

Politecnico di Torino

Politecnico di Torino

Master's Degree in Computer Engineering

A.y. 2025/2026

Graduation Session September 2026

JEPA Representation Learning for Intrusion Detection on Real-World Drifting Network Traffic

Supervisors:

Prof. Guido MARCHETTO

Prof. Alessio SACCO

MSc Cristiano SERRA

Candidate:

Bekhzod ANVAROV

Abstract

Today, the use of the Anomaly Detection Classifier (ADC) is standard practice in Intrusion Detection Systems. The ADC is trained to map raw features of network flow records to binary labels using a dataset collected in a single time period. However, network traffic keeps evolving after the training data is collected. The same feature values come to carry different labels, so the feature-label relation the ADC was fitted on no longer holds, a shift known as concept drift. The raw features are the only input the ADC has. Under drift, the ADC misclassifies traffic from later periods and its accuracy falls. What is needed is a model that can correctly classify flows under concept drift without refitting on new traffic.

In this thesis, the ADC was fitted on learned representations rather than raw features. The representations were produced by the context encoder of a self-supervised JEPA (Joint Embedding Predictive Architecture) model, pre-trained on the flow records without labels. The encoder was frozen, and the same standard shallow classifiers were fitted in two arms, one on its representations and one, as a control, directly on the raw features, so that the comparison turns on the representation. Learned from the traffic itself, these representations are not fitted to the label relation that drifts. Three labeled samples were taken from MAWIFlow, real backbone traffic recorded in January 2011, 2016 and 2021, years far enough apart for drift to be observable. Both arms were evaluated in domain and across years, and in a cross-year pair nothing was refitted on the target year. Because the class balance varies between years, the comparison rests on the threshold-free metrics.

On the cross-year cells, the representation arm is expected to retain more of its performance than the raw-feature control. In domain, where drift is absent, the two arms are expected to sit close, so the gap between them across years is what carries the comparison. If instead the two arms track each other, the extra pre-training stage brings no benefit. If the expectation holds, a representation pre-trained once on unlabeled traffic would lengthen the interval between refits, though the classifier still needs labels. If it does not, the limit on cross-year performance lies somewhere other than the input representation, and effort belongs there instead.

Table of Contents

1 Introduction **1**
- 1.1 Goal . . . 2
- 1.2 Solution . . . 3
- 1.3 Results in Brief . . . 3
- 1.4 Structure of this thesis . . . 4

2 Background **5**
- 2.1 Anomaly Detection . . . 5
  - 2.1.1 Signature-Based and Anomaly-Based Detection . . . 5
  - 2.1.2 Supervised Classification of Flow Records . . . 6
  - 2.1.3 Concept Drift . . . 7
- 2.2 Data . . . 8
  - 2.2.1 The MAWI Archive . . . 8
  - 2.2.2 Flow Features with CICFlowMeter . . . 8
  - 2.2.3 MAWILab Labels . . . 9
  - 2.2.4 The Dataset . . . 10
- 2.3 Self-Supervised Representation Learning . . . 11
  - 2.3.1 Representation Collapse . . . 12
  - 2.3.2 Joint-Embedding Predictive Architectures . . . 13

3 Related Work **15**
- 3.1 Intrusion Detection under Concept Drift . . . 15
- 3.2 Self-Supervised Representation Learning . . . 16
- 3.3 Joint-Embedding Predictive Architectures . . . 17
- 3.4 The MAWIFlow Benchmark . . . 18
- 3.5 Positioning of This Work . . . 20

4 Methodology **21**
- 4.1 Experimental Design . . . 21
- 4.2 Preprocessing . . . 22
- 4.3 The JEPA Model . . . 23

4.4 Downstream Classification 25
4.5 Evaluation Protocol and Metrics 27

5 Results **29**
5.1 Metrics 29
5.2 Reading the Results 31
5.3 In Domain 31
5.4 Cross-Year Transfer 33
5.5 Degradation 35
5.6 The Representation and the Choice of Classifier 39
5.7 Alternative Regularizers 40

6 Conclusions and Future Work **42**
6.1 Conclusions 42
6.2 Limitations 43
6.3 Future Work 43

Bibliography **45**

List of Tables

2.1 The three MAWIFlow samples 11

3.1 Cross-temporal baselines published with MAWIFlow 18
3.2 In-domain figures used in the prevalence recovery 19

4.1 Scaling groups 23
4.2 Pre-training configuration 26

5.1 Results for every train-test pair and classifier 32
5.2 Single-input ROC-AUC per sample 37
5.3 Degradation between 2016 and 2021 38
5.4 Logistic regression against XGBoost, in domain 40

List of Figures

2.1 The joint-embedding predictive architecture 13

4.1 The JEPA model of this thesis 24

5.1 ROC curves in domain, 2021 33
5.2 ROC curves, 2011 to 2021 34
5.3 ROC curves, 2016 to 2021 35

Acronyms

ADC

Anomaly Detection Classifier

IDS

Intrusion Detection System

JEPA

Joint Embedding Predictive Architecture

Chapter 1

Introduction

Attacks are now delivered at a rate that outpaces human monitoring of network flows, which creates a need for a response faster than manual inspection can provide [1]. For this reason, the Anomaly Detection Classifier (ADC) was adopted in Intrusion Detection Systems (IDSs), and its use is today standard practice.

The ADC is a supervised model. It is fitted on traffic records whose class is already known, and what it learns from them is a relation: which combinations of the quantities measured about a record belong to benign traffic, and which belong to traffic that is not. Once fitted, it is applied to records it has never seen and assigns each of them a class of its own. The whole arrangement rests on that one relation, learned once from the records that were available when the model was built.

Traffic, however, does not stand still. The services carried over a network are replaced, the way applications communicate changes, and the behavior that constitutes an attack is revised in response to the defenses raised against it. The measurements themselves remain the same quantities, but what they signify does not: a combination of values that belonged to one class when the training records were collected may belong to the other class some years later. The relation the model was fitted on has moved out from under it. This is what is meant by concept drift.

Nothing in the arrangement carries that change to the model. The measurements are the only input it has, and they continue to arrive in the same form as before, so there is no signal by which the model could discover that their meaning has shifted. It goes on applying a relation that has ceased to hold, and its decisions on traffic from later periods grow less reliable the further that traffic lies from the records it was fitted on.

The established answer is to fit the model again, on records drawn from the traffic as it is now. This works, and it is what deployments do, but it carries a cost that is easy to overlook. Fitting requires records whose class is already known,

and that knowledge does not come with the traffic. It has to be produced, by a procedure that must itself be run and trusted, and the effort that production takes is what sets how often a model can be renewed. A detector is therefore only as current as the last labeling effort someone was willing to undertake.

What would be worth having is a model that continues to classify correctly as the traffic moves away from what it was fitted on, without being fitted again on that later traffic each time.

This thesis examines one route to it. Rather than fitting the classifier on the measurements themselves, it fits the classifier on a representation of them: a description of each record learned beforehand from the traffic alone, with no labels involved at any point. The reason this is worth trying follows from where the problem lies. What drifts is the relation between the measurements and the classes, and a description learned without ever consulting a class is not built on that relation. Whether a classifier given such a description holds up better than one given the measurements directly, when both are applied to traffic recorded years later, is the question the remainder of this thesis puts to the test.

1.1 Goal

The goal of this thesis is to determine whether representations learned from unlabeled traffic can support the correct classification of flow records under concept drift, with no refitting on later traffic.

To this end, the work sets out to:

- pre-train a self-supervised Joint Embedding Predictive Architecture (JEPA) model [2, 3] on the unlabeled flow records of the training year alone, no traffic from any later year being seen at any point, and take as representations the output of its context encoder;
- freeze the encoder and fit the same standard shallow classifiers in two arms, one on the representations and one, as a control, directly on the raw features, so that the comparison turns on the representation;
- draw three labeled samples from MAWIFlow [4], real backbone traffic recorded in January 2011, 2016 and 2021, years far enough apart for drift to be observable;
- evaluate both arms in domain and across years, refitting nothing on the target year in a cross-year pair;
- compare the two arms on the threshold-free metrics, because the class balance varies between years.

The gap between the two arms across years is what carries the comparison. In domain, where drift is absent, the two arms are expected to sit close.

1.2 Solution

The approach taken is to change the input to the classifier rather than the classifier itself. A JEPA, a self-supervised model made of a context encoder, a target encoder and a predictor, is pre-trained on the flow records of one year without any label: part of each record is masked, and the model learns to predict, in latent space, the representation of the complete record from what remains of it. Once trained, the context encoder is frozen and its output becomes the representation on which four standard classifiers, a logistic regression, a decision tree, a random forest and XGBoost, are fitted. The same four classifiers, with the same settings, the same split and the same seed, are also fitted directly on the preprocessed measurements, so that the two arms differ in their input alone. The data are three samples of three million labeled flow records each from the MAWIFlow benchmark, real backbone traffic recorded in January 2011, 2016 and 2021. Both arms are evaluated on every pair of fitted and evaluated year, in domain and across years, with nothing refitted on the later year, and degradation is measured as what a classifier gives up when it is carried from the year it was fitted on to another. Chapter 4 gives the details.

1.3 Results in Brief

In domain, where drift is absent, the representation leaves performance where it was: the two arms differ by under half a hundredth of F_1 on average, and the only classifier that moves appreciably is the logistic regression, which gains between 2.5 % and 5 %. Across years, the representation does not reduce degradation. On the one pair of years where the comparison is sound, 2016 and 2021, it loses less than the raw measurements in two of eight comparisons and more in six; the one large cross-year gain, on the widest interval, lies on the 2011 sample, which the thesis shows to be unlike the other two and does not build on. What the representation does provide is linear separability: fitted on it, a logistic regression removes between 58 % and 74 % of its shortfall to XGBoost on the raw measurements, on the area under the ROC curve. The measurements also show why transfer across a decade fails on this data at all: of the inputs that separate the two classes on their own in both years, none points the same way in 2011 and 2021, so what changed between the samples is what the labels mean, not what the traffic looks like. Chapter 5 reports the measurements and Chapter 6 draws the conclusions.

1.4 Structure of this thesis

This thesis is structured as follows:

- Chapter 2 — Background introduces the anomaly detection classifier and the assumption it rests on, the concept drift that breaks that assumption, the archive and the tools from which the traffic records and their labels are obtained, and the self-supervised representation learning the approach draws on.
- Chapter 3 — Related Work reviews what has been proposed for detection under drift and what each approach costs, the alternatives to the representation learning used here, the earlier instantiations of the architecture and how the present one departs from them, and the benchmark that supplies the data, closing on the gap this thesis addresses.
- Chapter 4 — Methodology sets out the experiment: the two arms and what is held fixed between them, the preprocessing, the form the model takes and the reasons for it, the classifiers fitted on top of the frozen encoder, and the protocol and metrics under which everything is evaluated.
- Chapter 5 — Results reports what was measured: the two arms in domain and across years, the degradation each of them suffers when the evaluated year is not the fitted year and what the measurements show to have reversed between the samples, the effect of the representation on the choice of classifier, and two alternative regularizers that were tried.
- Chapter 6 — Conclusions and Future Work answers the question the thesis put, states what bounds that answer, and sets out what would have to follow.

Chapter 2

Background

2.1 Anomaly Detection

The traffic crossing a modern network arrives in quantities that no operator can review by hand. Far more connections are established in a working day than could ever be examined individually, and each of them would have to be looked at before it could be declared harmless. Attacks compound the problem, because an intrusion that unfolds in seconds cannot be answered by a procedure that takes hours. Detection therefore has to be carried out automatically, and the question is not whether to automate it but what the automated component should be asked to decide [1].

2.1.1 Signature-Based and Anomaly-Based Detection

Automated detection takes two broad forms. Signature-based detection compares observed traffic against descriptions of attacks that are already known and reports a match. It is precise, since a match is evidence of one particular attack, but it is bounded by its catalog, and an attack for which no description has yet been written passes unremarked [1].

