4.4.3 Preprocessing and Data Seperation

The primary objective of the preprocessing stage is to enhance the SSVEP data for character identification by focusing on relevant channels and frequency bands. To achieve this, data epochs from nine-channel SSVEP signals were extracted around stimulus onset considering a 140ms visual system latency. To optimize classification accuracy and reduce computational demands, the epochs were down-sampled to 250Hz as explained in [275]. After this, we apply the bandpass filter to remove any unwanted noise or interference. This filtering procedure is applied to both new and previous users in the dataset to improve the quality of the signals and emphasize the SSVEP responses related to the target characters. This, in turn, facilitates the subsequent identification and classification algorithms in effectively analyzing and distinguishing between different characters.

Moreover, the dataset is divided into new user and previous users. This division is accomplished by selecting one user as the new (test) user and utilizing all the remaining users in the dataset as the previous (training) users. The purpose of this separation is to evaluate the effectiveness of the overall solution when faced with a new user whose SSVEP signal statistics are totally unknown.

4.4.4 Generation of Initial Models

In the initial phase, we begin with training a global model Γ_{global}, designed to encompass the data from all previous users U^n_{train} for a new user n. Our primary objective is to capture the comprehensive, shared characteristics exhibited by the previous users. Subsequently, we focus on Γ_{global}, which represents a standardized DNN architecture.

Within this stage, the global model Γ_{global} undergoes a meticulous fine-tuning process tailored to each individual previous user $i \in U^n_{train}$. By leveraging the unique characteristics from their respective data, we create individual fine-tuned models $\Gamma^{i,j=1}$ (initialization: $\Gamma^{i,0} = \Gamma_{global}$) for each $i \in U^n_{train}$, with j denoting the outer

loop iterations. These fine-tuned models, distinguished by their precise adjustments, exhibit a notable enhancement in accuracy.

Of paramount importance during this fine-tuning phase is our strategic focus on updates to the feature generator $f^{i,j}$, which is defined as the layers preceding the FC layer. This carefully chosen strategy does also ensure that the newly minted models harmoniously share a common global classifier h_{global} with the global network Γ_{global}. This approach strikes an elegant balance, in the sense that it reflects all the fine-tuned models onto the same classification plane that is known to be effective as it comes from the global model but also maintains the previous user specific features. Hence, it acts as a strong levelization among the fine-tuned features by removing the degree of freedom regarding the classification layer. Note that, for example, a set of features can be translated without changing the classification accuracy by also correspondingly adjusting the classifier, which (this undesirable ambiguity) is nicely prevented in our approach.

4.4.5 Model Fine-Tuning

In the fine-tuning phase, we focus on feature extractors $f^{i,j}$ with an effective shared classifier h^{j}_{global}. We begin by extracting features $v_{i,m}$ from the fine-tuned feature extractors at the dropout layer outputs just before the classification layer, of each previous user in U^{n}_{train}. These extracted features $v_{i,m}$ serve as valuable inputs for the subsequent adaptation of the global classifier during training.

To implement our approach, we design a new global classifier h^{j+1}_{global}, which consists of a single FC layer that is analogous to the FC layer of the overall network. We train this newly created global classifier h^{j+1}_{global} using features $v_{i,m}$ extracted from the fine-tuned feature extractors $f^{i,j}$ of previous users, along with their corresponding data and labels. This method allows the global classifier to capitalize on the collective knowledge derived from the fine-tuned models of prior users, enhancing its performance through the integration of this aggregated expertise.

As the next step in model fine-tuning, we replace the FC layers (individual classifiers) of each fine-tuned model $\Gamma^{i,j}$ with the newly trained global classifier h^{j+1}_{global}. Following this, we further train the fine-tuned feature generators $f^{i,j}$ of these models while keeping the weights and biases of their individual copies of the global classifier h^{j+1}_{global} frozen. This results in the acquisition of new fine-tuned models $\Gamma^{i,j+1}$. Through this meticulous process, we ensure the seamless integration of knowledge

from the global classifier h^{j+1}_{global} into the individual fine-tuned models, fostering a harmonious synergy between collective and individual expertise.

4.4.6 First Model Adaptation (of the outer and inner loops)

In the first model adaptation phase, we refine a feature extractor $\overline{f}^{j}_{global}$ as well as an adapted network $\overline{\Gamma}^{j}_{global}$ belonging to new user instances iteratively (outer loop: j, and inner loop k but we drop here k for simplicity in notation) for optimal performance. Given the adapted model $\overline{\Gamma}^{j}_{global}$ from iteration j, we create a new model $\overline{\Gamma}^{j+1}_{global}$ by replacing its classifier layer h^{j}_{global} with the freshly generated common classifier h^{j+1}_{global}. Concurrently, we freeze the weights and biases of the classification layer and train the feature extractor with the inner loop iterations (k) in an unsupervised fashion. In this way, the model is prepared to generate a more finely-tuned adapted version through the same adaptation process, leveraging the shared global classifier.

Remark: The initially trained global model (at the very beginning) denoted by Γ_{global} is global in the sense that it encodes information from all the previous users. It also roots our adaptation due to the initialization $\overline{\Gamma}^{j=0}_{global} = \Gamma_{global}$, which becomes specific to the new user at the end of the iterations and so $\overline{\Gamma}^{j=\text{final}}_{global}$ is not global but rather specific and adapted. We keep the subscript “global" to indicate that it inherits the structure from the global model acquired at the end of initialization.

For this first unsupervised adaptation, it is necessary to determine the proxy labels (“pseudo labels") for the new user instances $v_{n,m}$ since the true labels $l_{n,m}$ are not known for the new user. This issue is handled in our first adaptation phase by using the predictions (as pseudo labels) $\tilde{l}_{n,m}$ of the model $\overline{\Gamma}^{j+1}_{global}$ that itself is being continuously adapted, yielding a self-learning or rather self-adaptation process which happens in an inner loop with cross-entropy loss. Using one's own network's responses as a loss signal can be viewed as a form of entropy regularization [291]. From another perspective, as our network architecture utilizes dropout regularization, the full network generates pseudo labels while the partial networks are only used for training. This discrepancy between full and partial networks enhances the performance, reinforcing the use of pseudo labels since the full network responses are never seen during training. Consequently, we save the newly adapted model $\overline{\Gamma}^{j+1}_{global}$. These fine-tuning and adaptation processes are iterated until either the classification performance reaches a satisfactory level or further improvements become marginal.

In the next section below, we continue with our second and final adaptation phase and its unsupervised loss function. The introduced loss function is designed to fully leverage a potent source of information embedded within the data: similarities between instances across different users.

4.4.7 Second and Final Model Adaptation

After completing the fine-tuning and first adaptation phases, we start our second and final model adaptation phase. In this second phase, rather than relying solely on the conventional cross-entropy loss, we consider that if there exist some previous users who are statistically similar to the new user, then their predictions regarding the new user instances might be beneficial. Motivated by this, in generating the pseudo labels, we do not only use the adapted model's own responses (self adaptation term) but also incorporate those fine tuned networks whose users are similar to the new user. This is embodied in our self adaptation (or self learning) loss term $\mathcal{L}_{sl}$ which is particularly effective in unsupervised and semi-supervised settings, aiding the network's ability to learn well-separated classes. Furthermore, we utilize a local regularity term that enforces that the nearby new user instances are to be classified similarly, yielding our local regularity loss term $\mathcal{L}_{loc}$. The resulting complete loss function is similar to that used in [15], but that is a source free approach whereas the presented approach here is source-dependent (see also our experiments where we compare the two methods in terms of their ITR performances). Consequently, we devise two types of similarities, one is user to user and the other is instance to instance, and in this study, both of them are based on silhouette scores [292].

We introduce a customized and unsupervised novel loss function that combines self-adaptation L_{sl} and local regularity L_{loc} terms, moving beyond the conventional cross-entropy loss function as outlined in [15]. After completing the first model adaptation phase, we further refine (in this second and final phase) the model using the introduced customized loss to enhance overall performance and better align the model with the new user's unlabeled data. The self-adaptation loss impacts the model by encouraging it to become more confident and consistent with its own predictions on the unlabeled data. By treating its predictions as pseudo labels and minimizing this loss, the model adjusts its parameters to reinforce these predictions, effectively tailoring itself to the target domain despite the absence of labeled data. The local regularity term is designed to fight the overfitting as detailed later.

The self-adaptation (or self learning) loss at iteration r (note that these iterations

start after the previous outer j / inner k iterations of the first phase are finished) in this second and final adaptation phase is given by

$$\mathcal{L}_{sl}^{(r)} = -\frac{1}{N}\sum_{m=1}^{N}\log(s^r_{m,\tilde{l}_{n,m}}),$$

where $s^r_m = [\forall y : s^r_{m,y}]$ represents (at iteration r) the adapted network's soft-max responses ($\overline{\Gamma}^{w_r}_{global}$ with $w_r(\lambda)$ denoting the adapted parameters) to $x_{n,m}$, and $\tilde{l}_{n,m}$ are the predicted labels (so pseudo labels) in the same iteration. Although a better notation here would be $\tilde{l}^r_{n,m}(\lambda)$, we dropped the superscript and the λ argument for simplicity. Here, λ is the weight of this self adaptation loss in our complete loss function, which we keep track of for hyper-parameter optimization later.

Since the self-adaptation loss does not account for the relationships between different new user data instances, we combine it with a local regularity loss in order to ensure that similar instances receive similar predictions. This local regularity helps to alleviate overfitting issues and enhances overall adaptation. For this purpose, we use the correlation coefficient to define closeness (similarity), yielding in the end the local regularity loss $\mathcal{L}_{loc}^{(t)}$ as

$$\mathcal{L}_{loc}^{(r)} = -\frac{1}{N}\sum_{m=1}^{N}\frac{1}{K_m}\sum_{q=1}^{K_m}\log(s^r_{m,\tilde{l}_{n,I_m(q)}}),$$

where I_m is the set of indexes sorted in descending order based on the correlation coefficient values between the target domain instances $\{x_{n,q}\}_{q=1}^{N}, q \neq m$, and the target domain instance $x_{n,m}$. Specifically, $x_{n,I_m(1)}$ is the most correlative (closest) to $x_{n,m}$. K_m is the number of neighbors considered for the instance x_m. The neighborhood size can vary depending on factors like user behavior or noise levels. Minimizing this loss ensures that the network provides similar predictions for closely related instances. Again, a better notation here would be $\tilde{l}^r_{n,I_m(j)}(\lambda)$, but we dropped the superscript and the λ argument for simplicity which should be kept in mind for the exposition continuing below.

Combining self-adaptation with local regularity, the total loss function is expressed as

$$\mathcal{L}_{total}^{(r)}(\lambda) = \lambda\mathcal{L}_{sl}^{(r)} + (1-\lambda)\mathcal{L}_{loc}^{(r)} + \beta\|w\|^2,$$

where $\lambda \in \Lambda$ is the weight of the self adaptation loss, and β is the coefficient of L2 regularization, set to 0.001.

This second and final model adaptation phase starts by transferring the weights of the last adapted model $\overline{\Gamma}^{j_{end}}_{global}$ into a new model $\overline{\Gamma}^{w_0}_{global}$ whose aim is to minimize the combined loss $\mathcal{L}^{r}_{total}(\lambda)$ at each iteration r until it converges. We utilize the same convergence criteria as explained in [15].

The model $\overline{\Gamma}^{w_{r-1}}_{global}$ is adapted for the new user by minimizing the total loss $\mathcal{L}^{r-1}_{total}(\lambda)$ using a candidate λ value from the set of Λ. This process generates adapted parameters $w_r(\lambda)$ and predictions $\tilde{l}_{n,m}(\lambda)$ for each value of λ. Subsequently, we assess how effectively each set of adapted network parameters $\{w_r(\lambda) : \lambda \in \Lambda\}$ clusters the unlabeled data from the target domain to determine which adapted network parameters should be used for the final prediction, either at the end of iterations or as needed during intermediate stages of adaptation.