Anomaly-based detection instead forms a notion of what ordinary traffic looks like and reports departures from it, which allows it to react to activity that has never been cataloged. A further distinction is drawn within it between unsupervised methods, which infer normality from traffic that carries no labels, and supervised ones, which are fitted on records whose class is already known. This thesis is concerned throughout with the supervised case, and the term ADC refers to it: a classifier fitted on labeled records to separate benign from anomalous traffic. Every later reference to anomaly detection in this work means that classifier.

2.1.2 Supervised Classification of Flow Records

The object on which the ADC operates is the flow record rather than the individual packet. A flow groups the packets exchanged between two endpoints over the course of a single conversation and summarizes them as one row of numeric quantities, among them how long the conversation lasted, how many packets traveled in each direction, how large those packets were and how they were spaced in time. Working at this level keeps the number of decisions tractable, and it describes behavior rather than content, which matters when the payload is encrypted and unavailable for inspection. It also means that the classifier never observes traffic directly. It observes whatever the flow extractor chose to measure, so the features it is given are a property of the tool that produced them.

In the supervised formulation the classifier is a function fitted to data. Each flow record is a vector of features and carries a binary label, benign or anomalous. A set of such pairs is collected from one capture and a model is fitted so that it reproduces the labels of that set while remaining general enough to be useful beyond it. The fitted model is then applied to records it has not seen, and assigns a label to each of them. Nothing of the capture itself is retained. What the model carries forward is only the relation it inferred between the features and the labels.

The models used for this task on tabular data belong to a small number of families. Logistic regression fits a single linear boundary through the feature space and is fast to fit and simple to interpret, but it can only separate classes that a linear boundary can separate. A decision tree partitions the space by thresholding one feature at a time, which lets it describe conditions that are not linear, at the cost of fitting the training set closely. A random forest averages many such trees, each fitted on a resampled version of the data, and thereby reduces the variance of a single tree. Gradient boosting builds trees in sequence, each one fitted to the errors left by those before it. In published comparisons on tabular problems the boosted variants are consistently among the strongest of these, which is why a new method on such data is measured against them rather than against a linear model [5]. These four families are the classifiers used throughout this thesis, and they are fitted unchanged in both arms of the comparison reported in Chapter 5.

Whatever the family, the immediate output is a score, and a decision is obtained by comparing that score against a threshold. The quantities computed from those decisions, accuracy among them, therefore depend on where the threshold was placed. They depend, too, on the proportion in which the two classes are present, which is a property of the capture rather than of the classifier and can differ from one capture to another: accuracy in particular moves with that proportion as much as with the quality of the decision, so two captures with different proportions cannot be compared through it. Measures that summarize the score over all of its possible thresholds do not carry that dependence and remain comparable across

captures whose balance differs. The six quantities reported in this thesis, and the rule for which of them serves which comparison, are set out in Chapter 5, where they are first used.

2.1.3 Concept Drift

Everything described so far rests on an assumption that is seldom stated explicitly. The relation between features and labels that the classifier inferred from the training capture is taken to be the same relation that holds when the classifier is put to work. Training data and future traffic are treated as two draws from one stable phenomenon, and the quality measured on a held-out portion of the capture is read as an estimate of the quality that will be obtained later.

Networks do not honor that assumption. Protocols are replaced, applications change how they communicate, defensive measures alter what ordinary traffic looks like, and the behavior that constitutes an attack is revised in response. The consequence is that a feature vector which belonged to one class in the year the capture was taken may belong to the other class some years afterward. When the distribution of the label conditioned on the features changes in this way, the phenomenon is called concept drift [6]. It is distinct from covariate shift, in which the traffic itself moves but the relation between features and labels holds. The distinction matters, because the two call for different answers: a change confined to the inputs can be addressed by reweighting or by adjusting the features, whereas a change in the relation itself invalidates what the model learned, and no reweighting of the inputs recovers it. Drift is further described by the manner in which it arrives: suddenly, when the relation changes at one point in time; gradually, as one regime gives way to another; or recurrently, when a relation seen earlier returns. The kind that separates captures recorded years apart is the gradual sort, accumulated over a long interval rather than produced by a single event.

The established responses to drift are of three kinds. The classifier can be refitted at intervals on a recent capture; drift can be monitored, so that refitting is triggered when the relation is seen to move [7]; or the model can be made adaptive, updating continuously as new records arrive. What these share is a dependence on labels, since each of them requires records from the new period whose class is already known. Labels for network traffic are not freely available. They are the output of a labeling procedure, they cost something to produce, and their meaning is a property of that procedure rather than of the traffic. What that procedure is, for the data used in this thesis, is the subject of the next section.

2.2 Data

A study of concept drift makes demands of its data that a single capture cannot meet. The traffic has to be observed at more than one point in time; the same measurement procedure has to be applied on each occasion, so that the feature space does not change along with the traffic; and every record has to carry a label, since both the fitting and the evaluation of an ADC depend on one. Traffic produced in a laboratory does not serve either, because the behavior it contains is the behavior that was scripted into it and does not evolve on its own between one capture and the next. What is needed is real traffic, recorded repeatedly over a long interval, and labeled by a procedure applied in the same way each time [4].

2.2.1 The MAWI Archive

The traffic used in this thesis comes from the archive maintained by the MAWI working group of the WIDE project [8]. The archive publishes packet traces captured at a measurement point on the WIDE backbone, on its transit link toward an upstream provider, and traces from that point have been collected daily since 2006. Each daily trace is a fifteen-minute packet capture of a trans-Pacific backbone link between Japan and the United States [4]. Addresses in the published traces are anonymized, and the archive is distributed for research use.

Two properties make it suitable here. The record is long enough that captures can be chosen far enough apart for drift to have accumulated between them, and the traffic is operational rather than generated, so whatever changed between two captures changed because the network changed. The measurement point has not been entirely static over that period, however: the monitored link was upgraded in 2007, a rate limit applied to it was removed in 2016, and a second upstream connection was added in 2023, all of them recorded by the archive itself [8]. A comparison across years inherits those changes along with the ones it means to study.

2.2.2 Flow Features with CICFlowMeter

A packet trace is not in a form an ADC can consume. The conversion is performed by a flow exporter, and the one used for this data is CICFlowMeter [9, 10]. It groups packets into bidirectional flows keyed on source and destination address, source and destination port and protocol, and emits one fixed-length vector of statistics per flow: the duration of the conversation, the number and total size of the packets sent in each direction, the minimum, maximum, mean and standard deviation of the packet lengths and of the intervals between packets, counts of the TCP flags observed, and rates derived from these.

In the files used here each flow occupies one row of ninety columns. Seven of them identify or time-stamp the flow rather than measure it, namely the flow identifier, the source and destination addresses, the source and destination ports, the protocol and the timestamp. Seven more carry the label and the provenance of the labeling. The remaining seventy-six columns are the flow measurements themselves. Which of those measurements are presented to a model, and in what form, is a preprocessing decision rather than a property of the data, and it is described in Chapter 4.

This is what was meant in Section 2.1.2 by the classifier never observing traffic directly. Everything it can learn about a conversation has to be expressed in these columns, and a change in the traffic that the exporter does not measure is a change the classifier cannot see.

2.2.3 MAWILab Labels

Labels are not observed in the traffic; they are assigned to it. For the MAWI archive they are produced by MAWILab [11], which annotates the daily traces automatically. Four unsupervised detectors are run over each trace, based respectively on the Hough transform, a gamma model, the Kullback-Leibler divergence and principal component analysis, each in three configurations of differing sensitivity. Their alarms are then combined, using singular value decomposition together with graph theory, into a single annotation per flow.

The procedure assigns one of four labels. A flow is *anomalous* if the combination judges it abnormal, *suspicious* if it is probably anomalous but not clearly identified as such by the method, and *notice* if MAWILab did not identify it as anomalous although at least one detector reported it. Everything that no detector reported is *benign*.

Two consequences follow, and both matter for how the results of this thesis should be read. The labels are the output of a detection procedure rather than ground truth, so a classifier fitted on them learns to reproduce the judgment of that procedure, and it inherits whatever that procedure gets wrong. And the benign class is defined negatively, as the traffic no detector reported, so it is not a positive statement that the traffic was harmless. This is the labeling procedure that Section 2.1.3 referred to when it observed that labels are the output of a procedure and that their meaning is a property of it. The four labels have to be reduced to the two an ADC decides between; that reduction, and the dataset on which it was performed, are the subject of the next section.

2.2.4 The Dataset

The three samples examined in this thesis are not selected here. They are the subsets that define the MAWIFlow benchmark, which draws approximately three million flows from January 2011, January 2016 and January 2021 and publishes them as a fixed collection [4]. Each of the three is referred to throughout this thesis as a sample, or simply by its year. What was decided in this work is to adopt that benchmark rather than to assemble captures independently. The reason is that the object of study is the representation given to the classifier, and holding the data fixed at a published definition keeps it from becoming a variable of the experiment. The spacing that comes with the benchmark suits the question as well: the samples are five years apart and span a decade, which is the kind of interval over which the gradual drift described in Section 2.1.3 accumulates.

Each of the three files contains three million flow records described by ninety columns, of which seventy-six are flow measurements, as set out in Section 2.2.2. The column names and their types are identical in all three files, so a model fitted on one year can be applied to another without any adjustment to the input.

The label supplied with each record is one of the four MAWILab classes, and a binary decision requires two. The benchmark defines that reduction: benign is mapped to the negative class and every other label, anomalous, suspicious and notice alike, is mapped to the positive one [4]. The mapping is adopted here unchanged, both because it is the benchmark's own and because the alternative treatments are worse. Discarding the uncertain classes would remove the traffic about which the detectors disagreed, which is the traffic a detector most needs to handle, and assigning notice to the benign class would assert that traffic reported by at least one detector is harmless, which MAWILab does not claim.

Table 2.1 gives the composition of the three samples. The proportions are not stable across them. The share of records that are not benign is 20.57 % in the 2016 sample, 50.61 % in the 2021 sample and 71.23 % in the 2011 sample, so the positive class is a minority in one year, half the traffic in another and the clear majority in the third. This is the situation Section 2.1.2 anticipated: an accuracy obtained on one of these samples cannot be set beside an accuracy obtained on another, because the two figures are computed against different balances. The four MAWILab classes are also not all present, since no record in the 2016 or 2021 sample carries the notice label.

The counts in the table are those of the published subsets. The cleaning applied before a model sees them, described in Chapter 4, removes only the records whose measurements are missing or not finite, and it removes very few: thirty-nine records in the 2011 sample, none in the 2016 sample and slightly over six thousand in the 2021 sample, at most two records in every thousand. The positive rate is unchanged in the first two samples and moves by five hundredths of a percentage point in the

Table 2.1: Composition of the three MAWIFlow samples. Counts are of flow records as they appear in the published subsets. The positive rate is the share of records that are not benign, which is the share the binarized label assigns to the anomalous class.

	MAWILab label				
Sample	Benign	Anomalous	Suspicious	Notice	Positive rate
2011-01	863,018	1,616,188	439,459	81,335	71.23 %
2016-01	2,382,820	496,104	121,076	–	20.57 %
2021-01	1,481,712	1,217,457	300,831	–	50.61 %
Total	4,727,550	3,329,749	861,366	81,335	

third, so the balances in the table are also the balances the models are fitted and evaluated against.

Because the samples, the labels and the binarization are all taken from the benchmark, the data used here is the data the MAWIFlow paper used. The results reported in Chapter 5 are nonetheless not a like-for-like extension of the baselines published with it, since the two differ in how the records are prepared: a different subset of the flow measurements is retained and a different scaling is applied to them, and the evaluation protocol is not the paper's. Those choices are described in Chapter 4, and the comparison is treated in Chapter 3. What the shared data does establish is the premise this chapter has been building toward. The seventy-six measurements are the same columns, produced by the same exporter, in every one of the three years. Whatever separates the traffic of 2011 from the traffic of 2021 in the eyes of a classifier is therefore not a change in the feature space, but a change in the labels attached to it. Section 5.5 measures that change column by column.