4.4.8 Silhouette Score

We utilize the silhouette score [292] as a clustering metric or as a similarity score, which evaluates not only the accuracy of the adapted model's predictions for each new user instance but also the confidence level of the model in these predictions. This assessment involves measuring the confidence level associated with the assigned label by comparing the distance of the new user instance to the distances in a given previous user instances with the same predicted label. The objective is to ensure that the model's predictions exhibit both accuracy and a high level of confidence when clustering new and previous user instances together.

To compute the silhouette score, we provide a two-step calculation aimed at measuring the labeling quality and confidence. In the initial step, for each new user instance denoted as $x_{n,m}$, we compute the average distance between this instance and each other $x_{u,m}$ from a given previous user $u \in U^{n}_{train}$ where the pseudo label assigned to new user instance is the same as the label of the previous user instance. This can be represented as (while keeping in mind that the iteration r and the λ argument were dropped for simplicity: model prediction $\tilde{l}^{r}(\lambda)$ is the assigned pseudo-label) $\tilde{l}_{n,m} = l_{u,z}$ for all $m \in \{1, 2, \cdots, N\}, z \in \{1, 2, \cdots, I_u\}$, and for a given $u \in U^{n}_{train}$, where N and I_u are the total number of instances for new user n and previous user $u \in U^{n}_{train}$, respectively. This calculation helps to determine the cohesion within the cluster of instances sharing the same label. Accordingly, we obtain

$$D^u_{AID}(m) = \frac{D^u_{PID}(m) + D^u_{NID}(m)}{\sum_{z=1}^{I_u}(\mathbb{1}_{\{\tilde{l}_{n,m}=l_{u,z}\}}) + \sum_{z=1}^{N}(\mathbb{1}_{\{\tilde{l}_{n,m}=\tilde{l}_{n,z}\}}) - 1},$$

$$D^u_{PID}(m) = \sum_{z=1}^{I_u} \sqrt{\sum_k (v_{n,m}(k) - v_{u,z}(k))^2 \mathbb{1}_{\{\tilde{l}_{n,m}=l_{u,z}\}}},$$

and

$$D^u_{NID}(m) = \sum_{i=1}^{N} \sqrt{\sum_k (v_{n,m}(k) - v_{n,z}(k))^2 \mathbb{1}_{\{\tilde{l}_{n,m}=\tilde{l}_{n,z}\}}},$$

where $\mathbb{1}$ is identity function which returns 1 if its argument is true (0 otherwise). Note that this is computed for a given pair of new user instance $x_{n,m}$ (its corresponding feature vector $v_{n,m}$ is rather used) and previous user u. We call it as the Average Instance Distance (AID) denoted as D^u_{AID} which is summation of two terms, Previous User Instance Distance (PID) as D^u_{PID}, and New User Instance Distance (NID) as D^u_{NID} for a given previous user u. Its main purpose is to gauge the quality of the assigned label concerning previous user instances with the same label, thereby providing insight into the model's labeling proficiency.

In the second step of the silhouette score, we calculate the average distance between the new user instance $x_{n,m}$ and previous user instances with a *different* label: $\tilde{l}_{n,m} \neq l_{u,z}$. This step aims to assess whether the achieved clustering through the model's labeling aligns with expectations. Similar to the previous step, the average distance for each label $l_{u,z} \neq \tilde{l}_{n,m}$ is computed as explained earlier, and sorted in ascending order. The minimum average distance, denoted as $D^u_{AID_{min}}$, is then determined to evaluate the potential for creating a more appropriate label through clustering. The expectation is that this distance should be larger than D^u_{AID}, signifying $D^u_{AID_{min}} > D^u_{AID}$: so the model's labeling leads to a clustering that is distinctively different from instances with other labels, reinforcing the accuracy and confidence of the model's predictions.

After completing both steps, we proceed to calculate the overall silhouette score for given new user's instances $x_{n,m}$ and a previous user u. The Silhouette Score S^u_{silh} is defined for given new user's instances $x_{n,m}$ and a previous user u as

$$S^u_{silh} = \frac{1}{N} \sum_{m=1}^{N} \frac{D^u_{AID}(m) - D^u_{AID_{min}}(m)}{\max(D^u_{AID}(m), D^u_{AID_{min}}(m))},$$

which falls within the range $[-1, 1]$. A score of -1 indicates the highest confidence and 1 denotes the least confidence.

As explained above, the silhouette score is typically calculated by incorporating data from previous users. However, when selecting the initial predictions and comparing the quality of the adapted model's predictions with those from fine-tuned models, we must compute the silhouette score (S^n_{silh}) for the adapted model under a specific condition: previous user data should be excluded when calculating the silhouette score for this model.

This means that, for the adapted model, the silhouette score is computed as if no previous users' data were present. Specifically, we set $D^n_{PID}(m) = 0$ and $\sum_{z=1}^{I_u} \mathbb{1}_{\tilde{l}_{n,m}=l_{u,z}} = 0$, ensuring that the calculations are based solely on the current user's data.

4.4.9 Enhancements

Besides minimizing the mentioned loss $\mathcal{L}^{(r)}_{total}(\lambda)$, there are three important enhancements we additionally use for the second adaptation phase: Neighbor Selection, Instance Confidence, and Initialization of Pseudo labels.

4.4.9.1 Neighbor Selection

A critical part of the second adaptation phase is neighbor selection, because the local regularity loss $\mathcal{L}_{loc}$ heavily depends on the size and selection of neighbors from previous users U^n_{train} for the new user n. As also explained in [15], to properly determine the neighbor set for each new user instance, it is assumed that the correlation coefficients between the new user instance $x_{n,m}$ and its most related neighboring instances from previous users $x_{u,z}$ (which are typically high when the labels are same $\tilde{l}_{n,m} = l_{u,z}$) are significantly higher compared to other instances (with the same label $\tilde{l}_{n,m} = l_{u,z}$) that are loosely correlated. Namely, one can expect a significant drop in terms of the correlation coefficients from the highly correlated neighbor instances to the loosely correlated ones. When this large drop in correlation coefficients is observed, the neighbors before the drop are assigned to the neighbor list.

4.4.9.2 Instance Confidence

As we label the new user instances with different levels of certainty, instance confidence becomes a crucial factor for efficacy. Hence, we decide to dismiss instances with a positive (with threshold being 0) silhouette score during the adaptation of the network to enhance the model's overall confidence in subsequent steps. This strategy ensures that the adaptation process incorporates instances with confident labels, leading to an improvement in the model's overall accuracy. The rationale for this idea is rooted in the interpretation of silhouette scores: Instances with a positive silhouette score indicate proximity to a different cluster than the one assigned to them, making them more likely to be incorrectly classified. To ensure the accuracy of updates to the model, we exclusively utilize the pseudo labels of instances that receive positive silhouette scores. Moreover, to enhance the accuracy resulting from the λ selection of $w_r(\lambda)$ based on the silhouette score and increase the overall instance confidence, we consider augmenting the number of samples as an additional step. This can be achieved by selecting the nearest previous user instances and appending them to the new user instances to be labeled. Subsequently, the two steps described earlier are executed with double the number of instances. This augmentation contributes to an increase in the overall confidence of the labeling process. As stated in [15], there is a probability that the self adaptation term or the local regularity term may lose its functionality due to dismissing most instances (i.e. much more than a desired level) on the positive silhouette score basis. In such cases, one needs to update the threshold from 0 to a larger value for the adaptation process to continue.

4.4.9.3 Initialization of Pseudo labels

Recall that in both adaptation phases (first adaptation phase of outer / inner loops, and the second and final adaptation phase), our method uses the developing adapted network's own responses ($\overline{\Gamma}^{j}_{global}$ in the first phase and $\overline{\Gamma}^{w_r}_{global}$ in the second phase) as pseudo labels and evokes training by treating the pseudo labels as true labels, which works recursively across the iterations j, k and r. Hence, only initial pseudo labels are necessary for initialization. Using the responses of the randomly initialized very first network as the initial pseudo labels would obviously not be a good option. For the outer / inner loops of first adaptation phase, we use the responses of the global network trained on all previous users' data, which we consider is good enough.

On the other hand, for the performance boost we expect in particularly the second adaptation phase, initialization is another important issue, since the model typically yields better results in terms of rectifying misclassified instances with a better initial start. We would like to have a high quality initialization for the second adaptation phase as well.

Building on the previous model fine-tuning and adaptation phases, we already have fine-tuned, global and adapted (from the first adaptation phase) models available to produce predictions (as pseudo label initialization) at the beginning of the second and final adaptation phase for the new user data. For that, we propose an integrated approach that combines the silhouette score, with a new performance assessment score which evaluates the distances between features generated by the adapted model and those produced by each fine-tuned model. By calculating the average of these distances, we can determine the overall distance between each fine-tuned model and the adapted model of the new user. In doing so, our idea is to benefit from the predictions of the fined-tuned models of the previous users who might be statistically similar to the new user and so whose fined-tuned models might provide an enhancement. This could potentially help in producing more accurate pseudo label initialization when combined with the silhouette score. To be more precise, our approach involves creating a weighted confidence score that incorporates both the silhouette score and the feature distances. The mathematical formulation of this approach is as follows.

The distance score S^u_{dist} is calculated by averaging the dot products of features coming from the most recent adapted feature generator $\overline{f}^{j_{end}}_{global}$ and the features from the fine-tuned feature generators of a previous user's $f^{u,j_{end}}, u \in U^n_{train}$. Here, j_{end} represents the end of first adaptation phase, so represents the most recent information at the beginning of the second adaptation phase where psuedo label initialization takes effect. Mathematically, this can be expressed for a given previous user u and for each new user instance $x_{n,m}$ as

$$S^u_{dist} = \frac{1}{N} \sum_{m=1}^{N} \overline{f}^{j_{end}}_{global}(x_{n,m}) \cdot f^{u,j_{end}}(x_{n,m}).$$

Then the Overall Score S^u_{ovr} is a weighted combination of the silhouette score S^u_{silh} and the distance score S^u_{dist}, adjusted by a normalization factor γ, and given by

$$S^u_{ovr} = S^u_{silh} + \frac{1}{\gamma} S^u_{dist}.$$

In this formula, γ serves as a normalizing factor that is used to modulate the influence of S^u_{dist} on S^u_{ovr} for previous user u. When implementing this approach, it is

critical to ensure that both S^u_{silh} and S^u_{dist} are normalized or scaled appropriately to prevent scale discrepancies from skewing S^u_{ovr}. Additionally, the choice of γ may vary depending on the specific characteristics of the data and models involved, and it may require fine-tuning to achieve optimal performance in different scenarios. In our experiments with BENCH and BETA datasets, the value of γ is determined empirically and found to be around 20. This adjustment ensures that S^u_{silh} has a predominant effect on S^u_{ovr} while still allowing S^u_{dist} to contribute meaningfully to the final assessment.

By employing this refined method, we aim to enhance the accuracy and reliability of the model selection process, ultimately leading to the generation of higher-quality labels and improved model performance. After introducing the score that will be used as a metric to select networks for initial predictions, S^u_{ovr} can be calculated for each network.

In this way, we select the top five feature generators from the fine-tuned and most recent adapted models during the first adaptation phase, based on S^u_{ovr}. The class probabilities generated by the softmax layers of the selected models are then summed and averaged. The label with the highest probability is treated as a pseudo-label for each test instance $x_{n,m}$, aiming to provide more reliable labels for calculating $\mathcal{L}^{(t)}_{total}(\lambda)$.