2.3 Self-Supervised Representation Learning

The preceding section left the classifier with a fixed set of columns and a label relation that moves beneath them. A representation offers a different input to fit on, and the property that makes it worth considering is that it can be learned without labels at all. A model trained in that way has no notion of benign and no notion of anomalous; it is presented with flow records and treats all of them alike. It is therefore not committed in advance to either of the two positions a labeled method has to take, modeling only the ordinary traffic and calling everything else a departure from it, or modeling only the relation between features and the classes as they were labeled at one moment. It represents the traffic it is given, whichever

class the records belong to.

A representation is a learned mapping from the raw feature vector into another space. What it holds is not the record itself, nor a pattern copied out of it, but the position of that record with respect to all the others the model was shown. Two flows that behave alike are placed near one another whether or not anything in the raw columns says so directly, and a flow is described by that placement rather than by its coordinates.

Such a mapping can be learned without labels, because a training signal can be constructed out of the data itself. Part of a record is withheld and the model is asked to account for what was withheld from what remains. Nothing in that task refers to a class. What the model is obliged to learn is how the measurements of a flow are put together, which of them follow from which, and what an ordinary combination of them looks like. The relation between features and labels, the relation that Section 2.1.3 described as moving, plays no part in the objective.

Methods of this kind divide according to where the comparison is made. A generative method restores the withheld input and is scored on how closely the restoration matches it. A joint-embedding method forms a representation of each of the two views and is scored on the agreement between those representations, so nothing is ever restored [12]. The distinction matters for flow records. Reproducing an exact byte count or an exact inter-arrival time commits the model to detail that is largely incidental to how a flow behaves, whereas a comparison made between representations is free to discard it.

2.3.1 Representation Collapse

An objective that rewards agreement between two representations admits a solution that satisfies it perfectly and means nothing. If the model maps every input to the same output, the two representations agree exactly, and the loss falls to its minimum and stays there. Read from the training curve the run looks like a model that has learned its task early and completely. Nothing has been learned. The representation carries no information about the record it was computed from, and the failure is invisible in the quantity being monitored, which is what makes it dangerous rather than merely inconvenient.

The objective therefore has to be arranged so that the trivial solution is not available. One established arrangement makes the two branches asymmetric: only one of them is trained by gradient descent, and the other is derived from it. It is not frozen, since a fixed target would give the trained branch nothing to follow, and it is not trained either. Its parameters are an exponential moving average of the trained branch's, so it changes at every step, but slowly, and it lags behind rather than being pulled toward the same trivial solution at the same rate [13]. No gradient reaches it, and the representation it produces is treated as a fixed quantity

by the loss.

2.3.2 Joint-Embedding Predictive Architectures

A JEPA puts these pieces together as shown in Figure 2.1. One record enters the architecture and two views of it are formed. The first has part of its content masked and is given to the context encoder, whose output is passed to a predictor, which is told which part was masked and produces from that an estimate. The second view is the record as it stands, unmasked, and it is given to the target encoder, the branch maintained by the moving average of Section 2.3.1. The estimate produced by the predictor and the representation produced by the target encoder are then compared, and the loss is that comparison.

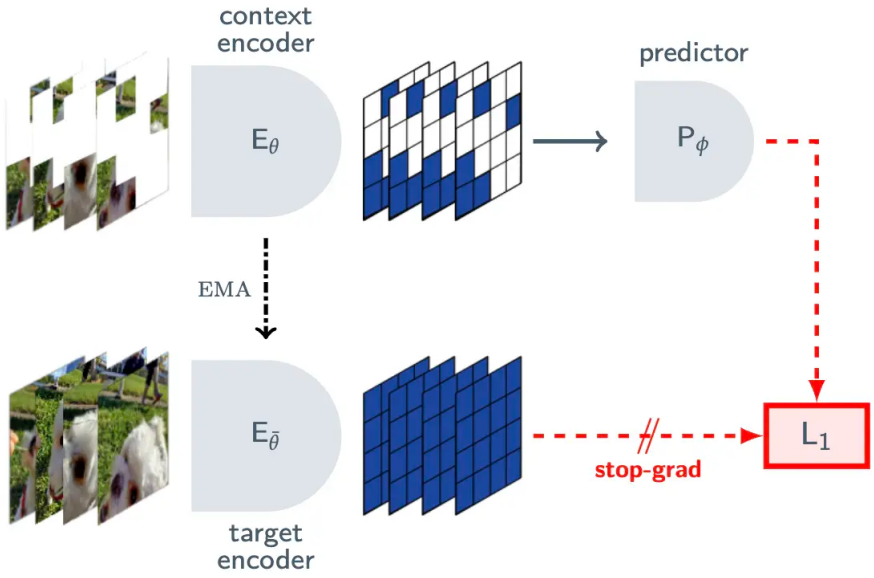


Figure 2.1: The joint-embedding predictive architecture as realized on video by V-JEPA [14]: context encoder E_θ, predictor P_ϕ and target encoder $E_{\bar{\theta}}$, the last updated by exponential moving average (EMA). Figure reproduced from [15].

The comparison takes place between representations and not between records. The predictor is never asked what the masked measurements were, only what the representation of the complete record looks like, and the target encoder supplies the answer. This is what places the architecture in the joint-embedding family rather than the generative one, and it is the sense in which the prediction is made in latent space.

The architecture was first realized on images, as I-JEPA, where the views are formed by masking blocks of an image and the prediction is made for the representations of those blocks [2]. It has since been carried to tabular data, as T-JEPA, where a record has no spatial structure to mask and the views are formed instead from subsets of its features [3]. The second of these is the setting of this thesis, since a flow record is a row of measurements rather than an image.

What such a model yields at the end of pre-training is an encoder, and what the encoder yields is a representation computed from the flow record alone. No label entered its objective at any point. Whether a representation obtained in this way serves a classifier better than the raw columns do, when the classifier is applied to traffic from a later year, is the question this thesis puts to the test. Chapter 3 places the approach against the work that surrounds it, and Chapter 4 describes the form in which it was implemented here.

Chapter 3

Related Work

3.1 Intrusion Detection under Concept Drift

The difficulty described in Section 2.1.3 is not new, and the security literature has approached it from two directions. One asks how a model that has aged should be evaluated, so that the aging is visible at all. The other asks what to do about it once it is.

- **TESSERACT** [16] addresses the first question. It identifies two sources of bias that inflate reported results: spatial bias, when the class distribution used in testing does not resemble a deployment, and temporal bias, when training and testing sets are split without regard to time. It proposes constraints on experiment design that remove both, together with a metric summarizing the robustness a classifier can be expected to retain, and demonstrates on three Android malware classifiers over a dataset of 129,000 applications spanning three years that earlier results had been biased.
- **Transcend** [7] addresses the second. Rather than adapting the model, it declines to use it where it should not be trusted: predictions that are likely to be wrong are rejected and quarantined for expert analysis, on a criterion drawn from conformal prediction theory.
- **TRANSCENDENT** [17] revisits that framework and gives it a formal treatment, refining the conformal evaluation underneath it and deriving two further evaluators that match or exceed the original at lower computational cost. It is evaluated on a five-year malware dataset constructed so as to remove the experimental bias present in the earlier evaluation.
- **INSOMNIA** [18] carries the problem into network intrusion detection specifically. It is a semi-supervised detector that updates its model continuously as

traffic characteristics move, using active learning to reduce the delay before an update and label estimation to reduce how much labeling that update requires, and it extends the TESSERACT evaluation framework from malware to the network domain.

- The general families, surveyed outside the security setting by Gama et al. [6], are the same three named in Section 2.1.3: retraining on a recent sample, detecting drift and reacting to it, and learning incrementally as data arrives.

What these approaches share is more instructive than what separates them. Every one of them consumes labels from the period the model has drifted into. Retraining needs a labeled sample of the new traffic; drift detection needs labels to confirm that what it detected was drift; active learning reduces the number of labels required but does not remove the requirement, and label estimation lowers their cost rather than the dependence on them. Rejection is the closest to an exception, since a rejected record is passed to an analyst rather than to a labeling procedure, but the analyst is the labeling procedure. The work in this thesis takes a different position on that constraint. It does not attempt to obtain labels for the later period more cheaply; it asks whether a different input, learned without labels, leaves less for the drift to damage.

3.2 Self-Supervised Representation Learning

Learning representations without labels has been studied largely outside the security domain, and the line of work most relevant here is the one concerned with the failure described in Section 2.3.1.

- **BYOL** [13] showed that a joint-embedding method needs no negative pairs. It trains two networks that learn from each other, one by gradient descent and the other as a slow-moving average of the first, and the asymmetry between them is what prevents the two from settling on a constant.

- **VICReg** [19] removes the asymmetry and regularizes the embeddings directly instead. A variance term keeps each dimension of the representation from collapsing, and a covariance term decorrelates the dimensions from one another so that they do not encode the same thing twice.

- **LeJEPA** [20] argues that the embeddings of a joint-embedding predictive architecture should follow an isotropic Gaussian distribution, and introduces Sketched Isotropic Gaussian Regularization to constrain them toward it. Its claim bears directly on the mechanism used in this thesis: it presents the regularizer as a replacement for the heuristics that surround JEPA training,

the stop-gradient, the teacher-student arrangement and the schedules that accompany them, at linear cost and with a single hyperparameter.

These are alternative answers to one question, how to keep a representation from degenerating, and the choice among them is a design decision rather than a settled matter. All three are demonstrated on data collected once. None of them speaks to whether a representation learned in this way still serves when it is applied to data recorded years after the data it was learned from, which is the question this thesis puts to them.

3.3 Joint-Embedding Predictive Architectures

The architecture itself was described in Section 2.3.2. What follows concerns the instantiations of it, and what each of them was asked to do.

- The architecture is proposed in LeCun's position paper [12] as one component of a larger argument about how machine intelligence should be organized, rather than as a method with an evaluation attached to it.
- **I-JEPA** [2] is the first realization. It operates on images: a context block of an image is encoded, and the representations of several other blocks of the same image are predicted from it.
- **V-JEPA** [14] carries the same objective to video, treating feature prediction as sufficient on its own for learning visual representations.
- **T-JEPA** [3] carries it to tabular data, which is the setting of this thesis. A record has no spatial structure to mask, so the two views are formed from disjoint subsets of its features, and the representation of one subset is predicted from the representation of the other. It reports improvements on classification and regression, and reaches or exceeds gradient-boosted trees on some of them.

The instantiation used in this thesis departs from these in a way that is worth stating precisely, because it is not a matter of hyperparameters. In I-JEPA, V-JEPA and T-JEPA the encoders are transformers, and a record, an image or a clip is represented by a set of embeddings, one for each token, patch or feature. The predictor accordingly predicts a set of embeddings, and it is told which positions to predict by means of positional mask tokens. The encoder used here is a multilayer perceptron, and it represents a flow record as a single pooled vector rather than as a set. The predictor therefore estimates one whole-record representation rather than a set of positional ones, and it is informed of what was masked by concatenating the binary mask vector to the context representation. The change is a simplification, and whether it costs anything is part of what the comparison in Chapter 5 measures.

3.4 The MAWIFlow Benchmark

The benchmark from which the data of this thesis is taken is itself a study of the problem this thesis addresses [4]. It was assembled because evaluations of intrusion detection are commonly performed on synthetic captures collected once, and it offers instead flow representations of real backbone traffic drawn from several years, so that a model can be tested on traffic recorded after the traffic it was fitted on. The samples, the labels and the binarization it defines are those described in Section 2.2.4.

Its own evaluation compares four conventional classifiers, the same families named in Section 2.1.2, against a CNN-BiLSTM model, in domain and across years. Its cross-temporal results are reproduced in Table 3.1. The finding it reports is that the tree-based classifiers degrade sharply when the training year is far from the test year, a random forest fitted on 2011 falling to an area under the curve of 0.61 on 2021, close to what an uninformative model would obtain, while the CNN-BiLSTM loses considerably less over the same interval.

Table 3.1: Cross-temporal baselines reported with the MAWIFlow benchmark. Every row is tested on the 2021 sample; the training year varies. Reproduced from the values published with the benchmark.