Similar to the silhouette score calculation for the adapted model, the distance score S^n_{dist} cannot be computed in the same way as for models that include previous user data. This is because the term $f^{u,j_{end}}(x_{n,m})$ becomes $\overline{f}^{jend}_{global}(x_{n,m})$ for the adapted model, effectively making S^n_{dist} equal to 1. As a result, the contribution of S^n_{dist} loses significance in the overall score. To address this, S^n_{dist} is experimentally set to 0.05.

We propose an integrated approach to enhance model performance for label creation, combining the silhouette score with a novel method that evaluates the distances between features generated by a general model and those produced by fine-tuned models. S^u_{dist}, derived by averaging the dot product of these feature sets, quantifies the overall dissimilarity between models for a new user. Incorporating this score into S^u_{ovr}, alongside S^u_{silh}, yields a weighted confidence metric. This formula ensures that S^u_{silh} maintains prominence while allowing S^u_{dist} to contribute meaningfully, facilitated by a normalization factor, γ. Implementation considerations include appropriate normalization of both scores and fine-tuning γ. By leveraging this refined approach, we aim to improve model selection accuracy, leading to higher-quality labels and enhanced overall performance.

The overall process can be summarized as follows: Initially, a confidence score for

labeling is calculated by determining the average distance $D^u_{AID}(m)$ for instances with the same label. This provides an assessment of the confidence in the assigned label compared to other instances within the same cluster. Subsequently, the same method is applied to other labels, determining their respective average distances to assess whether alternative clusters exhibit higher confidence. This step evaluates the overall quality of clustering across different labels. A S^u_{silh}, which provides a quantifiable measure of confidence, is assigned to each instance $x_{n,m}$, calculated based on the confidence scores obtained earlier, where lower values indicate higher confidence while ranging between -1 and 1. A S^u_{ovr} is calculated by combining S^u_{silh} and S^u_{dist} to create the overall performance score. This process ensures that instances with confident labels contribute more significantly to the overall adaptation of the network, leading to an improved and more accurate model.

4.4.10 Algorithm

In this section, we present a comprehensive explanation of our algorithm, as seen in Alg. 1. The primary objective is to acquaint the reader with our algorithm and provide clear instructions for replicating our results.

4.5 Performance Evaluations

In this section, we present our extensive experiments with the datasets BENCH [20] and BETA [44]. All the details of these datasets are given in Section 4.4.1. We employ the metrics ITR and accuracy (along with the standard error bars) to quantify the proposed method's target character identification performance on the new user data. In our setup, for each dataset, users take turns to become new users. Namely, we separate one user from the others, and the separated user is designated as the new user, while the remaining users are designated as previous users. This defines one round for which we compute the ITR and accuracy, hence the mean ITR and mean accuracy across all rounds (35 rounds for BENCH and 70 for BETA) yield the final performance. In our experiments, we compare our method with other prominent calibration-free techniques from the literature, as categorized in the 4.2. For our comparison, we include DA methods such as Source-Free DA [15], OACCA

Algorithm 1: SSVEP Classification with DA

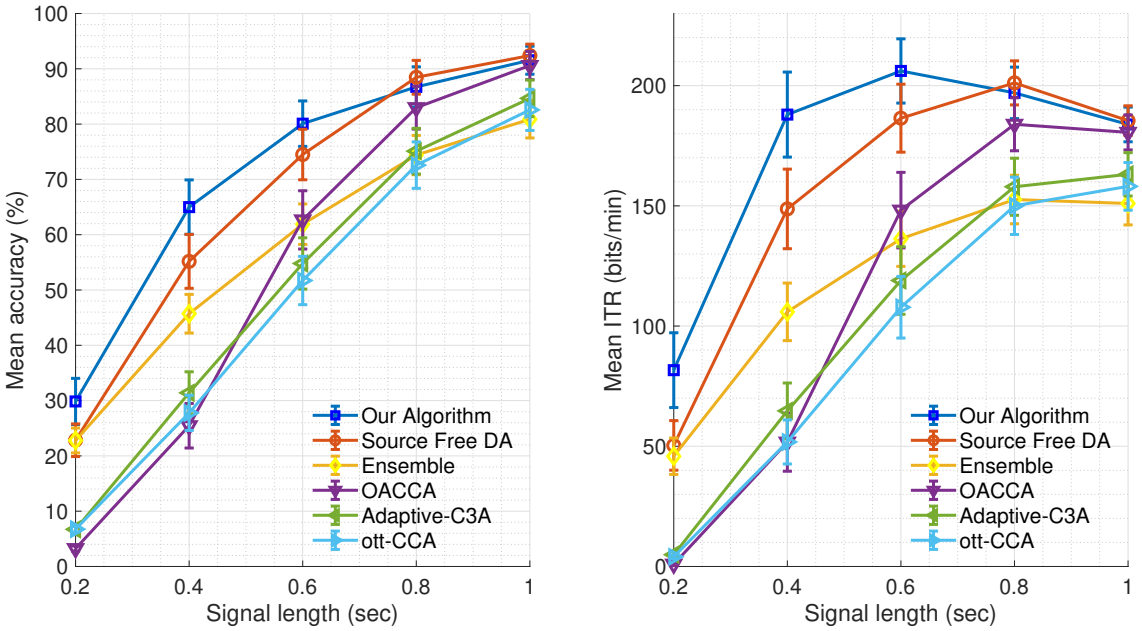
Data: $x_{i,m}$, for each user $i \in U$ including the all the previous users and the new user.

Result: $\tilde{l}_{n,m}$, labels for each signal for the new user n

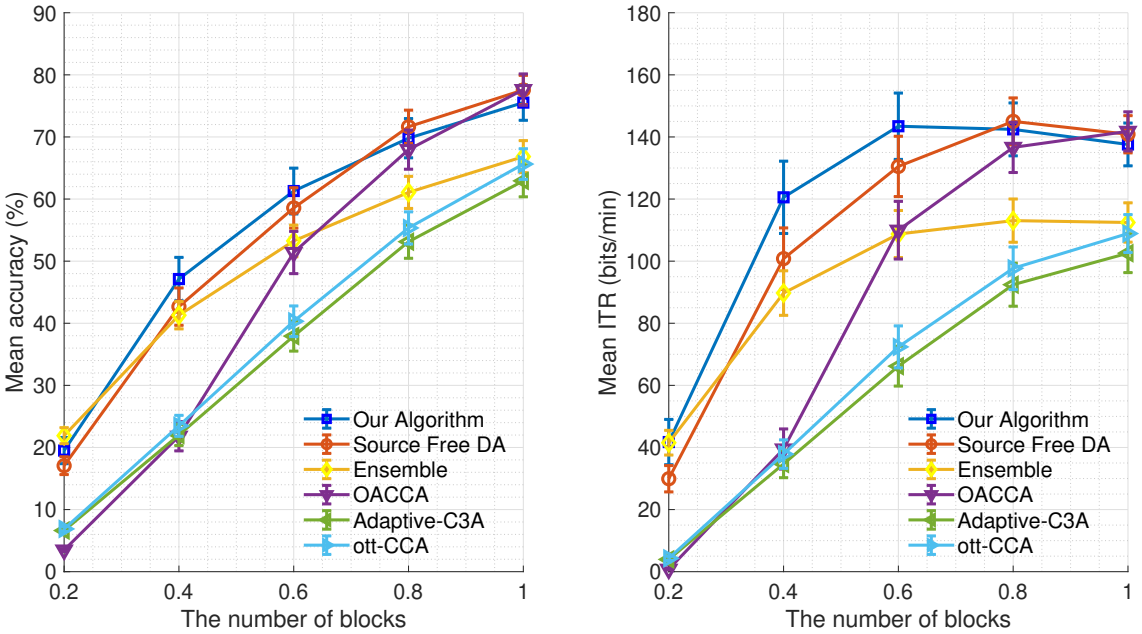
Train a global model, $\overline{\Gamma}^{n}_{global}$, on X where for each loop n is the new user in U

for $n \in U$ **do**

- Run the first adaptation phase
- Initialize $\overline{\Gamma}^{n,j}_{global} = \Gamma^{n}_{global}$, set iteration count $j = 0$.
- **while** $\tilde{l}^{j}_{n,m} \neq \tilde{l}^{j+1}_{n,m}$ **do**
  - **for** $i \in U^{n}_{train}$ **do**
    - Fine-tune the feature extractor $f^{i,j}$ from Γ_{global} using previous user i's signals, X_i
  - **for** $i \in U_{train}$ **do**
    - Extract features $v_{i,m} = f^{i,j}(x_{i,m})$
  - Train a new classifier h^{j}_{global} using extracted features $\{v_{i,m}\}_{i \in U_{train}}$ from features generated from the previous loop.
  - **for** $i \in U^{n}_{train}$ **do**
    - Fine-tuned models $\Gamma^{i,j}$ are fine tuned once more with the corresponding user's signals, $x_{i,m}$ after replacing the fine-tune classifier h^{j} with the global classifier $h^{n,j}_{global}$ and freezing it.
  - Initialize the feature generator $\overline{\Gamma}^{n,j}_{global}$ by combining $\overline{f}^{n,j}_{global}$ and the adapted classifier $h^{n,j}_{global}$ into FC layer.
  - Produce predictions $\tilde{l}_{n,m}$ of new user features $v_{n,m}$ coming from the feature extractor of $\overline{\Gamma}^{n,j}_{global}$.
  - The model $\overline{\Gamma}^{n,j}_{global}$ is trained using predictions $\tilde{l}_{n,m}$ as true labels by freezing $\overline{h}^{n,j}_{global}$, while X_n is presented as test data.
  - Produce new predictions $\tilde{l}_{n,m}$ of test features $v_{n,m}$ coming from the feature extractor of newly adapted $\overline{\Gamma}^{n,j}_{global}$
- Run the second adaptation phase
- Set iteration count $j = 0$.
- **for** $\lambda \in \Lambda$ **do**
  - Set the distance score S^{n}_{dist} to 0.2 for the new user n.
  - **for** $i \in U_{train}$ **do**
    - Calculate the silhouette score S^{i}_{silh} and distance score S^{i}_{dist} for each $\Gamma^{i,j}$ and $\overline{\Gamma}^{n,j}_{global}$
    - Combine silhouette score and distance score to calculate the overall score S^{i}_{ovr}
    - Optimize each network $\overline{\Gamma}^{n,j}_{global}$



(a) BENCH [20] Results. On the left is the accuracy vs. signal length and on the right is ITR vs. signal length



(b) BETA [44] Results. On the left is the accuracy vs. signal length and on the right is ITR vs. signal length

Figure 4.2 (a) An assessment of performance across our algorithm, Source Free DA [15], Ensemble [16], online adaptive CCA (OACCA) [17], adaptive combined-CCA (Adaptive-C3A) [18], and online transfer template CCA (ott-CCA) [19], depicted in the left panel, examines their mean accuracy across various signal lengths (sec) within BENCH [20] Dataset. Meanwhile, the right panel presents a separate performance comparison, focusing on their Information Transfer Rates (ITRs) across various signal lengths (sec) within the same BENCH [20] Dataset. (b) The same performance evaluation is replicated using the BETA dataset , as presented in (a). Error bars in (a) and (b) represent standard error.

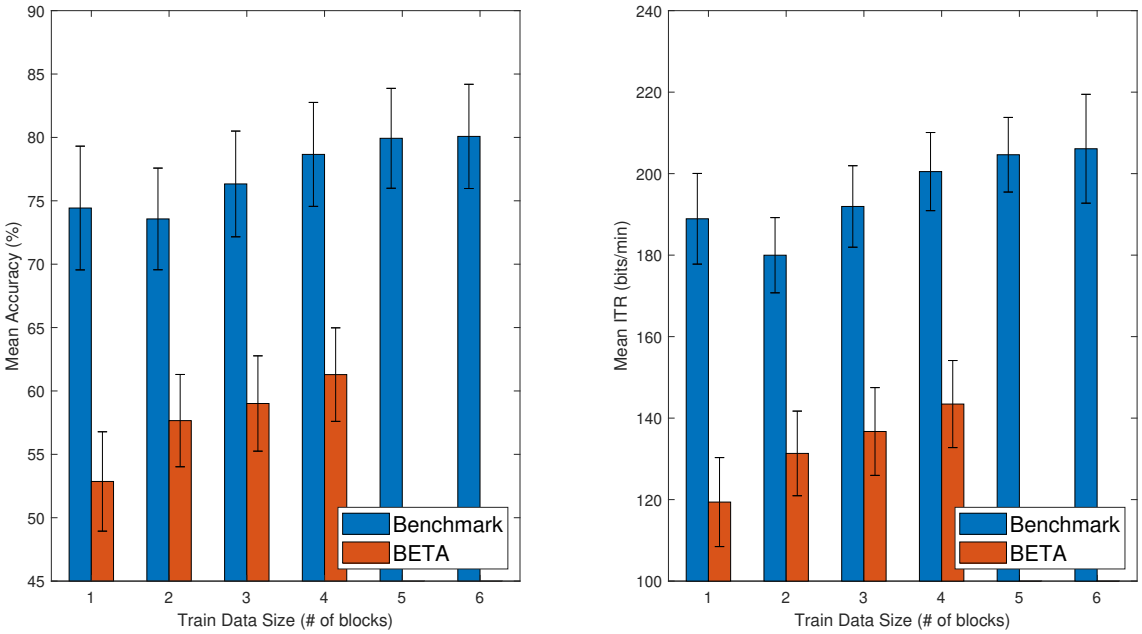


Figure 4.3 In the above figure, we observe the impact of adding new user data blocks with pseudo labels to the training process on both BENCH and BETA datasets, specifically in terms of Mean Accuracy and Mean ITR. As expected, as the number of data blocks from the new user increases—thereby adding more unlabeled user data—the performance of our method improves accordingly.

[17], and Adaptive-C3A [18], as well as DG methods like the ensemble technique (Ensemble) [16] and OTT-CCA [19]. Additionally, we consider completely training-free methods such as those proposed in [274; 275]. This comprehensive comparison allows us to evaluate the effectiveness of our method relative to these established calibration-free approaches.

4.5.1 Results

As shown in Fig. 4.2 (upper row) for the dataset BENCH, our findings highlight the superior performance of our method compared to the existing prominent DA techniques, as we achieve the highest ITR / accuracy results and strongly outperform all the others especially for shorter signal durations (i.e., 0.4, 0.6 seconds). For longer signal durations 0.8 and 1 seconds, our method performs similarly with its most successful competitor Source Free DA [15] while still outperforming the others by a significant margin. As for the BETA dataset (Fig. Fig. 4.2 lower row), again our method and Source Free DA are generally strongly superior over all the others. Since the dataset BETA is noisier, the performance difference between ours and Source Free DA seems to be insignificant at the low signal duration 0.2 seconds; and

it is also insignificant for the longer durations (0.8 and 1 seconds) as both methods converge and achieve the saturation. However, at 0.4 and 0.6 seconds (considering the accuracy and ITR together), our method outperforms Source Free DA. When taking into account all the signal durations, our method is the one that achieves the highest ITR overall for both datasets. Fig. 4.2 shows the highest ITRs of 207.54 bits/min (BENCH) and 145.05 bits/min (BETA) when utilizing a signal length of 0.6 seconds with our method, surpassing the closest-performing method Source Free DA with ITRs of 201.15 bits/min (BENCH) and 145.02 bits/min (BETA), respectively.

Of significant note, our method demonstrates its most substantial contribution with a remarkable 47% increase in ITR when the signal length is reduced to 0.2 seconds on the BENCH dataset and a 19% increase in ITR when the signal length is set to 0.4 seconds on the BETA dataset. However, it's important to highlight that our method does not maintain its dominance and converges to the performance of Source Free DA for signal lengths of 0.8 seconds and 1 second (for both datasets and metrics ITR / accuracy, but Source free DA is slightly better in the case of the BETA dataset).

	BENCH Dataset			BETA Dataset		
Sig. Len.	1^{st} Step	2^{nd} Step	3^{rd} Step	1^{st} Step	2^{bd} Step	3^{rd} Step
0.2	21.75%	28.28%	29.42%	19.44%	20.66%	22.12%
0.4	38.01%	62.05%	65.72%	34.28%	46.84%	46.64%
0.6	51.01%	76.84%	80.11%	43.78%	59.35%	61.76%
0.8	63.94%	83.83%	87.94%	51.28%	66.90%	69.41%
1.0	71.32%	88.34%	92.34%	60.37%	70.63%	75.87%

Table 4.1 The performance results at the conclusion of the first adaptation phase are evaluated for different signal lengths, with $f \in \{0.2, 0.4, 0.6, 0.8, 1\}$. These results are reported at three key stages: at the end of the initial training loop(1^{st} Step), after the first adaptation loop (2^{nd} Step), and finally, at the end of all adaptation loops (3^{rd} Step). This progression highlights how performance evolves as the signal length and adaptation phases advance.

In Table 4.1, we present the results of the adaptations and the performance improvements after each iteration. The table highlights significant progress, with the unadapted network showing substantial improvement even after the first adaptation loop. By the final stage, we observe that the network has converged in terms of performance, indicating that further iterations yield minimal gains.

To further demonstrate that adding new user data with pseudo labels during iterative training enhances the model's overall performance, we conducted an additional experiment where we systematically increased the amount of new user data by incrementally adding more data blocks. This also simulates the actual scenario in real life, where the new user provides an accumulation of unlabeled data as s/he uses the BCI speller system. As shown in Fig. 4.3, we observe that with each new block

added, the overall performance measured on both the BENCH and BETA datasets consistently improves. However, we also noted that the performance boost diminishes slightly. For example, in the BENCH dataset, the addition of the second block results in a performance that is similar to, or slightly lower than, the performance with one block. This could be due to the second block not contributing new insights that significantly impact overall performance. Interestingly, when a third block is added, the performance once again surpasses that of the experiment with a single block. Importantly, adding new pseudolabeled data does not degrade the model's performance; instead, it either maintains or enhances it. This observation supports our hypothesis that including new user data during training iterations positively impacts the model's performance, thereby validating our approach.

BENCH Dataset			BETA Dataset		
T	Mean Accuracy	ITR (bits/min)	T	Mean Accuracy	ITR (bits/min)
ovr	80.11± 4.31	207.54± 13.84	*ovr*	61.76±3.69	145.05±10.81
silh	78.95± 4.66	204.87± 10.50	*silh*	60.43± 3.70	140.78±10.68
last	78.08± 4.70	201.67± 10.59	*last*	52.65± 4.09	120.57±11.43
rand	80.49± 4.28	208.95± 9.76	*rand*	62.01± 3.53	144.25±10.35

Table 4.2 This table summarizes the results of applying our scoring method. The outcomes are grouped based on different scoring strategies: users with the highest overall scores (*ovr*), users ranked by silhouette scores alone (*silh*), users with the lowest overall scores (*last*), and finally, randomly selected users (*rand*). These comparisons highlight the effectiveness of our scoring system in identifying the most relevant users.

In the table above, we compare the results of selecting models from which we generate the initial pseudo labels. In this table, *ovr* refers to models selected based on the overall S_{ovr} score, while *silh* indicates models chosen based solely on the highest silhouette score. The term *last* represents models with the lowest S_{ovr} scores, and *rand* corresponds to randomly selected models.

The purpose of this comparison is to demonstrate the effectiveness of our scoring method and its impact on overall performance. As shown, our method consistently outperforms all other combinations, leading to a noticeable performance boost.

It's important to note that the lower efficiency of the random model selection may be influenced by the fact that in the *last* scenario, different models are selected for each value of λ, whereas the same models are used across all λ values in the *rand*

scenario.

As observed, the use of pseudo labels significantly improves model performance. This improvement results from leveraging previous user data, specifically selecting the user with the most similar properties to the new user. Additionally, our overall score ensures a high level of confidence in generating these pseudo labels, enabling the method to build an optimal model that excels in the task of character recognition.

This result is expected, as the pseudo labels increasingly resemble actual labels, approaching the performance limit of training with true labels. By building the model from confident initial predictions and progressively improving accuracy with each iteration, our method outperforms both fine-tuned and general models.

An important observation is the performance gap between the highest and lowest scoring samples. As shown in Table 4.2, the significant difference in scores demonstrates the effectiveness of our scoring method. By applying this approach, we can easily differentiate between related and unrelated users. Additionally, we observe that random selections outperform the lowest-scoring choices, further validating that our scoring system ranks users from most to least related in a meaningful way.

4.6 Discussion

In this chapter, we introduced a novel SSVEP-based BCI speller classifier designed to predict on unlabeled new user data by leveraging existing labeled datasets from prior users. Our approach employed an iterative fine-tuning process to enhance overall accuracy by generating and incorporating the model's own predictions as pseudolabels. This iterative process eliminated the need for new user calibration by combining prior user calibration data with the unlabeled new user data. The model's adaptation was self-regularizing, relying on its own pseudolabels for refinement. Additionally, we utilized a combination of silhouette and distance scores to fine-tune the model using data from the most similar prior users and to generate pseudolabels during the final adaptation cycle. The objective was to adapt the model to deliver accurate predictions for new users by leveraging both labeled data and pseudolabels.

We conducted extensive experiments using the BETA and BENCH datasets. In these experiments, we designated one user as the new user (without labels) and created a separate model for each user to evaluate performance individually. The

results showed that for both datasets, the classifier achieved remarkable classification accuracy and SOTA ITR results with shorter signal lengths. However, the rate of improvement diminished with longer signals, likely due to info saturation.

To address this limitation, we proposed incorporating more effective similarity metrics to better classify users and instances more closely related to the new user. Expanding this work could improve classification performance, particularly for longer signals.

5. Conclusion

In this thesis, we explored the fine-tuning of global models through two distinct approaches tailored to the properties of the data. In the first part, we introduced CTBAD, a method designed for hierarchical datasets that used context trees to partition video frames for anomaly detection. CTBAD facilitated the fine-tuning of global models by leveraging these partitions to focus on locational anomalies with spatially nonstationary statistics, even when the number of training samples was limited. By combining models of varying complexity and locality, it overcame the limitations of relying solely on global scene statistics. This method demonstrated robust performance across diverse scenes and actors, making it highly suitable for real-world video surveillance and easily integrable with SOTA techniques. However, CTBAD faced challenges with spatially stationary anomalies and mismatches between training and testing conditions, such as when certain image regions were inactive during training but exhibited activity during testing. To address these limitations, future work will focus on automating the tuning of the β parameter to reduce manual effort and integrating change detection mechanisms to dynamically select well-trained models during testing. Additional research directions includes developing adaptive partitioning strategies and refining the loss function to better handle limited data and stationary statistics.

The second part of this thesis focused on non-hierarchical data, where directly applying context trees to fine-tune global models was not feasible due to the data's non-hierarchical nature. Instead, similarity measures were employed to identify patterns between previous users and new users for model adaptation through fine-tuning. This approach was demonstrated in SSVEP-based BCI spellers, which enabled communication for individuals with disabilities. By integrating labeled data from previous users and pseudolabels for new users, the system iteratively fine-tuned the model to adapt to new users, significantly improving classification accuracy, particularly for shorter signal lengths. Future research addresses the limitations of this approach for longer signal lengths, where the benefits of shared information diminished due to the increased complexity of the signals. A promising direction involves

incorporating advanced similarity metrics to refine the loss function, potentially enhancing classification performance for extended signals. Additionally, techniques can be developed to optimize the method for scenarios involving longer signal durations, further improving its effectiveness. These efforts aim to expand the method's applicability and ensure robust performance across various signal lengths and domains.