Model	Trained on	Accuracy	F1	AUC
Random Forest	2011-01	0.65	0.75	0.61
Random Forest	2016-01	0.57	0.67	0.55
Random Forest	2021-01	0.84	0.89	0.90
CNN-BiLSTM	2011-01	0.78	0.86	0.79
CNN-BiLSTM	2016-01	0.75	0.83	0.70
CNN-BiLSTM	2021-01	0.83	0.89	0.89

That finding is the closest precedent for the present work, and it also marks where the two part company. The benchmark's answer to degradation across years is a different classifier, a deeper one trained on the same labeled features. The question here is whether the input itself can be changed, leaving the classifiers as they are. The two are therefore not competing measurements of the same quantity, and the numbers in Table 3.1 are not a baseline this thesis improves upon. As Section 2.2.4 set out, the data is shared but the preparation is not: a different subset of the flow measurements is retained, a different scaling is applied, and the evaluation protocol differs. The baselines of this thesis are the ones it computes itself, on its own preparation of the data, and they are described in Chapter 4.

A second obstacle to reading the two sets of numbers side by side concerns the

class balance they were computed against. Accuracy, recall and precision are not independent of that balance. Writing P for the share of records in the positive class and taking recall and precision in their usual sense for that class, the counts of a confusion matrix give $TP = R\,P$ and $FP = TP(1 - Pr)/Pr$, from which accuracy follows as

$$A = 1 - P\left[(1 - R) + R\,\frac{1 - Pr}{Pr}\right], \tag{3.1}$$

so that P can be recovered from any three of the published figures. The figures in question are reproduced in Table 3.2. Applied to those fifteen rows, the recovered value is the same for every model within a given year, to within 5×10^{-4}: about 0.718 for 2011, 0.562 for 2016 and 0.740 for 2021. The first of these agrees with the composition of the published 2011 sample given in Table 2.1, where the share of records that are not benign is 71.23 %. The other two do not: the corresponding shares in the published samples are 20.57 % and 50.61 %.

Table 3.2: The accuracy, recall and precision reported with the MAWIFlow benchmark for models fitted and tested within one sample. These are the values to which Equation 3.1 is applied; the remaining columns of the published table are not used here.

Sample	Model	Accuracy	Recall	Precision
2011-01	Decision Tree	0.8334	0.8822	0.8853
	Random Forest	0.8579	0.9131	0.8916
	XGBoost	0.8535	0.9269	0.8761
	Logistic Regression	0.7899	0.9361	0.8036
	CNN-BiLSTM	0.8460	0.9147	0.8762
2016-01	Decision Tree	0.8237	0.8614	0.8312
	Random Forest	0.8321	0.8869	0.8271
	XGBoost	0.7930	0.7799	0.8404
	Logistic Regression	0.7504	0.7202	0.8145
	CNN-BiLSTM	0.7843	0.7764	0.8291
2021-01	Decision Tree	0.8199	0.8771	0.8792
	Random Forest	0.8407	0.8993	0.8870
	XGBoost	0.8361	0.8956	0.8844
	Logistic Regression	0.7798	0.9155	0.8113
	CNN-BiLSTM	0.8329	0.8900	0.8849

The discrepancy is not accounted for by the preprocessing the benchmark describes, since discarding records whose measurements are missing removes far too few of them to move a proportion by that margin, as Section 2.2.4 reports.

Several explanations remain open. The evaluation may have been performed on a resampled or balanced partition rather than on the samples as published; the samples released may not be identical to those the table was computed on; or the precision and recall reported may be averaged over both classes rather than given for the positive one, in which case Equation 3.1 does not apply and the recovered quantity is not a prevalence. Nothing in the present work distinguishes between these. What can be said is narrower and sufficient for the purpose at hand: for two of the three years, the class balance implied by the published metrics cannot be reconciled with the balance of the published samples, and metrics computed against different balances are not comparable. This is the same property that Section 2.1.2 identified for accuracy in general, and it is a further reason why the figures in Table 3.1 are reported here as context rather than used as a reference point.

3.5 Positioning of This Work

Taken together, the work above leaves a specific gap. The drift literature answers degradation with labels, whether by retraining, by detecting when retraining is due, or by spending an analyst's attention on the cases a model should not decide. The self-supervised literature learns representations without labels but demonstrates them on data collected at one time, so nothing in it establishes what such a representation is worth when the traffic has moved. And the benchmark that supplies the data answers the problem by changing the classifier rather than its input. What none of them reports is a controlled comparison in which the classifiers, the split and the evaluation are held fixed and the input alone is varied, across years. That comparison is what Chapter 4 constructs.

Chapter 4

Methodology

4.1 Experimental Design

The raw measurements of a flow record are a fixed description of it. They are whatever the exporter chose to compute, and each of them carries its meaning alone, so a change in what the traffic does can leave the numbers intact while altering what they imply. A representation is a looser object. It is not tied to any one measurement, and what it records is how a record stands in relation to the others, so it has room to keep a structure that individual columns lose. That is the argument for expecting it to survive drift better: it does not hold the feature-label relation in the first place, so there is less in it for the drift to invalidate.

The argument is only an argument. Turning it into an experiment requires that the representation be the sole thing that varies, because otherwise a difference in the outcome cannot be attributed to it. The design is therefore a comparison of two configurations that are identical in every respect except that one. Each of the two is referred to throughout this thesis as an *arm*: the *raw arm*, in which the classifiers are fitted on the preprocessed measurements of a flow record, and the *representation arm*, in which the same classifiers are fitted on the output of the encoder instead. In both arms the same three samples are used, split in the same way with the same seed, and the same four classifiers are fitted and evaluated by the same procedure against the same metrics. In the first arm the classifiers receive the preprocessed measurements directly. In the second they receive the output of an encoder that was pre-trained on those measurements without labels. Nothing else differs.

What such a design can establish is bounded, and the bound is worth stating before the results rather than after. It can show whether a representation obtained in this way helps, hurts or makes no difference, for these classifiers, on these three samples, under this preprocessing. It cannot show that no representation would

help, nor that a different architecture would behave the same way, nor that the effect would hold on traffic from another network. It is a controlled comparison of one input against another, and the strength of a controlled comparison is also its limit.

4.2 Preprocessing

The ninety columns of a sample are not a model input. The identifier and provenance columns described in Section 2.2.2 are set aside, leaving the seventy-six measurements; those are cleaned, reduced, scaled and concatenated with an encoding of the protocol, and the result is what both arms receive.

Thirteen of the seventy-six are removed because they carry nothing to learn from. Six hold a single value in all three samples, among them the three forward bulk-rate columns and both backward flag counts. Four are byte-identical copies of another column: the backward packet-length minimum, mean and average segment size are each exactly the backward packet-length maximum, and the forward packet-length minimum is exactly the overall packet-length minimum. The remaining three, the overall packet-length minimum, the forward packet-length mean and the forward average segment size, take at most eight distinct values across three million records. Removing them costs no information, and it matters for the pre-training that follows, where masking a column whose twin is left visible poses no question at all.

The sixty-three that remain are not scaled by one transform. The preprocessing originally applied, following the benchmark, normalized every numerical feature to a common range with a min-max scaler. Analysis of the three samples showed that this destroys most of the resolution the columns contain: between fifty-two and fifty-five of the seventy-six measurements have an interquartile range of zero, so a scaler fitted on quantiles collapses to unit scale and a symmetric clip flattens what is left. The three idle-time columns fall from roughly 290,000 distinct values to two, with the clip firing on 12 % to 20 % of rows. The columns are heavily zero-inflated and long-tailed, and no single global transform serves both that mass and the tail.

The measurements are therefore divided into four groups and scaled per group, as set out in Table 4.1. Eighteen well-behaved features are given a robust scaler fitted on the 1st to 99th percentile range and clipped at five scaled units. The remaining forty-five are given a quantile transform to a normal output, which preserves the ordering of values without depending on a spread the column does not have, and which needs no clipping. The protocol, which takes exactly three values in every sample, is one-hot encoded, and an unseen value raises rather than being silently encoded. Non-finite values are rewritten as missing and the affected rows are dropped, which removes at most two records in every thousand, as reported in Section 2.2.4. The model input is sixty-six dimensions.

Table 4.1: The four groups into which the retained measurements are divided, and the transform applied to each. Clipping is expressed in units of the scaled variable.

Group	Transform	Features	Clip
A	RobustScaler, 1–99 quantile range	1	±5
B	RobustScaler, 1–99 quantile range	17	±5
C	QuantileTransformer, normal output	33	none
D	QuantileTransformer, normal output	12	none
Retained measurements		63	
Protocol, one-hot		3	
Model input		**66**	

One consequence follows from doing it this way and should be recorded. The grouping and the transforms were chosen by analyzing these three samples. They are not a general recipe: a sample whose columns are distributed differently would need the analysis repeated and the groups redrawn. The preprocessing is fitted to the data of this thesis, deliberately, and it is a limitation of what the thesis measures rather than of what it claims.

4.3 The JEPA Model

The architecture is the one described in Section 2.3.2: a context encoder, a target encoder, a predictor and a masker. Figure 4.1 shows how they are arranged here, and what follows is the form each takes. Table 4.2 collects the settings.

Both encoders are multilayer perceptrons of the same shape, three hidden layers of 512, 256 and 256 units, each a linear map followed by layer normalization and a GELU activation, and then a projection to 64 dimensions. Layer normalization is used rather than batch normalization because it standardizes each record on its own and therefore does not couple the representation of a record to whichever other records happen to share its batch. GELU is used in preference to a rectified linear unit, which was less stable in this setting. Widening the hidden layers or adding to them did not improve the pre-training and at times degraded it, so the network is kept small. The projection dimension of 64 was chosen against measurement. The diagnostic used was RankMe [21], which summarizes how many directions of an embedding space a representation actually occupies, and which was proposed as a way of judging a pre-trained representation without labels. At 32 the space was too small; at 128 the encoder occupied only forty to fifty percent of the directions available to it, so the extra width was carrying nothing.

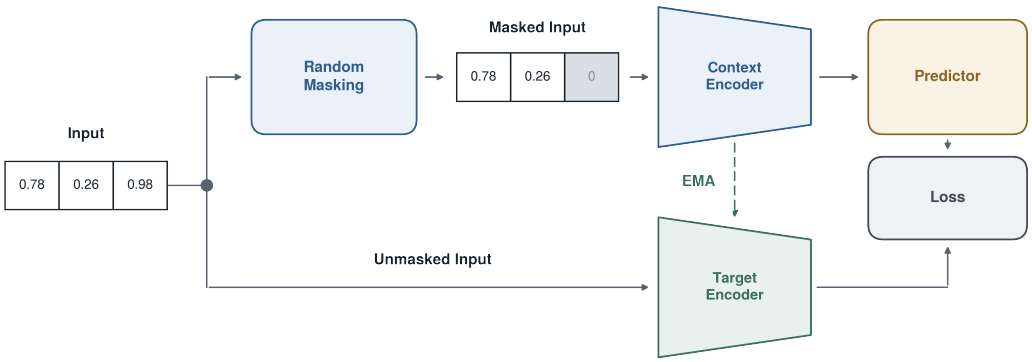


Figure 4.1: The model as implemented in this thesis, on a preprocessed flow record: the masked record feeds the context encoder and predictor, the unmasked record the target encoder, and the loss compares the two outputs.

The masker hides between eleven and twenty-two of the sixty-six input dimensions, drawn per record, and replaces what it hides with the constant -2.0. Two choices in it were settled by measurement rather than by preference. The first is that features are hidden in correlated groups rather than independently: clusters of columns whose absolute correlation reaches 0.9 are hidden together with probability 0.25. Hiding cells independently leaves a masked column's near-copy visible, and the prediction is then a lookup rather than an inference. Measured with a ridge probe on the 2011 sample, a per-cell mask allows the hidden cells to be recovered at a mean coefficient of determination of 0.713, and raising the per-cell rate does not change this, 0.713 at fifteen percent hidden against 0.712 at thirty-seven. Grouping lowers it by 0.171 at the same hidden share, and it is the only intervention tested that moved the figure at all. The second is the stand-in value. A constant far outside the data range was tried first and proved actively harmful, so the value was selected for how well a masked record remains recognizable as itself; -2.0 scored highest of the constants tested at 88.5 %, sits 1.51 units from the nearest real value in any column, and leaves the standard deviation of the input almost unchanged.