BIBLIOGRAPHY

[1] Yin Zhu, Yuqiang Chen, Zhongqi Lu, Sinno Pan, Gui-Rong Xue, Yong Yu, and Qiang Yang. Heterogeneous transfer learning for image classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 25, pages 1304–1309, 2011.

[2] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[3] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 4. Granada, 2011.

[4] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[5] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Flownet 2.0 Brox. Evolution of optical flow estimation with deep networks. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 2462–2470, 2017.

[6] K. Doshi and Y. Yilmaz. Any-shot sequential anomaly detection in surveillance videos. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition Workshops*, pages 934–935, 2020.

[7] Y. Ouyang and V. Sanchez. Video anomaly detection by estimating likelihood of representations. *2020 25th International Conference On Pattern Recognition (ICPR)*, pages 8984–8991, 2021.

[8] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.

[9] S. Bansod and A. Nandedkar. Detection and localization of anomalies from videos based on optical flow magnitude and direction. *ICCASP*, page 1, 2017.

[10] C. Lu, J. Shi, and J. Jia. Abnormal event detection at 150 fps in matlab. *Proceedings Of The IEEE International Conference On Computer Vision*, pages 2720–2727, 2013.

[11] W. Liu, W. Luo, D. Lian, and S. Gao. Future frame prediction for anomaly detection–a new baseline. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 6536–6545, 2018.

[12] Bharathkumar Ramachandra and Michael Jones. Street scene: A new dataset and evaluation protocol for video anomaly detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2569–2578, 2020.

[13] Keval Doshi and Yasin Yilmaz. Any-shot sequential anomaly detection in surveillance videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 934–935, 2020.

[14] Yuqi Ouyang and Victor Sanchez. Video anomaly detection by estimating likelihood of representations. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 8984–8991. IEEE, 2021.

[15] Osman Berke Guney, Deniz Kucukahmetler, and Huseyin Ozkan. Source-free domain adaptation for ssvep-based brain-computer interfaces, 2023. URL https://arxiv.org/abs/2305.17403.

[16] Osman Berke Guney and Huseyin Ozkan. Transfer learning of an ensemble of dnns for ssvep bci spellers without user-specific training. *Journal of Neural Engineering*, 20(1):016013, 2023.

[17] Chi Man Wong, Ze Wang, Masaki Nakanishi, Boyu Wang, Agostinho Rosa, CL Philip Chen, Tzyy-Ping Jung, and Feng Wan. Online adaptation boosts ssvep-based bci performance. *IEEE Transactions on Biomedical Engineering*, 69(6):2018–2028, 2021.

[18] Nicholas R Waytowich, Josef Faller, Javier O Garcia, Jean M Vettel, and Paul Sajda. Unsupervised adaptive transfer learning for steady-state visual evoked potential brain-computer interfaces. In *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 004135–004140. IEEE, 2016.

[19] Peng Yuan, Xiaogang Chen, Yijun Wang, Xiaorong Gao, and Shangkai Gao. Enhancing performances of ssvep-based brain–computer interfaces via exploiting inter-subject information. *Journal of neural engineering*, 12(4):046006, 2015.

[20] Yijun Wang, Xiaogang Chen, Xiaorong Gao, and Shangkai Gao. A benchmark dataset for ssvep-based brain–computer interfaces. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 25(10):1746–1752, 2016.

[21] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[22] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[23] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[24] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep learning. In *MIT Press*, 2016. URL https://www.deeplearningbook.org/. Chapter 7: Regularization for Deep Learning.

[25] Sinno Jialin Pan and Qiang Yang. Transfer learning via instance weighting. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence*, pages 1200–1205, 2010.

[26] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *Proceedings of the 30th Conference on Neural Information Processing Systems (NeurIPS)*, pages 3303–3311, 2016. URL https://arxiv.org/abs/1606.04671.

[27] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5542–5550, 2017.

[28] Zihan Liu, Shuyuan Zhu, Jingkuan Song, and Nicu Sebe. Contrastive learning for cold-start recommendation. *arXiv preprint arXiv:2107.05315*, 2021. URL https://arxiv.org/abs/2107.05315.

[29] Steven CH Hoi, Doyen Sahoo, Jing Lu, and Peilin Zhao. Online learning: A comprehensive survey. *Neurocomputing*, 459:249–289, 2021.

[30] Frans MJ Willems, Yuri M Shtarkov, and Tjalling J Tjalkens. The context-tree weighting method: Basic properties. *IEEE transactions on information theory*, 41(3):653–664, 1995.

[31] Huseyin Ozkan, N Denizcan Vanli, and Suleyman S Kozat. Online classification via self-organizing space partitioning. *IEEE Transactions on Signal Processing*, 64(15):3895–3908, 2016.

[32] N. Vanli, M. Sayin, M. Mohaghegh, H. Ozkan, and S. Kozat. Nonlinear regression via incremental decision trees. *Pattern Recognition*, 86:1–13, 2019.

[33] Başarbatu Can and Hüseyin Özkan. Active learning for online nonlinear neyman-pearson classification. In *2022 30th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2022.

[34] Başarbatu Can and Hüseyin Özkan. Neyman-pearson classification via context trees. In *2020 28th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2020.

[35] M. Kerpicci, H. Ozkan, and S. Kozat. Online anomaly detection with bandwidth optimized hierarchical kernel density estimators. *IEEE Transactions On Neural Networks And Learning Systems*, 32:4253–4266, 2020.

[36] Başarbatu Can. *Online anomaly detection in the Neyman-Pearson hypothesis testing framework*. PhD thesis, Sabanci University, 2022.

[37] Angela A. Sodemann, Mark P. Ross, and Benjamin J. Borghetti. A review of anomaly detection in automated surveillance. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1257–1272, 2012. doi: 10.1109/TSMCC.2012.2215319.

[38] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. *arXiv preprint arXiv:1801.04264*, 2018. Proposes a Multiple Instance Learning framework for detecting anomalies in weakly labeled video data.

[39] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8):1177–1193, 2012.

[40] J. Tan, Z. Aijun, X. Wang, M. Cheng, and Q. Yang. A survey on transfer learning. In *Proceedings of the 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 130–134. IEEE, 2018.

[41] M Schneider. Brain-computer interfaces: Principles and practice, eds jr wolpaw and ew wolpaw, 2012.

[42] John P Donoghue. Bridging the brain to the world: a perspective on neural interface systems. *Neuron*, 60(3):511–521, 2008.

[43] Yijun Wang, Xiaorong Gao, Bo Hong, Chuan Jia, and Shangkai Gao. Brain-computer interfaces based on visual evoked potentials. *IEEE Engineering in medicine and biology magazine*, 27(5):64–71, 2008.

[44] Bingchuan Liu, Xiaoshan Huang, Yijun Wang, Xiaogang Chen, and Xiaorong Gao. Beta: A large benchmark database toward ssvep-bci application. *Frontiers in neuroscience*, 14:627, 2020.

[45] Soner Özgün Pelvan, Başarbatu Can, and Hüseyin Özkan. Anomaly detection with false alarm rate controllable classifiers. In *2023 31st Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2023.

[46] Soner Ö Pelvan, Basarbatu Can, and Huseyin Ozkan. A hierarchical approach for improved anomaly detection in video surveillance. *IEEE Access*, 2023.

[47] F. Willems, Y. Shtarkov, and T. Tjalkens. The context-tree weighting method: Basic properties. *IEEE Transactions On Information Theory*, 41:653–664, 1995.

[48] Suleyman S Kozat, Andrew C Singer, and Georg Christoph Zeitler. Universal piecewise linear prediction via context trees. *IEEE Transactions on Signal Processing*, 55(7):3730–3745, 2007.

[49] Osman Berke Guney, Muhtasham Oblokulov, and Huseyin Ozkan. A deep neural network for ssvep-based brain-computer interfaces. *IEEE Transactions on Biomedical Engineering*, 69(2):932–944, 2022. doi: 10.1109/TBME.2021.3110440.

[50] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[51] Richa Upadhyay, Ronald Phlypo, Rajkumar Saini, and Marcus Liwicki. Sharing to learn and learning to share–fitting together meta-learning, multi-task learning, and transfer learning: A meta review. *arXiv preprint arXiv:2111.12146*, 2021.

[52] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton Van Den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[53] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[54] Jing Ren, Feng Xia, Yemeng Liu, and Ivan Lee. Deep video anomaly detection: Opportunities and challenges. In *2021 international conference on data mining workshops (ICDMW)*, pages 959–966. IEEE, 2021.

[55] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[56] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhavoronkov, and et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[57] Waqas Sultani, Chen Chen, and Mubarak Shah. Real-world anomaly detection in surveillance videos. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6479–6488, 2018.

[58] Edward Verenich, Alvaro Velasquez, MG Murshed, and Faraz Hussain. The utility of feature reuse: Transfer learning in data-starved regimes. *arXiv preprint arXiv:2003.04117*, 2020.

[59] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. In *arXiv preprint arXiv:1704.04861*, 2017.

[60] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. *Communications Of The ACM*, 60:84–90, 2017.

[61] Peiguo Jiang, Yu Chen, Baohua Liu, Dongjian He, and Chunjiang Liang. Real-time detection of apple leaf diseases using deep learning approach based on improved convolutional neural networks. *IEEE Access*, 8:20796–20805, 2020.

[62] Amir R. Zamir, Alexander Sax, William B. Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3712–3722, 2018.

[63] Christos Matsoukas, Johan Fredin Haslum, Moein Sorkhei, Magnus Söderberg, and Kevin Smith. What makes transfer learning work for medical images: Feature reuse & other factors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9225–9234, 2022.

[64] Ahmad Waleed Salehi, Shakir Khan, Gaurav Gupta, Bayan Ibrahimm Alabduallah, Abrar Almjally, Hadeel Alsolai, Tamanna Siddiqui, and Adel Mellit. A study of cnn and transfer learning in medical imaging: Advantages, challenges, future scope. *Sustainability*, 15(7), 2023. ISSN 2071-1050. URL https://www.mdpi.com/2071-1050/15/7/5930.

[65] Rashmiranjan Nayak, Umesh Chandra Pati, and Santos Kumar Das. A comprehensive review on deep learning-based methods for video anomaly detection. *Image and Vision Computing*, 106:104078, 2021. ISSN 0262-8856. doi: https://doi.org/10.1016/j.imavis.2020.104078. URL https://www.sciencedirect.com/science/article/pii/S0262885620302109.

[66] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[67] Jeremy Howard and Sylvain Gugger. *Deep Learning for Coders with Fastai and PyTorch*. O'Reilly Media, 2020.

[68] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information fusion*, 76: 243–297, 2021.

[69] Md Zahangir Alom, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal, and Vijayan K Asari. A state-of-the-art survey on deep learning theory and architectures. *electronics*, 8(3):292, 2019.

[70] Mikhail V Koroteev. Bert: a review of applications in natural language processing and understanding. *arXiv preprint arXiv:2103.11943*, 2021.

[71] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7482–7490, 2017.

[72] Wouter M Kouw and Marco Loog. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv:1812.11806*, 2018.

[73] Xiaofeng Liu, Chaehwa Yoo, Fangxu Xing, Hyejin Oh, Georges El Fakhri, Je-Won Kang, Jonghye Woo, et al. Deep unsupervised domain adaptation: A review of recent advances and perspectives. *APSIPA Transactions on Signal and Information Processing*, 11(1), 2022.

[74] Y. Ganin and V. Lempitsky. Domain-adversarial training of neural networks. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 1180–1189. PMLR, 2016.

[75] K. Saito, K. Watanabe, and K. Saenko. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 369–384, 2018.

[76] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[77] & Gool L. V. Bay H., Tuytelaars T. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2006.

[78] P. J. Burt and E. H. Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983.

[79] W. Hoyer. Image pyramids for multiresolution analysis and image processing. In *Proceedings of the IEEE*, volume 92, pages 1056–1068, 2004.

[80] & Harwood D. Ojala T., Pietikäinen M. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

[81] & Wu M. Zhang L., Zhang D. A comparative study of local contrast normalization algorithms for image processing. *Pattern Recognition Letters*, 30(4): 345–353, 2009.

[82] & Morel J.-M. Buades A., Coll B. A non-local algorithm for image denoising. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 60–65, 2005.

[83] Y. Chen, Y. & Wu. Patch-based image restoration: A new approach to image denoising. *Journal of Visual Communication and Image Representation*, 21 (2):104–117, 2010.

[84] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[85] He K. Gkioxari G. Dollár P. & Girshick R. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.

[86] E. Ahmed, I. Hussain, and A. Mian. A survey on anomaly detection in video surveillance. *ACM Computing Surveys (CSUR)*, 50(1):1–34, 2016.

[87] Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. *arXiv preprint arXiv:1510.01553*, 2015.

[88] Y. Zeng, Y. Li, and J. Li. Abnormal event detection in video surveillance by a novel 3d deep learning framework. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 829–837, 2019.

[89] B. Ghanem, A. Gaidon, and N. Sunderhauf. Anomaly detection in crowded scenes using local motion features. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 241–249, 2017.

[90] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[91] Ruixuan Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Characterizing and avoiding negative transfer. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11293–11302, 2019.

[92] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3:1–40, 2016.

[93] Lei Wang and Wei Hu. Negative transfer in fine-tuning pre-trained language models for legal text classification. In *Proceedings of the 14th International Conference on Knowledge Science, Engineering and Management*, pages 21–30. Springer, 2021.

[94] Sebastian Ruder. Neural transfer learning for natural language processing: A review. *arXiv preprint arXiv:1901.11504*, 2019.

[95] Xu Tan, Di Wu, Xuchu Deng, Song Zhang, Xin Jiang, Zhifang Zhao, and Ruifeng Zhao. Otce: A unified framework for overcoming negative transfer in pre-trained model fine-tuning. *Neural Networks*, 140:1–12, 2021.

[96] Chen J., Zhang H., J. Liu, Wu M., and Yao W. Transfer learning for medical image analysis: A survey. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1148–1155. IEEE, 2020.

[97] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90. URL https://doi.org/10.1109/CVPR.2016.90.

[98] Yair Lifshitz, Erez Farkash, Ido Zukerman, and Shai Shalev-Shwartz. Adaptive learning rate for transfer learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, pages 3850–3860, 2018.

[99] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, 1:328–339, 2018.

[100] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Rethinking the value of network pruning. *International Conference on Learning Representations (ICLR)*, 2020.

[101] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwińska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the National Academy of Sciences (PNAS)*, pages 3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://www.pnas.org/content/114/13/3521.

[102] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. *Advances in Neural Information Processing Systems (NeurIPS)*, 32:3342–3352, 2019.

[103] Nima Tajbakhsh, Jianming Yang, Zhongyi Liang, Xiaoqin Jiang, Gregory R. W. Bogunovic, Mehran Shah, Kenneth J. Wu, Xun Xu, and Daniel L. Rubin. Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312, 2016. doi: 10.1109/TMI.2016.2535302. URL https://doi.org/10.1109/TMI.2016.2535302.

[104] Connor Shorten and Taghi M. Khoshgoftaar. A survey on image data augmentation for deep learning. In *Journal of Big Data*, volume 6, pages 1–48, 2019. doi: 10.1186/s40537-019-0197-0. URL https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0.

[105] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Deep learning algorithms for domain adaptation. *Foundations and Trends in Machine Learning*, 10(3-4):223–407, 2018. doi: 10.1561/2200000055. URL https://doi.org/10.1561/2200000055.

[106] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Proceedings of the 28th International Conference on Neural Information Processing Systems (NeurIPS)*, pages 1–9, 2015.

[107] Mark Sandler, Andrew G Howard, Menglong Zhu, Alex Zhmoginov, and Liang Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. *arXiv preprint arXiv:1801.04381*, 2018.

[108] Anwar K. Bhat and Sanjay Jain. Transfer learning: A review. *International Journal of Computer Applications*, 975:8887, 2019.

[109] Md Abdur Ahsan and Md Iqbal Hossain. A survey on deep learning techniques for ecg signal processing. *Journal of Biomedical Informatics*, 109:103533, 2020. doi: 10.1016/j.jbi.2020.103533.

[110] Yifan Wang and Zheng Zhang. Transfer learning for out-of-distribution generalization: A survey. *arXiv preprint arXiv:1909.13664*, 2019.

[111] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. URL http://arxiv.org/abs/1506.02640.

[112] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Joshua Susskind, Wenda Wang, and Russell Webb. Learning from simulated and unsupervised images through adversarial training. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2107–2116, 2017.

[113] Chang Wang and Sridhar Mahadevan. Heterogeneous domain adaptation using manifold alignment. In *IJCAI proceedings-international joint conference on artificial intelligence*, volume 22, page 1541. Citeseer, 2011.

[114] Lixin Duan, Dong Xu, and Ivor Tsang. Learning with augmented features for heterogeneous domain adaptation. *arXiv preprint arXiv:1206.4660*, 2012.

[115] Brian Kulis, Kate Saenko, and Trevor Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR 2011*, pages 1785–1792. IEEE, 2011.

[116] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *CoRR*, abs/1911.02685, 2019. URL http://arxiv.org/abs/1911.02685.

[117] Maxime Oquab, Léon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Computer Vision and Pattern Recognition*, pages 3320–3327, 2014.

[118] Zhongqi Lu, Erheng Zhong, Lili Zhao, Evan Wei Xiang, Weike Pan, and Qiang Yang. Selective transfer learning for cross domain recommendation. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 641–649. SIAM, 2013.

[119] Joey Zhou, Sinno Pan, Ivor Tsang, and Yan Yan. Hybrid heterogeneous transfer learning through deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 28, 2014.

[120] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *Proceedings of the 16th International Conference on Machine Learning*, pages 200–209. Morgan Kaufmann, 1999.

[121] Olivier Chapelle and Andreas Zien. Semi-supervised classification by low density separation. In *Proceedings of the 24th International Conference on Machine Learning*, pages 57–64. ACM, 2005.

[122] M. Long, J. Cao, J. Wang, and J. Sun. Learning transferable features with deep adaptation networks. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 97–105. PMLR, 2015.

[123] X. Zhuang, C. Wang, S. Qiu, and et al. Self-supervised learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1121–1138, 2022.

[124] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y. Ng. Self-taught learning: Transfer learning from unlabeled data. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, pages 759–766. Omnipress, 2007.

[125] W. Dai, Q. Yang, Y. Xu, and Y. Yu. Instance-based transfer learning with svm. In *Proceedings of the 21st National Conference on Artificial Intelligence*, volume 2, pages 376–381, 2007.

[126] M. Wang, Z. Jiang, Q. Yu, and Y. Zhang. Instance-based transfer learning for knowledge-intensive applications. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 1231–1237, 2019.

[127] J. Chen, Z. Xu, and D. Zhao. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, pages 1285–1294, 2012.

[128] M. Long, J. Wang, and P. S. Yu. Asymmetric feature mapping for domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 703–711, 2015.

[129] J. Zhou, J. Liu, and X. Wang. Deep symmetric neural networks for transfer learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2152–2160, 2019.

[130] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. Transfer learning for image classification with deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[131] I. Goodfellow, J. Pouget-Abadie, M. Mirza, and et al. Generative adversarial networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, 2014.

[132] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial domain adaptation via domain discrepancy. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7167–7176, 2018.

[133] X. Liu, H. Xu, Y. Zhang, and et al. Adversarial learning for semi-supervised domain adaptation. In *Proceedings of the 2020 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9090–9098, 2020.

[134] David Silver, Julian Schrittwieser, Kovalev Simonyan, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Andrew Bolton, and Neil Brown. Mastering the game of go without human knowledge. In *Nature*, volume 550, pages 354–359. Nature Publishing Group, 2017. doi: 10.1038/nature24270.

[135] Hong Liu, Mingsheng Long, Jianmin Wang, and Yu Wang. Learning to adapt to evolving domains. *Advances in neural information processing systems*, 33: 22338–22348, 2020.

[136] Zhenyi Wang, Li Shen, Tiehang Duan, Donglin Zhan, Le Fang, and Mingchen Gao. Learning to learn and remember super long multi-domain task sequence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7982–7992, June 2022.

[137] Abu Md Niamul Taufique, Chowdhury Sadman Jahan, and Andreas Savakis. Unsupervised continual learning for gradually varying domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 3740–3750, June 2022.

[138] Abolfazl Farahani, Sahar Voghoei, Khaled Rasheed, and Hamid R Arabnia. A brief review of domain adaptation. *Advances in data science and information engineering: proceedings from ICDATA 2020 and IKE 2020*, pages 877–894, 2021.

[139] Garrett Wilson and Diane J Cook. A survey of unsupervised deep domain adaptation. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(5):1–46, 2020.

[140] Shiliang Sun, Honglei Shi, and Yuanbin Wu. A survey of multi-source domain adaptation. *Information Fusion*, 24:84–92, 2015.

[141] Y. Zhou, J. Wu, S. Wang, and et al. Adversarial feature alignment for unsupervised domain adaptation. In *Proceedings of the 2021 IEEE International Conference on Image Processing (ICIP)*, pages 2656–2660, 2021.

[142] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Unsupervised domain adaptation via domain discrepancy. In *Proceedings of the 2018 European Conference on Computer Vision (ECCV)*, pages 593–609, 2018.

[143] R. Socher, A. Karpathy, and L. Fei-Fei. Zero-shot learning with visual semantic embeddings. In *Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1520–1528, 2018.

[144] Z. Huang, Y. Sun, F. Liu, and et al. Towards robust and efficient domain adaptation for image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 556–565, 2021.

[145] H. Wang, Y. Zheng, X. Hu, and et al. Domain adaptation via source-conditional distribution matching. In *Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1000–1008, 2019.

[146] Wei Zhang, Fei Wang, and Ling Hu. Adaptive transfer learning for ssvep-based bci spellers. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 1–4. IEEE, 2020.

[147] Jin Li, Zhiqiang Zhang, and Li Zhao. Improving ssvep speller performance with domain adaptation techniques. *IEEE Transactions on Biomedical Engineering*, 69(5):1257–1266, 2022.

[148] Yi Chen, Shanshan Wang, and Cheng Liu. Domain adaptation for eeg-based brain-computer interfaces. *Journal of Neural Engineering*, 18(2):026005, 2021.

[149] Jian Zhang, Ying Wang, and Zheng Zhang. Cross-domain anomaly detection in video surveillance. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1537–1541. IEEE, 2020.

[150] Dong Lu, Qi Wu, and Xiaoyang Zhang. Unsupervised domain adaptation for anomaly detection in video surveillance. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3372–3381. IEEE, 2022.

[151] Yu Wang, Wei Zeng, and Weilin Liang. Anomaly detection in videos with domain adaptation. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(11):4134–4145, 2020.

[152] Paras Sheth, Raha Moraffah, K Selçuk Candan, Adrienne Raglin, and Huan Liu. Domain generalization–a causal perspective. *arXiv preprint arXiv:2209.15177*, 2022.

[153] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and S Yu Philip. Generalizing to unseen domains: A survey on domain generalization. *IEEE transactions on knowledge and data engineering*, 35(8):8052–8072, 2022.

[154] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4396–4415, 2022.

[155] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. *Advances in neural information processing systems*, 24, 2011.

[156] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011.