The predictor is a two-layer perceptron of 256 units. It receives the context representation concatenated with the binary mask, a 130-dimensional input, and returns an estimate in the projection space. Both I-JEPA and T-JEPA also condition their predictors on which positions are to be predicted, so conditioning is not a departure; the form of it is. They represent a record as one embedding per token or per feature and can therefore attach a learnable mask token to each position that needs predicting. The encoder used here pools all sixty-six inputs into a single vector, so no per-position representation exists to attach anything to, and

the mask can only be supplied as a vector alongside the pooled embedding. The pooling is what forces the difference, and it follows from the choice of a multilayer perceptron rather than a transformer.

The target encoder receives the record whole and unmasked, which is what both I-JEPA and T-JEPA do; they read out the target positions after encoding rather than before. Reading out per position is not available here, again because the encoder pools. The alternative, encoding a view restricted to the hidden features, was implemented and tested and collapsed during training, which is why the whole record is used.

The objective has two terms. The first is one minus the cosine similarity between the predictor's estimate and the target encoder's representation, both normalized, so the model is scored on direction rather than magnitude. The second is a spread term, the squared length of the mean of the normalized context representations over a batch, weighted at 0.1. It is small when the batch points in many directions and large when it points in one, and it is a second guard against the collapse described in Section 2.3.1, alongside the moving-average target. The target encoder is initialized from the context encoder and never receives a gradient; it is updated only as an exponential moving average of the context encoder at a decay of 0.996. These two together are not the only arrangement available: the literature offers regularizers that act on the embeddings directly, two of which were implemented and run for this work and are reported in Section 5.7.

The training schedule was settled in the same way. Each encoder is pre-trained for 200 epochs, a length arrived at by observation rather than by rule: shorter runs were still improving when they ended, and longer ones added nothing once the objective had leveled off. The learning rate is 1×10^{-4}, between two values that did not serve. At 1×10^{-5} the run failed to train, and at 1×10^{-3} the steps were large enough to put the run at risk. Batches hold 1024 records, which the size of the samples both permits and recommends, three million rows making smaller batches needlessly slow without making the gradient any more useful.

4.4 Downstream Classification

Pre-training produces two encoders and a predictor, of which only the context encoder is retained. Its parameters are frozen, meaning that no gradient reaches them and they take no further update of any kind, and it is thereafter used only to map records to representations.

The representation handed to a classifier is not the 64-dimensional projection used during pre-training. That projection is discarded with the predictor, and what is used is the output of the encoder's last hidden layer, 256 dimensions. This is a standard arrangement in self-supervised learning, where the projection serves

Table 4.2: The configuration under which each encoder was pre-trained. The same values are used for all three years.

	Setting	Value
Encoder	hidden widths	512, 256, 256
	projection dimension	64
	representation dimension	256
Predictor	hidden widths	256, 256
	input dimension	130
Masker	hidden features per row	11 to 22
	stand-in value	-2.0
	group masking rate	0.25
	correlation threshold	0.9
Objective	spread weight	0.1
	EMA decay	0.996
Optimizer	AdamW learning rate	1×10^{-4}
	weight decay	1×10^{-4}
	gradient clipping	1.0
	warmup fraction	0.02
Training	epochs	200
	batch size	1024
	seed	3407

the pre-training objective and the layer beneath it carries the representation that transfers.

Four classifiers are fitted on top: a logistic regression, a decision tree, a random forest and a gradient-boosted ensemble, XGBoost. Their settings are modest and fixed once for both arms. The logistic regression is allowed 5000 iterations so that it converges on either input; the random forest splits on entropy; the gradient-boosted ensemble uses one hundred trees of depth six at a learning rate of 0.1, drawing four fifths of the rows and four fifths of the columns for each of them. Each is given a means of not ignoring the smaller class under the imbalance described in Section 2.2.4: the first three by balanced class weights, and the gradient-boosted ensemble by a positive-class weight computed from the label counts of the training portion it is fitted on. In the representation arm they are fitted on the 256-dimensional encodings; in the raw arm the identical four are fitted on the sixty-six preprocessed columns. The classifiers, their settings, the seed, the split and the evaluation are the same in both arms. The input is the only difference between them.

4.5 Evaluation Protocol and Metrics

Each sample is shuffled once and divided into a training portion of eighty per cent and a test portion of twenty, using the same seed for both arms so that the partitions are identical. Everything that is fitted is fitted on the training portion of one sample and on nothing else: the scalers, the encoder, and the classifiers. Evaluation is performed only on test portions.

Pre-training is run once per year, on the unlabeled training portion of that year alone. No traffic from any other year is seen at any point in the process, either labeled or unlabeled. This is what makes the cross-year comparison a statement about transfer rather than about adaptation: when a model fitted on one year is applied to another, nothing in it has been exposed to the later traffic in any form.

Each of the three fitted years is then evaluated against the test portion of all three, giving nine train-test pairs per classifier per arm. Three of the nine are in domain, where the fitted year and the evaluated year coincide and no drift is present. The remaining six are cross-year. With four classifiers and two arms this gives seventy-two measurements.

Six quantities are computed for each of them: accuracy, precision, recall and the F_1 score at a decision threshold of 0.5, together with the area under the receiver operating characteristic curve and the area under the precision-recall curve. Section 5.1 defines them and states which comparison each serves. In brief, the two arms are evaluated on one and the same test partition, so any of the six compares them fairly, and F_1 is used; a comparison across samples, whose class balances

differ, is made only on the two threshold-free areas, and the degradation on which the thesis question turns is therefore computed on the area under the ROC curve.

Every stage draws on one seed, 3407, which fixes the split, the initialization, the masking stream and the classifiers. A repeated run reproduces the reported figures. What this does not provide is an estimate of how much the figures would move under a different seed, since each configuration was run once; the variability of the procedure is therefore not quantified, and differences of the order of a few thousandths should not be read as meaningful.

Chapter 5

Results

5.1 Metrics

Every classifier in this work returns, for each flow record, a score between zero and one. A score is turned into a decision by comparing it with a threshold, fixed here at 0.5: a record scoring above it is reported as anomalous, one scoring below it is passed as benign. Comparing those decisions with the labels gives four counts. A true positive (TP) is an anomalous record reported as such, a false positive (FP) a benign record reported as anomalous, a false negative (FN) an anomalous record passed as benign, and a true negative (TN) a benign record correctly passed.

Four of the six quantities reported here are computed from those counts, and therefore depend on where the threshold was placed.

- *Accuracy* is the share of all records decided correctly,

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \tag{5.1}$$

  It is the most direct summary and the most easily misread, because a sample in which one class is rare can be scored highly by a classifier that never reports that class at all.

- *Precision* is the share of the records reported as anomalous that really are,

$$\text{Precision} = \frac{TP}{TP + FP}. \tag{5.2}$$

  It answers how much of what the detector raises is worth an analyst's time.

- *Recall* is the share of the anomalous records that were reported,

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{5.3}$$

  It answers how much of what is there the detector finds.

- The F_1 *score* is the harmonic mean of precision and recall,

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\,TP}{2\,TP + FP + FN}, \tag{5.4}$$

  so it is high only when both are. It is the single figure most often used to summarize a detector, and it is the one on which the comparison between the two arms is read here.

The remaining two summarize the score itself rather than a decision taken from it, by considering every threshold at once.

- The *area under the ROC curve* plots the true positive rate against the false positive rate as the threshold is swept from one extreme to the other, and measures the area beneath. With $TPR = TP/(TP+FN)$ and $FPR = FP/(FP+TN)$,

$$\text{ROC-AUC} = \int_0^1 TPR(FPR)\, dFPR. \tag{5.5}$$

  It is the probability that a randomly chosen anomalous record is scored above a randomly chosen benign one. A value of 0.5 is what an uninformative score achieves; below 0.5 means the ordering is reversed.

- The *area under the precision-recall curve* does the same for precision against recall,

$$\text{PR-AUC} = \int_0^1 \text{Precision}(\text{Recall})\, d\text{Recall}. \tag{5.6}$$

  It is more sensitive than the first when one class is much smaller than the other, because it does not reward a classifier for correctly passing an abundant benign class.

The distinction matters for what may be compared with what. Both arms are evaluated on the same test partition of the same sample, so the class balance is identical between them, and any of the six quantities compares them fairly. Comparing one sample with another is a different matter: the balance differs by more than a factor of three between these three, as Section 2.2.4 reported, and a figure taken at a fixed threshold moves with that balance as much as with the quality of the decision. Wherever this chapter compares across samples, therefore, it uses the two threshold-free areas; wherever it compares the raw measurements against the representation on one and the same test set, it uses F_1.

5.2 Reading the Results

The comparison described in Chapter 4 produces seventy-two measurements: nine train-test pairs, four classifiers, two arms. As Section 4.1 set out, an arm is one of the two configurations being compared. They are identical in every respect but their input. The *raw arm* fits the classifiers on the sixty-six preprocessed measurements of a flow record; the *representation arm* fits the same four classifiers on the 256-dimensional output of the frozen encoder. Every table in this chapter puts the two side by side under the headings Raw and Repr., and a train-test pair is written as the fitted sample followed by the evaluated one.

Table 5.1 gives all seventy-two measurements under four of the six quantities defined above. Precision and recall were computed and are on record but are not reproduced, since F_1 summarizes both. Values are given to three decimals throughout this chapter, and wherever the text reports a change as a percentage, the percentage is relative to the value being changed and is computed from the unrounded measurements.

Three of the nine pairs are in domain, where the fitted and evaluated samples coincide, and six are cross-year. One property of the data governs how the table should be read and is treated separately in Section 5.5. Every cross-year measurement whose evaluated sample is 2011 gives an area under the ROC curve below 0.5, in both arms alike and for every classifier, and the two arms sit at almost the same mean. A value below 0.5 is worse than scoring at random, which is not degradation but reversal, and no property of either input explains a result that both inputs share.

5.3 In Domain

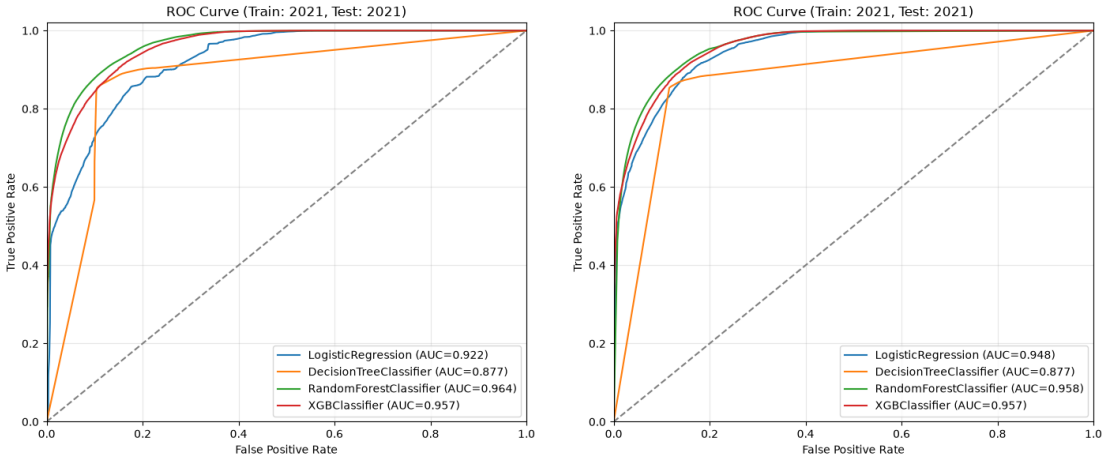
The three pairs on which the fitted and evaluated samples coincide are where drift is absent by construction, and they serve as a check on the whole comparison. If the two arms diverged here, the difference would be attributable to the representation itself rather than to anything about transfer.