[157] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei A Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I 12*, pages 158–171. Springer, 2012.

[158] Raghuraman Gopalan, Hongdong Li, and Rama Chellappa. Domain generalization via invariant feature subspaces. In *2011 IEEE International Conference on Computer Vision*, pages 1631–1638. IEEE, 2011.

[159] Ruichi Zhang, Jian Chen, Yifan Xu, and Dandan Yu. Domain generalization by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 162–179. Springer, 2016.

[160] Yi Li, Yan Wu, and Y. Zhang. Domain generalization with adversarial feature learning. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 78–82. IEEE, 2019.

[161] H. Zhang, Y. Li, and X. Wang. Domain generalization via learning data augmentation strategies. In *2021 IEEE International Conference on Computer Vision (ICCV)*, pages 9058–9067. IEEE, 2021.

[162] Yi Deng, Jian Zhou, and Z. Wang. Domain generalization with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2020–2028, 2019.

[163] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Kunle Kavukcuoglu, and Daan Wierstra. Matching networks for one-shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

[164] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

[165] L. A Lin. World with a billion cameras watching you is just around the corner. *Wall Street Journal*, page 12, 2019. URL https://www.wsj.com/articles/a-billion-surveillance-cameras-forecast-to-be-watching-within-two-years-11575565402.

[166] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv*, 41, 2009.

[167] G. Pang, C. Shen, L. Cao, and A. Hengel. Deep learning for anomaly detection. *ACM Computing Surveys*, 54:1–38, 2022. URL https://doi.org/10.11452F3439950.

[168] Tudor Ionescu, Smeureanu R., and Alexe S. B. & popescu, m. unmasking the abnormal events in video. *Proceedings Of The IEEE International Conference On Computer Vision*, pages 2895–2903, 2017.

[169] Y. Liu, C. Li, and B. Póczos. Classifier two sample test for video anomaly detections. *BMVC*, 71, 2018.

[170] J. Andrews, T. Tanay, E. Morton, and L. Griffin. *Transfer representation-learning for anomaly detection*. JMLR, 2016.

[171] K. Doshi and Y. Yilmaz. Continual learning for anomaly detection in surveillance videos. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition Workshops*, pages 254–255, 2020.

[172] K. Doshi and Y. Yilmaz. Online anomaly detection in surveillance videos with asymptotic bound on false alarm rate. *Pattern Recognition*, 114, 2021.

[173] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[174] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 770–778, 2016.

[175] I. Jolliffe and J. Principal component analysis Cadima. a review and recent developments. *Philosophical Transactions Of The Royal Society A: Mathematical, Physical And Engineering Sciences*, 374:20150202, 2016.

[176] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. *Proceedings Of The Seventh ACM SIGKDD International Conference On Knowledge Discovery And Data Mining*, pages 245–250, 2001.

[177] M. Ravanbakhsh, M. Nabi, E. Sangineto, L. Marcenaro, C. Regazzoni, and N. Sebe. Abnormal event detection in videos using generative adversarial nets. *2017 IEEE International Conference On Image Processing (ICIP)*, pages 1577–1581, 2017.

[178] Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. *arXiv preprint arXiv:1510.01553*, 2015.

[179] M. Hasan, J. Choi, J. Neumann, A. Roy-Chowdhury, and L. Davis. Learning temporal regularity in video sequences. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 733–742, 2016.

[180] M. Sabokrou, M. Fayyaz, M. Fathy, Z. Moayed, and R. Deep-anomaly Klette. Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision And Image Understanding*, 172:88–97, 2018.

[181] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning–ICANN 2011: 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I 21*, pages 44–51. Springer, 2011.

[182] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

[183] E. Forgy. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769, 1965.

[184] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. *Proceedings Of The 2000 ACM SIGMOD International Conference On Management Of Data*, pages 427–438, 2000.

[185] G. Pang, L. Cao, L. Chen, and H. Liu. Learning representations of ultrahigh-dimensional data for random distance-based outlier detection. *Proceedings Of The 24th ACM SIGKDD International Conference On Knowledge Discovery & Data Mining*, pages 2041–2050, 2018.

[186] H. Wang, G. Pang, C. Shen, and C. Ma. Unsupervised representation learning by predicting random distances. *CoRR*, abs/1912.12186, 2019. URL http://arxiv.org/abs/1912.12186.

[187] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*, 2018.

[188] M. Nguyen and N. Vien. Scalable and interpretable one-class svms with deep learning and random fourier features. *Joint European Conference On Machine Learning And Knowledge Discovery In Databases*, pages 157–172, 2019.

[189] P. Wu, J. Liu, and F. A Shen. deep one-class neural network for anomalous event detection in complex scenes. *IEEE Transactions On Neural Networks And Learning Systems*, 31:2609–2622, 2019.

[190] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. Siddiqui, A. Binder, E. M"uller, and M. Deep one-class classification Kloft. International conference on machine learning. *p*, pages 4393–4402, 2018.

[191] M. Caron, P. Bojanowski, A. Joulin, and M. Douze. Deep clustering for unsupervised learning of visual features. *Proceedings Of The European Conference On Computer Vision (ECCV)*, pages 132–149, 2018.

[192] Nat Dilokthanakul, Pedro AM Mediano, Marta Garnelo, Matthew CH Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *arXiv preprint arXiv:1611.02648*, 2016.

[193] Ghasedi Dizaji, Herandi K., Deng A., and Cai C. W. & huang, h. deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. *Proceedings Of The IEEE International Conference On Computer Vision*, pages 5736–5745, 2017.

[194] J. Xie, R. Girshick, and A. Farhadi. Unsupervised deep embedding for clustering analysis. *International Conference On Machine Learning*, pages 478–487, 2016.

[195] J. Yang, D. Parikh, and D. Batra. Joint unsupervised learning of deep representations and image clusters. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 5147–5156, 2016.

[196] G. Pang, C. Yan, C. Shen, A. Hengel, and X. Bai. Self-trained deep ordinal regression for end-to-end video anomaly detection. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 12173–12182, 2020.

[197] M. Oh and G. Iyengar. Sequential anomaly detection using inverse reinforcement learning. *Proceedings Of The 25th ACM SIGKDD International Conference On Knowledge Discovery & Data Mining*, pages 1480–1490, 2019.

[198] Ting Chen, Lu-An Tang, Yizhou Sun, Zhengzhang Chen, and Kai Zhang. Entity embedding-based anomaly detection for heterogeneous categorical events. *arXiv preprint arXiv:1608.07502*, 2016.

[199] M. Sabokrou, M. Khalooei, M. Fathy, and E. Adeli. Adversarially learned one-class classifier for novelty detection. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 3379–3388, 2018.

[200] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *Proceedings Of The IEEE Conference On Computer Vision And Pattern Recognition*, pages 779–788, 2016.

[201] Vladimir Vapnik. *Estimation of dependences based on empirical data*. Springer Science & Business Media, 2006.

[202] V. Saligrama and Z. Chen. Video anomaly detection based on local statistical aggregates. *2012 IEEE Conference On Computer Vision And Pattern Recognition*, pages 2112–2119, 2012.

[203] S. Wang, E. Zhu, J. Yin, and F. Porikli. Video anomaly detection and localization by local motion based joint video representation and ocelm. *Neurocomputing*, 277:161–175, 2018.

[204] Basarbatu Can and Huseyin Ozkan. A neural network approach for online nonlinear neyman-pearson classification. *IEEE Access*, 8:210234–210250, 2020.

[205] H Vincent Poor. *An introduction to signal detection and estimation*. Springer Science & Business Media, 2013.

[206] David Casasent and Xue-wen Chen. Radial basis function neural networks for nonlinear fisher discrimination and neyman–pearson classification. *Neural networks*, 16(5-6):529–535, 2003.

[207] Mark A Davenport, Richard G Baraniuk, and Clayton D Scott. Tuning support vector machines for minimax and neyman-pearson classification. *IEEE transactions on pattern analysis and machine intelligence*, 32(10):1888–1898, 2010.

[208] Xu-Ying Liu and Zhi-Hua Zhou. Learning with cost intervals. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 403–412, 2010.

[209] Gilles Gasso, Aristidis Pappaioannou, Marina Spivak, and Léon Bottou. Batch and online learning algorithms for nonconvex neyman-pearson classification. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–19, 2011.

[210] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. *Advances in neural information processing systems*, 20, 2007.

[211] Xin Tong, Yang Feng, and Jingyi Jessica Li. Neyman-pearson classification algorithms and np receiver operating characteristics. *Science advances*, 4(2): eaao1659, 2018.

[212] Clayton D Scott and Robert D Nowak. Learning minimum volume sets. *Journal of Machine Learning Research*, 7(Apr):665–704, 2006.

[213] Manqi Zhao and Venkatesh Saligrama. Anomaly detection with score functions based on nearest neighbor graphs. *Advances in neural information processing systems*, 22, 2009.

[214] Xin Tong. A plug-in approach to neyman-pearson classification. *The Journal of Machine Learning Research*, 14(1):3011–3040, 2013.

[215] M. E. J. Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46(5):323–351, 2005.

[216] E. Keogh et al. Locally adaptive real-time anomaly detection with incremental probability distribution estimation. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 212–222, 2002.

[217] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 1st edition, 2006.

[218] Ian Jolliffe. Principal component analysis. *Encyclopedia of statistics in behavioral science*, 2005.

[219] P. Domingos. A unified bias-variance decomposition for zero-one and squared loss. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 564–569, 2000.

[220] P. Bergmann, M. Fauser, D. Sattlegger, and C. Uninformed students Steger. Student-teacher anomaly detection with discriminative latent embeddings. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 4183–4192, 2020.

[221] M. Salehi, N. Sadjadi, S. Baselizadeh, M. Rohban, and H. Rabiee. Multiresolution knowledge distillation for anomaly detection. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 14902–14912, 2021.

[222] M. Georgescu, A. Barbalau, R. Ionescu, F. Khan, M. Popescu, and M. Shah. Anomaly detection in video via self-supervised and multi-task learning. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 12742–12752, 2021.

[223] Ioannis Kontoyiannis, Lambros Mertzanis, Athina Panotopoulou, Ioannis Papageorgiou, and Maria Skoularidou. Bayesian context trees: Modelling and exact inference for discrete time series. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(4):1287–1323, 2022.

[224] E. Sabeti, S. Oh, P. Song, and A. A Hero. Pattern dictionary method for anomaly detection. *Entropy*, 24:1095, 2022.

[225] K. Yue, L. Yang, R. Li, W. Hu, F. Zhang, and W. TreeUNet Li. Adaptive tree convolutional neural networks for subdecimeter aerial image segmentation. *ISPRS Journal Of Photogrammetry And Remote Sensing*, 156:1–13, 2019.

[226] R. Shang, J. Zhang, L. Jiao, Y. Li, N. Marturi, and R. Stolkin. Multi-scale adaptive feature fusion network for semantic segmentation in remote sensing images. *Remote Sensing*, 12:872, 2020.

[227] G. Pang, C. Shen, L. Cao, and A. Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54:1–38, 2021.

[228] S. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie. High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, 58:121–134, 2016.

[229] W. Yu, W. Cheng, C. Aggarwal, K. Zhang, H. Chen, and W. Netwalk: A Wang. flexible deep embedding approach for anomaly detection in dynamic networks. *Proceedings Of The 24th ACM SIGKDD International Conference On Knowledge Discovery & Data Mining*, pages 2672–2681, 2018.

[230] R. Ionescu, F. Khan, M. Georgescu, and L. Shao. Object-centric auto-encoders and dummy anomalies for abnormal event detection in video. *Proceedings Of The IEEE/CVF Conference On Computer Vision And Pattern Recognition*, pages 7842–7851, 2019.

[231] Y. Chong and Y. Tay. Abnormal event detection in videos using spatiotemporal autoencoder. *International Symposium On Neural Networks*, pages 189–196, 2017.