They do not diverge. Of the twelve in-domain measurements the representation arm is higher in five on F_1 and in four on the area under the ROC curve, and on either metric the mean difference between the arms is under half a hundredth (Table 5.1). The largest movement in either direction belongs to the logistic regression, which gains between 2.5 % and 5 % of F_1 on the three samples; no other classifier moves by more than 1.5 % of its F_1 in either direction.

Figure 5.1 shows the 2021 pair, where the four curves in the two panels are close to indistinguishable. That check passes: replacing sixty-six preprocessed measurements with a representation learned without labels leaves in-domain performance where

Table 5.1: The complete set of measurements. Each row is one classifier applied to one train-test pair. Raw is the arm fitted on the preprocessed measurements, Repr. the arm fitted on the frozen representation. The classifiers are the logistic regression (LR), decision tree (DT), random forest (RF) and XGBoost (XGB). In each pair the higher of the two values is in bold.

		ROC-AUC		PR-AUC		F_1		Accuracy	
Fitted → Tested	Classifier	Raw	Repr.	Raw	Repr.	Raw	Repr.	Raw	Repr.
2011 → 2011	LR	0.746	**0.809**	0.842	**0.892**	0.807	**0.828**	0.731	**0.764**
	DT	**0.797**	0.791	**0.862**	0.857	**0.872**	0.870	**0.817**	0.814
	RF	**0.882**	0.866	**0.938**	0.925	**0.886**	0.876	**0.838**	0.824
	XGB	**0.855**	0.852	**0.924**	0.922	**0.856**	0.852	**0.800**	0.795
2011 → 2016	LR	**0.285**	0.163	**0.204**	0.124	**0.128**	0.094	**0.261**	0.252
	DT	0.291	**0.304**	0.168	0.168	0.170	**0.181**	0.245	**0.248**
	RF	**0.179**	0.173	0.127	0.127	0.090	**0.105**	0.234	0.234
	XGB	0.143	**0.150**	0.123	0.123	0.038	**0.056**	0.208	**0.213**
2011 → 2021	LR	0.534	**0.688**	0.489	**0.772**	0.226	**0.645**	0.395	**0.670**
	DT	0.366	**0.608**	0.469	**0.578**	0.253	**0.643**	0.384	**0.610**
	RF	0.302	**0.653**	0.401	**0.661**	0.175	**0.618**	0.402	**0.635**
	XGB	0.637	**0.680**	0.672	**0.732**	0.632	**0.662**	0.673	**0.688**
2016 → 2011	LR	**0.385**	0.363	**0.643**	0.636	**0.278**	0.265	0.316	**0.317**
	DT	0.455	**0.458**	0.699	0.699	**0.151**	0.144	0.299	0.299
	RF	0.337	**0.401**	**0.673**	0.671	**0.203**	0.196	0.303	**0.305**
	XGB	0.340	**0.346**	0.633	0.633	**0.267**	0.264	0.315	**0.316**
2016 → 2016	LR	0.942	**0.950**	0.733	**0.766**	0.704	**0.738**	0.829	**0.858**
	DT	**0.811**	0.800	**0.553**	0.536	**0.694**	0.684	**0.873**	0.871
	RF	**0.954**	0.947	**0.792**	0.764	**0.763**	0.759	0.883	**0.887**
	XGB	0.956	0.956	**0.809**	0.800	0.746	**0.749**	0.862	**0.865**
2016 → 2021	LR	**0.712**	0.652	**0.607**	0.596	**0.786**	0.539	**0.730**	0.519
	DT	0.517	**0.531**	0.527	**0.532**	0.385	**0.395**	0.508	**0.524**
	RF	0.649	**0.660**	**0.609**	0.601	**0.490**	0.472	0.520	**0.532**
	XGB	**0.689**	0.654	**0.655**	0.613	**0.791**	0.781	**0.736**	0.727
2021 → 2011	LR	**0.352**	0.321	**0.643**	0.618	**0.195**	0.135	**0.326**	0.307
	DT	0.436	**0.443**	**0.697**	0.696	**0.127**	0.121	**0.288**	0.285
	RF	0.328	**0.387**	0.642	**0.669**	**0.124**	0.117	**0.286**	0.282
	XGB	**0.351**	0.335	0.629	0.629	**0.132**	0.131	**0.285**	0.281
2021 → 2016	LR	0.909	**0.928**	0.582	**0.705**	0.644	**0.679**	0.855	**0.878**
	DT	**0.728**	0.722	**0.442**	0.424	**0.566**	0.556	**0.840**	0.834
	RF	**0.921**	0.878	**0.660**	0.605	**0.652**	0.615	**0.863**	0.852
	XGB	0.931	0.931	**0.722**	0.706	0.678	0.678	0.870	0.870
2021 → 2021	LR	0.922	**0.948**	0.910	**0.945**	0.844	**0.872**	0.840	**0.868**
	DT	0.877	0.877	0.830	**0.839**	**0.874**	0.869	**0.874**	0.868
	RF	**0.964**	0.958	**0.961**	0.953	**0.891**	0.887	**0.890**	0.885
	XGB	0.957	0.957	**0.958**	0.957	0.881	**0.885**	0.876	**0.879**



(a) Raw measurements **(b)** Representation

Figure 5.1: ROC curves for the four classifiers fitted and evaluated on the 2021 sample, with the raw measurements as input on the left and the frozen representation on the right.

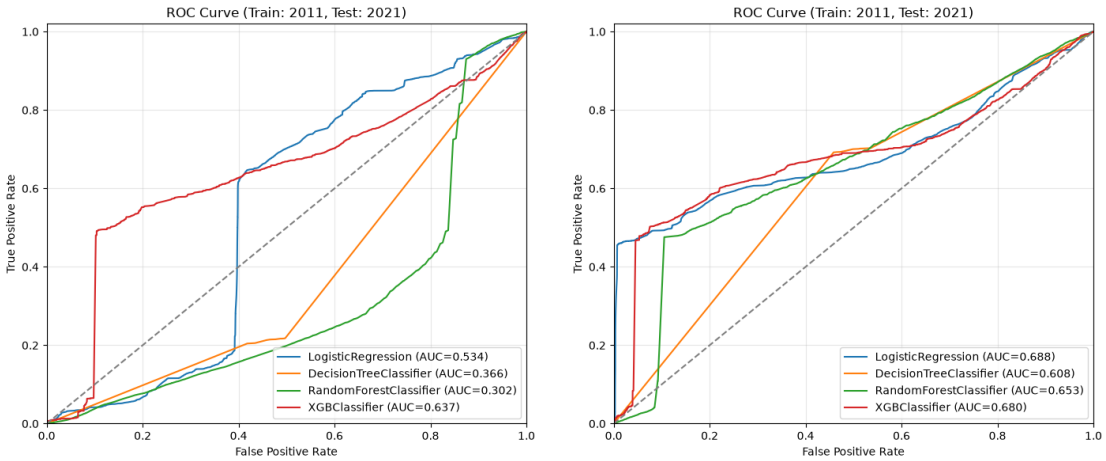
it was. One pattern within those twelve measurements is not uniform, and it is taken up in Section 5.6: the gains belong to one classifier.

5.4 Cross-Year Transfer

The six remaining pairs apply a classifier to a sample recorded years away from the one it was fitted on. Averaged over the four classifiers, four of the six directions carry little or no signal in either arm (Table 5.1). Both directions evaluated against 2011 sit below chance, as already noted, and fitting on 2011 and evaluating on 2016 is lower still, the furthest below chance of any pair. The two directions between 2016 and 2021 are the ones in which both arms rank substantially better than chance, and in both of them the arms sit within a few hundredths of each other, the raw arm slightly ahead.

The exception is the pair that spans the widest interval. Fitted on 2011 and evaluated on 2021, the raw arm averages below chance while the representation arm averages well above it, and the gain holds for every classifier. On the area under the ROC curve the random forest more than doubles, the decision tree gains about two thirds, the logistic regression about 30 % and XGBoost about 7 %. On F_1 the movement is larger still: the logistic regression, the decision tree and the random forest each reach between roughly two and a half and three and a half times their raw-arm value, and XGBoost gains about 5 %. Two of the four raw-feature

classifiers are below chance on this pair and all four representation classifiers are above it.



(a) Raw measurements **(b)** Representation

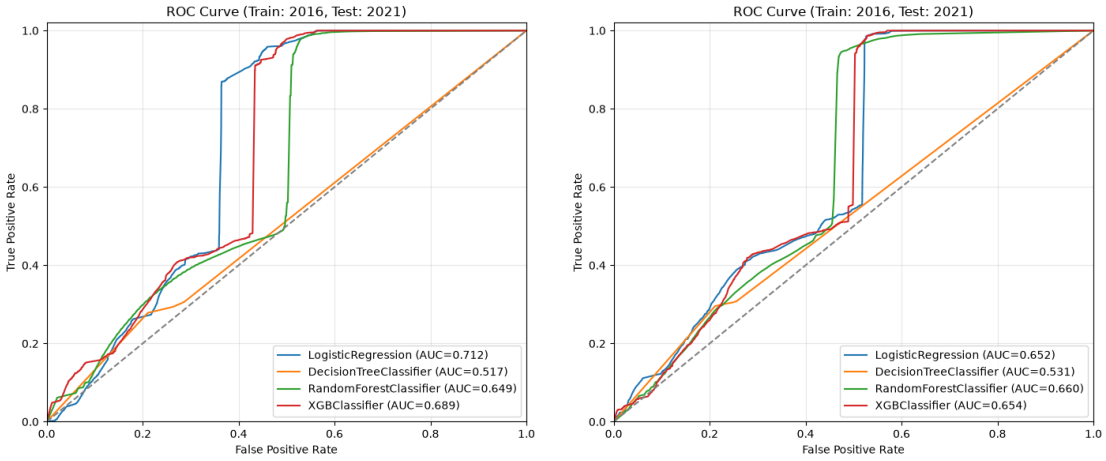
Figure 5.2: ROC curves for the four classifiers fitted on the 2011 sample and evaluated on the 2021 sample, with the raw measurements as input on the left and the frozen representation on the right. This is the pair on which the two arms differ most.

This is the largest effect in the table, and it is also the pair whose fitted sample is the one Section 5.5 sets aside. Both statements are true at once, and neither cancels the other.

Why the representation is less reversed than the raw measurements on this pair, and only on this pair, is not something the experiment can separate. The encoder was pre-trained on 2011 alone, so whatever it does here is a property of how it encodes 2011 traffic and not of anything learned from 2021; and the same encoder, evaluated on 2016, is reversed as badly as the raw measurements. The effect is recorded and is not built on.

It follows that any average taken over the whole table rests on this one group of four measurements. Over all thirty-six pairs of measurements, the representation arm is ahead in exactly half, and the mean difference in area under the ROC curve is about two hundredths in its favor. Removing the four measurements fitted on 2011 and evaluated on 2021 leaves it ahead in fewer than half and turns the mean slightly against it; removing every measurement in which 2011 appears on either side leaves sixteen, of which the representation arm wins six, with the mean slightly further against it. The median difference over the full table is zero to three decimals, which says the same thing in another way: the positive mean is not a

general tendency but a few large gains in one place.



(a) Raw measurements **(b)** Representation

Figure 5.3: ROC curves for the four classifiers fitted on the 2016 sample and evaluated on the 2021 sample, with the raw measurements as input on the left and the frozen representation on the right.

5.5 Degradation

The question the thesis puts is not whether one arm scores higher on a given pair, but whether the representation loses less when the evaluated sample is not the fitted one. That quantity is a difference of differences, and it is worth stating exactly how it is formed. For one classifier and one arm, take the value reached in domain, on the sample the classifier was fitted on, and subtract from it the value the same classifier reaches on the other sample. The result is what that classifier gives up in crossing between the two, and it is written Δ in the tables that follow. A larger number is a worse outcome. Comparing those numbers between the arms answers the thesis question.

Worked through for the random forest fitted on 2016: Table 5.1 gives 0.954 in domain and 0.649 on 2021 in the raw arm, a degradation of 0.305, which is 32 % of the in-domain value; the representation arm gives 0.947 and 0.660, a degradation of 0.287, or 30 %. The representation loses 0.018 less, about 6 % of the raw-arm Δ. Nothing in this quantity is a training loss; it is composed entirely of values printed in Table 5.1.

It is computed on the area under the ROC curve rather than on F_1, and the reason is the one given in Section 5.1. The two values being subtracted come from

different samples, whose class balances differ, so a figure taken at a fixed threshold is not subtractable between them. The areas are.

Taking the difference over all six cross-year directions would also not be informative, because four of the six carry little signal. The 2011 sample is the source of that problem, and two measured properties of it explain why it should not be treated as an ordinary sample. First, it is not spread over its month: 74.9 % of its records fall on the first three days of January, and 33.1 % on the first day alone; the three busiest days of the 2016 sample hold 15.9 % of its records, and those of the 2021 sample 27.1 %. Second, and decisively, every classifier evaluated against it scores below chance regardless of its input. A property shared by both arms is a property of the sample, not of the method.

What reverses can be read from the measurements themselves, one input at a time. For an input j and one sample, take the value of that input alone as the score and compute the area under the ROC curve it reaches against the labels of that sample, exactly as Equation 5.5 defines it. Written out, with $\mathcal{A}$ the anomalous records of the sample and $\mathcal{B}$ the benign ones,

$$\mathrm{AUC}_j = \frac{1}{|\mathcal{A}|\,|\mathcal{B}|} \sum_{a \in \mathcal{A}} \sum_{b \in \mathcal{B}} \Big[\mathbb{1}(x_{aj} > x_{bj}) + \tfrac{1}{2}\,\mathbb{1}(x_{aj} = x_{bj}) \Big], \tag{5.7}$$

where x_{aj} is the value of input j in record a and $\mathbb{1}$ is one when its condition holds and zero otherwise. It is the probability that a randomly chosen anomalous record has a larger value of input j than a randomly chosen benign one, ties counting one half, so it reads directly as a percentage of record pairs: the entry of 0.919 for Total Fwd Packet in 2016 says that in 91.9 % of anomalous–benign pairs the anomalous record has more forward packets, and the entry of 0.391 for the same input in 2011 says that there it is the benign record in most pairs. Chance is 50 %. A value above it means that larger values of the input go with the anomalous label in that sample, a value below it that they go with the benign label, and the distance from it says how strongly. The quantity depends only on the ordering of the values, so the scaling of Section 4.2 does not affect it. It was computed for each of the sixty-six inputs on every record of each cleaned sample, and Table 5.2 reports all of them. Reading one input's row across the three samples shows whether that input separates the two classes in the same direction in each of them.

Table 5.2: Area under the ROC curve of each of the sixty-six model inputs used on its own as a score, in each sample, by Equation 5.7. A value of 0.5 is chance; above it, larger values of the input go with the anomalous label, below it with the benign label. Computed on every record of the cleaned samples.

Input	2011	2016	2021	Input	2011	2016	2021
Fwd Seg Size Min	0.549	0.847	0.371	Down/Up Ratio	0.653	0.125	0.346
Flow Duration	0.364	0.754	0.733	Average Packet Size	0.526	0.496	0.450
Total Fwd Packet	0.391	0.919	0.840	Bwd Bytes/Bulk Avg	0.499	0.500	0.499
Flow Packets/s	0.645	0.362	0.402	Bwd Packet/Bulk Avg	0.499	0.500	0.499
Flow IAT Mean	0.354	0.600	0.590	Bwd Bulk Rate Avg	0.499	0.500	0.499
Flow IAT Std	0.490	0.822	0.564	Subflow Fwd Bytes	0.500	0.500	0.507
Flow IAT Max	0.361	0.701	0.582	Subflow Bwd Packets	0.499	0.477	0.472
Flow IAT Min	0.394	0.324	0.573	Subflow Bwd Bytes	0.501	0.500	0.497
Fwd IAT Total	0.340	0.869	0.746	Bwd Init Win Bytes	0.521	0.496	0.401
Fwd IAT Mean	0.326	0.844	0.593	Fwd Act Data Pkts	0.498	0.501	0.504
Fwd IAT Std	0.488	0.824	0.606	Active Mean	0.482	0.653	0.488
Fwd IAT Max	0.334	0.868	0.593	Active Std	0.500	0.530	0.507
Fwd IAT Min	0.317	0.818	0.613	Active Max	0.482	0.653	0.489
Fwd Header Length	0.469	0.892	0.821	Active Min	0.482	0.652	0.487
Fwd Packets/s	0.639	0.409	0.420	Idle Mean	0.480	0.695	0.491
SYN Flag Count	0.465	0.759	0.345	Idle Std	0.502	0.560	0.504
Subflow Fwd Packets	0.346	0.742	0.457	Idle Max	0.481	0.694	0.491
FWD Init Win Bytes	0.496	0.827	0.402	Idle Min	0.480	0.694	0.491
Total Bwd packets	0.646	0.122	0.282	Fwd Packet Length Max	0.526	0.496	0.450
Total Length of Fwd Packet	0.525	0.496	0.452	Fwd Packet Length Std	0.500	0.500	0.500
Total Length of Bwd Packet	0.527	0.495	0.442	Bwd Packet Length Max	0.528	0.495	0.442
Flow Bytes/s	0.522	0.496	0.448	Fwd PSH Flags	0.491	0.515	0.497
Bwd IAT Total	0.506	0.496	0.442	Fwd URG Flags	0.500	0.500	0.500
Bwd IAT Mean	0.506	0.496	0.442	Packet Length Max	0.526	0.496	0.450
Bwd IAT Std	0.509	0.497	0.452	Packet Length Mean	0.526	0.496	0.450
Bwd IAT Max	0.506	0.496	0.442	Packet Length Std	0.500	0.500	0.500
Bwd IAT Min	0.505	0.496	0.443	Packet Length Variance	0.500	0.500	0.500
Bwd Header Length	0.660	0.489	0.301	FIN Flag Count	0.513	0.721	0.495
Bwd Packets/s	0.667	0.122	0.282	RST Flag Count	0.613	0.571	0.414
PSH Flag Count	0.484	0.760	0.505	URG Flag Count	0.500	0.500	0.500
ACK Flag Count	0.578	0.815	0.471	Protocol = 0	0.461	0.152	0.470
CWR Flag Count	0.498	0.500	0.481	Protocol = 6	0.516	0.846	0.355
ECE Flag Count	0.498	0.501	0.483	Protocol = 17	0.523	0.502	0.675

Three things can be read from Table 5.2. First, how much direction there is to compare. Many inputs sit within five percentage points of chance in one sample or another and carry no direction there; those are set aside in what follows. The inputs of 2011 barely separate the classes at all, on average 4.9 percentage points away from the 50 % of chance over the sixty-six, while those of 2016 separate them strongly, at 13.5 points, and those of 2021 sit between, at 7.1 points. Second, which way the remaining inputs point. Between 2016 and 2021, sixteen inputs point the same way and six the opposite way. Between 2011 and 2016, three point the same way and fourteen the opposite; between 2011 and 2021, none point the same

way and sixteen the opposite. The reversal is not subtle: a longer flow with more forward packets makes a record less likely to be labeled anomalous in 2011 and far more likely in the two later samples, and the backward-packet counts move the other way (Total Fwd Packet, Flow Duration and Total Bwd packets in the table). Third, what this does to a classifier. One fitted on 2011 carries into either later sample a set of input directions of which every one that matters is inverted, and a score built from inverted directions ranks the later traffic backwards. That is what the classifiers' areas below chance in both arms record: a change in what the labels mean, not in what the traffic looks like. It also says why the reversal is deepest from 2011 to 2016: a classifier carrying 2011's directions is punished hardest where the later sample's own signal is strongest, and 2016 is where it is strongest.

The sound comparison is therefore between 2016 and 2021, in both directions, and it is given in Table 5.3, which reports Δ both as a difference and as a share of the in-domain value.

Table 5.3: Degradation of the area under the ROC curve between 2016 and 2021, as defined in Section 5.5: Δ is in domain minus cross-year, relative Δ is that difference as a share of the in-domain value. Values from Table 5.1; the smaller of each pair is in bold.

	Fitted 2016, tested 2021				Fitted 2021, tested 2016			
	Δ		Relative Δ		Δ		Relative Δ	
Classifier	Raw	Repr.	Raw	Repr.	Raw	Repr.	Raw	Repr.
LR	**0.230**	0.298	**24 %**	31 %	**0.013**	0.020	**1 %**	2 %
DT	0.293	**0.268**	36 %	**34 %**	**0.149**	0.156	**17 %**	18 %
RF	0.305	**0.287**	32 %	**30 %**	**0.043**	0.081	**4 %**	8 %
XGB	**0.267**	0.302	**28 %**	32 %	**0.026**	0.027	3 %	3 %

The representation loses less in two of the eight comparisons and more in six, and the relative Δ tells the same story as the absolute. From 2016 to 2021 it reduces the degradation for the decision tree and the random forest, in each case by under a tenth of the raw-arm Δ, while increasing it for the logistic regression by about 30 % and for XGBoost by about 13 %. From 2021 to 2016, where the raw arm loses little in the first place, it increases the degradation in all four, most for the random forest, whose Δ nearly doubles.

The representation does not reduce the degradation. On the pair where the comparison is sound, it is as exposed to the change between samples as the raw measurements are, and in more comparisons than not it is slightly more exposed. Section 5.4 showed that it can raise the cross-year *level* substantially, on the widest interval by over 40 % of the raw-arm mean; this section shows that raising the level

and reducing the degradation are not the same thing, and that only the first of them happened.

5.6 The Representation and the Choice of Classifier

The in-domain measurements left one pattern unexplained. The gains are not spread across the classifiers. On F_1 the logistic regression is ahead in all three samples, the decision tree and the random forest are behind in all three, and XGBoost barely moves in either direction (Table 5.1). The representation helps the one classifier that is linear and none of the three that are not.

That pattern calls for the measurement representation learning uses to judge an encoder, the linear probe. A linear classifier can separate only what is linearly separable in its input, so its performance on a representation, set against its performance on the raw measurements, shows how much of the structure in the data the encoder has made linearly accessible. What that gain should be measured against is the most that any classifier extracts from the raw measurements, which on tabular data is a gradient-boosted ensemble (Section 2.1.2). The distance between the logistic regression and XGBoost on the raw measurements is therefore the room the representation has to close, and the share of it closed is the quantity of interest. Write x for the value XGBoost reaches on the raw measurements, ℓ_{raw} for the value the logistic regression reaches on the same input, and ℓ_{repr} for the value it reaches on the representation, all three in domain on the same sample. The shortfall of the linear classifier on raw measurements is $x - \ell_{\text{raw}}$, and on the representation it is $x - \ell_{\text{repr}}$. The share of that shortfall which the representation removes is

$$1 - \frac{x - \ell_{\text{repr}}}{x - \ell_{\text{raw}}}. \tag{5.8}$$

Table 5.4 gives the three values, the two shortfalls and the share for each sample, on both the area under the ROC curve and F_1.

Worked through on the 2021 sample and the area under the ROC curve: fitted on the raw measurements, the logistic regression falls 0.036 short of XGBoost; fitted on the representation, it falls 0.009 short. By Equation 5.8 the representation has removed 74 % of the shortfall. Across the table the share lies between a half and three quarters on the area under the ROC curve, and from two fifths to just over four fifths on F_1, and it is positive in every sample and on both metrics.

No ensemble of the two arms is formed anywhere in this comparison, and no classifier is combined with another. Each figure in Table 5.4 is a single classifier fitted on a single input and evaluated on a single test partition, taken from Table 5.1.

Table 5.4: Shortfall of the logistic regression (LR) against XGBoost (XGB) in domain, on the raw measurements and on the representation, and the share of it the representation removes by Equation 5.8. Values from Table 5.1; gaps and shares computed from the unrounded measurements.