[232] R. Hinami, T. Mei, and S. Satoh. Joint detection and recounting of abnormal events by learning deep generic knowledge. *Proceedings Of The IEEE International Conference On Computer Vision*, pages 3619–3627, 2017.

[233] W. Luo, W. Liu, and S. Gao. Remembering history with convolutional lstm for anomaly detection. *2017 IEEE International Conference On Multimedia And Expo (ICME)*, pages 439–444, 2017.

[234] R. Groeneveld and G. Meeden. Measuring skewness and kurtosis. *Journal Of The Royal Statistical Society: Series D (The Statistician)*, 33:391–399, 1984.

[235] Michele Basseville, Igor V Nikiforov, et al. *Detection of abrupt changes: theory and application*, volume 104. prentice Hall Englewood Cliffs, 1993.

[236] Keval Doshi and Yasin Yilmaz. Zero-shot action recognition with transformer-based video semantic embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4858–4867, 2023.

[237] Keval Doshi and Yasin Yilmaz. Multi-task learning for video surveillance with limited data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3889–3899, 2022.

[238] Keval Doshi and Yasin Yilmaz. A modular and unified framework for detecting and localizing video anomalies. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3982–3991, 2022.

[239] Keval Doshi and Yasin Yilmaz. Towards interpretable video anomaly detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2655–2664, January 2023.

[240] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. *International Conference On Artificial Neural Networks*, pages 52–59, 2011.

[241] W. Lu, Y. Cheng, C. Xiao, S. Chang, S. Huang, B. Liang, and T. Huang. Unsupervised sequential outlier detection with deep architectures. *IEEE Transactions On Image Processing*, 26:4321–4330, 2017.

[242] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[243] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[244] W. Luo, W. Liu, and S. A Gao. revisit of sparse coding based anomaly detection in stacked rnn framework. *Proceedings Of The IEEE International Conference On Computer Vision*, pages 341–349, 2017.

[245] Maryam Qasim and Elena Verdu. Video anomaly detection system using deep convolutional and recurrent models. *Results in Engineering*, 18:101026, 2023.

[246] T. Schlegl, P. Seeb"ock, S. Waldstein, U. Schmidt-Erfurth, and G. Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. *International Conference On Information Processing In Medical Imaging*, pages 146–157, 2017.

[247] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient gan-based anomaly detection. *arXiv preprint arXiv:1802.06222*, 2018.

[248] T. Schlegl, P. Seeb"ock, S. Waldstein, G. Langs, and U. f-AnoGAN Schmidt-Erfurth. Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, 54:30–44, 2019.

[249] Weijia Liu, Jiuxin Cao, Yilin Zhu, Bo Liu, and Xuelin Zhu. Real-time anomaly detection on surveillance video with two-stream spatio-temporal generative model. *Multimedia systems*, 29(1):59–71, 2023.

[250] Francesco Carrera, Vincenzo Dentamaro, Stefano Galantucci, Andrea Iannacone, Donato Impedovo, and Giuseppe Pirlo. Combining unsupervised approaches for near real-time network traffic anomaly detection. *Applied Sciences*, 12(3):1759, 2022.

[251] Viet-Tuan Le and Yong-Guk Kim. Attention-based residual autoencoder for video anomaly detection. *Applied Intelligence*, 53(3):3240–3254, 2023.

[252] Yunpeng Chang, Zhigang Tu, Wei Xie, Bin Luo, Shifu Zhang, Haigang Sui, and Junsong Yuan. Video anomaly detection with spatio-temporal dissociation. *Pattern Recognition*, 122:108213, 2022.

[253] Zhongyue Wang and Ying Chen. Anomaly detection with dual-stream memory network. *Journal of Visual Communication and Image Representation*, 90: 103739, 2023.

[254] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pages 01–06. IEEE, 2021.

[255] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[256] Yunseung Lee and Pilsung Kang. Anovit: Unsupervised anomaly detection and localization with vision transformer-based encoder-decoder. *IEEE Access*, 10:46717–46724, 2022.

[257] Zhiwei Yang, Jing Liu, Zhaoyang Wu, Peng Wu, and Xiaotao Liu. Video event restoration based on keyframes for video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14592–14601, 2023.

[258] Peng Wu, Xuerong Zhou, Guansong Pang, Yujia Sun, Jing Liu, Peng Wang, and Yanning Zhang. Open-vocabulary video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18297–18307, June 2024.

[259] Jiawen Zhu and Guansong Pang. Toward generalist anomaly detection via in-context residual learning with few-shot sample prompts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17826–17836, 2024.

[260] Somboon Hongeng, Ram Nevatia, and Francois Bremond. Bayesian framework for video surveillance: Application to motion-based recognition. In *Proceedings IEEE International Conference on Computer Vision. ICCV 2000*, volume 2, pages 169–176. IEEE, 2000.

[261] Claudio Piciarelli, Christian Micheloni, and Gian Luca Foresti. Trajectory-based anomalous event detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(11):1544–1554, 2008.

[262] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, pages 5574–5584, 2017.

[263] Z. Liu, Z. Xie, Z. Zhang, and Y. Li. Bayesian nonparametric submodular video partition for robust anomaly detection. *arXiv*, 2022. Available: https://arxiv.org/abs/2203.12840.

[264] X. Li, Y. Liu, Z. Zhang, and W. Chen. Bayesian feed forward neural network-based efficient anomaly detection from surveillance videos, 2022. Available: https://www.researchgate.net/publication/359977095_Bayesian_Feed_Forward_Neural_Network-Based_Efficient_Anomaly_Detection_from_Surveillance_Videos.

[265] J. Pers, V. Sulic, M. Kristan, M. Perse, K. Polanec, and S. Kovacic. Histograms of optical flow for efficient representation of body motion. *Pattern Recognition Letters*, 31:1369–1376, 2010. URL https://www.sciencedirect.com/science/article/pii/S0167865510001121.

[266] V. Vapnik and A. Chervonenkis. The uniform convergence of frequencies of the appearance of events to their probabilities. *Doklady Akademii Nauk*, pages 781–783, 1968.

[267] BP Welford. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420, 1962.

[268] P. Mahalanobis. On the generalized distance in statistics. *Proceedings Of The National Institute Of Sciences (Calcutta)*, 2:49–55, 1936.

[269] C. Brown and H. Davis. Receiver operating characteristics curves and related decision measures: A tutorial. *Chemometrics And Intelligent Laboratory Systems*, 80:24–38, 2006.

[270] Keum-Shik Hong and Muhammad Jawad Khan. Hybrid brain–computer interface techniques for improved classification accuracy and increased number of commands: a review. *Frontiers in neurorobotics*, page 35, 2017.

[271] Jonathan R Wolpaw, Herbert Ramoser, Dennis J McFarland, and Gert Pfurtscheller. Eeg-based communication: improved accuracy by response verification. *IEEE transactions on Rehabilitation Engineering*, 6(3):326–333, 1998.

[272] Alexander Ya Kaplan, Andrew A Fingelkurts, Alexander A Fingelkurts, Sergei V Borisov, and Boris S Darkhovsky. Nonstationary nature of the brain activity as revealed by eeg/meg: methodological, practical and conceptual challenges. *Signal processing*, 85(11):2190–2212, 2005.

[273] Erwei Yin, Zongtan Zhou, Jun Jiang, Yang Yu, and Dewen Hu. A dynamically optimized ssvep brain–computer interface (bci) speller. *IEEE transactions on biomedical engineering*, 62(6):1447–1456, 2014.

[274] Zhonglin Lin, Changshui Zhang, Wei Wu, and Xiaorong Gao. Frequency recognition based on canonical correlation analysis for ssvep-based bcis. *IEEE transactions on biomedical engineering*, 53(12):2610–2614, 2006.

[275] Xiaogang Chen, Yijun Wang, Shangkai Gao, Tzyy-Ping Jung, and Xiaorong Gao. Filter bank canonical correlation analysis for implementing a high-speed ssvep-based brain–computer interface. *Journal of neural engineering*, 12(4): 046008, 2015.

[276] Ka Fai Lao, Chi Man Wong, Ze Wang, and Feng Wan. Learning prototype spatial filters for subject-independent ssvep-based brain-computer interface. In *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 485–490. IEEE, 2018.

[277] Chi Man Wong, Feng Wan, Boyu Wang, Ze Wang, Wenya Nan, Ka Fai Lao, Peng Un Mak, Mang I Vai, and Agostinho Rosa. Learning across multi-stimulus enhances target recognition methods in ssvep-based bcis. *Journal of neural engineering*, 17(1):016026, 2020.

[278] Yang Deng, Qingyu Sun, Ce Wang, Yijun Wang, and S Kevin Zhou. Trca-net: using trca filters to boost the ssvep classification with convolutional neural network. *Journal of Neural Engineering*, 20(4):046005, 2023.

[279] Masaki Nakanishi, Yijun Wang, Xiaogang Chen, Yu-Te Wang, Xiaorong Gao, and Tzyy-Ping Jung. Enhancing detection of ssveps for a high-speed brain speller using task-related component analysis. *IEEE Transactions on Biomedical Engineering*, 65(1):104–112, 2017.

[280] Jianli Yang, Songlei Zhao, Zhiyu Fu, and Xiuling Liu. Pmf-cnn: Parallel multi-band fusion convolutional neural network for ssvep-eeg decoding. *Biomedical Physics & Engineering Express*, 2024.

[281] Zhijiang Wan, Manyu Li, Shichang Liu, Jiajin Huang, Hai Tan, and Wenfeng Duan. Eegformer: A transformer–based brain activity classification method using eeg signal. *Frontiers in Neuroscience*, 17:1148855, 2023.

[282] Xiaogang Chen, Yijun Wang, Masaki Nakanishi, Xiaorong Gao, Tzyy-Ping Jung, and Shangkai Gao. High-speed spelling with a noninvasive brain–computer interface. *Proceedings of the national academy of sciences*, 112(44): E6058–E6067, 2015.

[283] Masaki Nakanishi, Yijun Wang, Xiaogang Chen, Yu-Te Wang, Xiaorong Gao, and Tzyy-Ping Jung. Enhancing detection of ssveps for a high-speed brain speller using task-related component analysis. *IEEE Transactions on Biomedical Engineering*, 65(1):104–112, 2018. doi: 10.1109/TBME.2017.2694818.

[284] David R Hardoon, Sandor Szedmak, and John Shawe-Taylor. Canonical correlation analysis: An overview with application to learning methods. *Neural computation*, 16(12):2639–2664, 2004.

[285] Masaki Nakanishi, Yijun Wang, Yu-Te Wang, Yasue Mitsukura, and Tzyy-Ping Jung. A high-speed brain speller using steady-state visual evoked potentials. *International journal of neural systems*, 24(06):1450019, 2014.

[286] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International conference on machine learning*, pages 6028–6039. PMLR, 2020.

[287] Li Yi, Gezheng Xu, Pengcheng Xu, Jiaqi Li, Ruizhi Pu, Charles Ling, A Ian McLeod, and Boyu Wang. When source-free domain adaptation meets learning with noisy labels. *arXiv preprint arXiv:2301.13381*, 2023.

[288] Shiqi Yang, Joost van de Weijer, Luis Herranz, Shangling Jui, et al. Exploiting the intrinsic neighborhood structure for source-free domain adaptation. *Advances in neural information processing systems*, 34:29393–29405, 2021.

[289] Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap by reducing style bias. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8690–8699, 2021.

[290] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[291] Yves Grandvalet and Yoshua Bengio. Entropy regularization. 2006.

[292] Peter J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. ISSN 0377-0427. doi: https://doi.org/10.1016/0377-0427(87)90125-7. URL https://www.sciencedirect.com/science/article/pii/0377042787901257.