		LR		XGB	Gap to XGB		
Metric	Sample	Raw	Repr.	Raw	Raw	Repr.	Removed
ROC-AUC	2011	0.746	0.809	0.855	0.109	0.046	58 %
	2016	0.942	0.950	0.956	0.014	0.006	58 %
	2021	0.922	0.948	0.957	0.036	0.009	74 %
F_1	2011	0.807	0.828	0.856	0.049	0.028	42 %
	2016	0.704	0.738	0.746	0.042	0.008	82 %
	2021	0.844	0.872	0.881	0.036	0.009	76 %

What the representation buys, in domain, is therefore not accuracy but linear separability. XGBoost on the raw measurements already extracts what the columns contain and gains nothing from the encoding; a linear classifier cannot, and gains most of what separates it from XGBoost. The practical reading is left to Chapter 6, but the measurement itself is narrow and consistent: the direction is the same in all three samples and on both metrics.

5.7 Alternative Regularizers

Section 4.3 described the mechanism that keeps the representation from degenerating: an exponential moving average of one encoder into the other, together with a spread term in the loss. That is one choice among several, and Section 3.2 named two alternatives from the literature. Both were implemented and run for this work, and this section reports what they gave.

One caution applies to everything below. Both were run before the pipeline described in Chapter 4 was settled, on a version with seventy-nine input features, a different masking constant and a different embedding width. Their figures cannot be placed beside the rest of this chapter. What they can show is whether either mechanism changes the outcome, and on that they agree.

VICReg replaces the asymmetry between the two encoders with two penalties applied directly to the embeddings, one holding the variance of each dimension up and one pushing the correlation between dimensions down. Its effect on the representation was large and easily measured. The effective rank, measured with RankMe [21], rose more than tenfold from the value with the penalty switched

off to its best setting, and above the value reached at the weights the original paper proposes: the representation went from occupying a handful of directions to occupying dozens. Its effect on the classifiers was nil. Across the same thirty-six pairs the outcome against the raw arm was parity, ahead in about half on F_1 and in about a third on the area under the ROC curve.

SIGReg constrains the embeddings toward an isotropic Gaussian instead. It was run as a controlled comparison against the identical configuration with the regularizer switched off, so the two runs differ in nothing else. In domain it changed nothing: on every in-domain pair the four-classifier mean moved by less than two thousandths. Over all thirty-six pairs it was neutral to negative, ahead on the area under the ROC curve in about two fifths and behind in three fifths.

For SIGReg the reason was recorded before the run rather than after it. The regularizer works by testing whether the embeddings, projected onto random directions, look Gaussian. At an embedding width of sixty-four, any such projection sums sixty-four coordinates and is approximately Gaussian by the central limit theorem, whatever the underlying distribution. The test therefore cannot distinguish the shapes it exists to distinguish at this width, and what remains of its effect is protection against collapse. The run it was added to was not collapsing, so there was nothing for it to buy.

Both point the same way. Each mechanism improved the geometric property it was designed to improve, and neither improvement reached the classifier. One caveat governs both: each was run once, at a single setting of its weight, with no repetition, and every aggregate difference reported above is below a hundredth on a scale from zero to one.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis asked whether a representation learned from network traffic without labels holds up better than the raw measurements when a classifier is applied to traffic recorded years after the traffic it was fitted on. The question was put as a controlled comparison: the same three samples, the same split, the same four classifiers, the same metrics, and only the input to those classifiers varying between the two arms.

The answer is no. On the pair of samples where the comparison is sound, the representation reduces the degradation of area under the ROC curve in two of eight comparisons and increases it in six. Where it does help, it raises the level at which a classifier operates on a later sample rather than reducing how much that classifier gives up in crossing to it. Raising the level and resisting drift are different properties, and only the first was observed.

That result should be read with the weight of the evidence behind it. The single large cross-year gain, on the pair fitted in 2011 and evaluated in 2021, carries the average over the whole table: removing those four measurements turns a small mean advantage into a small mean disadvantage, and removing every measurement involving 2011 leaves the representation ahead in six of sixteen. That sample is also the one this thesis has shown to be unlike the others, concentrated on three days and scored below chance by every model in both arms. The strongest evidence for the representation therefore comes from the least trustworthy part of the data, and the thesis does not rest a claim on it.

Two further observations were made in the course of answering the question, and neither reverses it. In domain, where drift is absent, the two arms are level,

and under the gradient-boosted ensemble they differ by less than 0.005 of F_1 and of area under the ROC curve in every sample. Whatever the encoder produces, a boosted ensemble on the raw measurements already extracts it. What the encoder does supply is linear separability: a logistic regression fitted on the representation closes 58 %, 58 % and 74 % of the distance to a boosted ensemble on the raw measurements, in the three samples respectively. The direction is consistent and the mechanism is plain, but it is a statement about what a weak classifier can reach, not about drift.

6.2 Limitations

Five limitations bound what the preceding chapters establish, and each of them was visible while the work was done rather than discovered afterward.

Every configuration was run once, under a single seed. No estimate exists of how far a figure would move under another, so differences of a few thousandths carry no weight, and several of the comparisons reported here fall into that range.

The data is three samples from one measurement point on one backbone link. The result is a statement about this traffic. Nothing in it establishes how a representation would behave on a network with a different mix of services or a different population of users.

The encoder is a multilayer perceptron, which pools a record into a single vector. That choice dictated the form of the masking, the conditioning of the predictor and the target view, as Chapter 4 described. A transformer over per-feature tokens would not face those constraints, and this thesis says nothing about how one would behave.

The preprocessing was fitted to these three samples. The grouping of features and the transform applied to each were chosen by examining their distributions, so a sample whose columns are distributed differently would require the analysis to be repeated.

Finally, a third of the evaluation matrix is not usable. The 2011 sample is concentrated on three days of its month and is scored below chance by every model in both arms, which removes the four measurements evaluated against it and weakens the four fitted on it.

6.3 Future Work

The limitations point directly at what should follow.

The first is replication. With one run per configuration the experiment cannot distinguish a real difference of a hundredth from noise, and several of the comparisons in Chapter 5 sit at that scale. Repeating the whole matrix under several seeds

would either sharpen those comparisons or show them to be indistinguishable, and either outcome is more informative than the present one.

The second is more data, of two kinds. More samples from the same archive would give the cross-year comparison more than two sound directions to rest on, and would dilute the influence any single sample has on an average. Samples from a different measurement point would test whether anything observed here is a property of representation learning or of this particular link.

The third follows from the one effect that was consistent. The representation closed most of the gap between a linear classifier and a boosted ensemble, in every sample and in the same direction. Whether that holds on other flow data, and whether the size of the gap it closes is predictable from properties of the input, is a narrower question than the one this thesis asked and appears to be a more tractable one.

Bibliography

[1] P. García-Teodoro, J. Díaz-Verdejo, G. Maciá-Fernández, and E. Vázquez. "Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges". In: *Computers & Security* 28.1–2 (2009), pp. 18–28. DOI: 10.1016/j.cose.2008.08.003 (cit. on pp. 1, 5).

[2] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. "Self-Supervised Learning from Images with a Joint-Embedding Predictive Architecture". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 15619–15629. arXiv: 2301.08243 [cs.CV] (cit. on pp. 2, 14, 17).

[3] Hugo Thimonier, José Lucas De Melo Costa, Fabrice Popineau, Arpad Rimmel, and Bich-Liên Doan. "T-JEPA: Augmentation-Free Self-Supervised Learning for Tabular Data". In: *International Conference on Learning Representations (ICLR)*. 2025. arXiv: 2410.05016 [cs.LG] (cit. on pp. 2, 14, 17).

[4] Joshua Schraven, Alexander Windmann, and Oliver Niggemann. "MAWI-Flow Benchmark: Realistic Flow-Based Evaluation for Network Intrusion Detection". In: *Proceedings of the 12th International Conference on Information Systems Security and Privacy (ICISSP)*. Vol. 1. SCITEPRESS, 2026, pp. 549–556. ISBN: 978-989-758-800-6. DOI: 10.5220/0014463900004061. arXiv: 2506.17041 [cs.LG] (cit. on pp. 2, 8, 10, 18).

[5] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. "Why Do Tree-Based Models Still Outperform Deep Learning on Typical Tabular Data?" In: *Advances in Neural Information Processing Systems 35 (NeurIPS 2022), Datasets and Benchmarks Track*. 2022. arXiv: 2207.08815 [cs.LG] (cit. on p. 6).

[6] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. "A Survey on Concept Drift Adaptation". In: *ACM Computing Surveys* 46.4 (2014), 44:1–44:37. DOI: 10.1145/2523813 (cit. on pp. 7, 16).

[7] Roberto Jordaney, Kumar Sharad, Santanu Dash, Zhi Wang, Davide Papini, Ilia Nouretdinov, and Lorenzo Cavallaro. "Transcend: Detecting Concept Drift in Malware Classification Models". In: *26th USENIX Security Symposium*. USENIX Association, 2017, pp. 625–642 (cit. on pp. 7, 15).

[8] MAWI Working Group. *MAWI Working Group Traffic Archive*. WIDE Project. URL: http://mawi.wide.ad.jp/mawi/ (visited on 09/09/2026) (cit. on p. 8).

[9] Arash Habibi Lashkari, Gerard Draper-Gil, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. "Characterization of Tor Traffic Using Time Based Features". In: *Proceedings of the 3rd International Conference on Information Systems Security and Privacy (ICISSP)*. SCITEPRESS, 2017, pp. 253–262. ISBN: 978-989-758-209-7. DOI: 10.5220/0006105602530262 (cit. on p. 8).

[10] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. "Characterization of Encrypted and VPN Traffic Using Time-Related Features". In: *Proceedings of the 2nd International Conference on Information Systems Security and Privacy (ICISSP)*. SCITEPRESS, 2016, pp. 407–414. ISBN: 978-989-758-167-0. DOI: 10.5220/0005740704070414 (cit. on p. 8).

[11] Romain Fontugne, Pierre Borgnat, Patrice Abry, and Kensuke Fukuda. "MAWILab: Combining Diverse Anomaly Detectors for Automated Anomaly Labeling and Performance Benchmarking". In: *Proceedings of the 6th International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*. ACM, 2010, 8:1–8:12. DOI: 10.1145/1921168.1921179 (cit. on p. 9).

[12] Yann LeCun. *A Path Towards Autonomous Machine Intelligence*. Version 0.9.2. 2022. URL: https://openreview.net/pdf?id=BZ5a1r-kVsf (cit. on pp. 12, 17).

[13] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. "Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning". In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*. 2020, pp. 21271–21284. arXiv: 2006.07733 [cs.LG] (cit. on pp. 12, 16).

[14] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. "Revisiting Feature Prediction for Learning Visual Representations from Video". In: *Transactions on Machine Learning Research* (2024). arXiv: 2404.08471 [cs.CV] (cit. on pp. 13, 17).

[15] Rohit Bandaru. *Deep Dive into Yann LeCun's JEPA*. July 31, 2024. URL: https://rohitbandaru.github.io/blog/JEPA-Deep-Dive/ (visited on 09/15/2026) (cit. on p. 13).

[16] Feargus Pendlebury, Fabio Pierazzi, Roberto Jordaney, Johannes Kinder, and Lorenzo Cavallaro. "TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time". In: *28th USENIX Security Symposium*. USENIX Association, 2019, pp. 729–746 (cit. on p. 15).

[17] Federico Barbero, Feargus Pendlebury, Fabio Pierazzi, and Lorenzo Cavallaro. "Transcending TRANSCEND: Revisiting Malware Classification in the Presence of Concept Drift". In: *43rd IEEE Symposium on Security and Privacy (S&P)*. IEEE, 2022, pp. 805–823. DOI: 10.1109/SP46214.2022.9833659. arXiv: 2010.03856 [cs.CR] (cit. on p. 15).

[18] Giuseppina Andresini, Feargus Pendlebury, Fabio Pierazzi, Corrado Loglisci, Annalisa Appice, and Lorenzo Cavallaro. "INSOMNIA: Towards Concept-Drift Robustness in Network Intrusion Detection". In: *Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security (AISec)*. ACM, 2021, pp. 111–122. DOI: 10.1145/3474369.3486864 (cit. on p. 15).

[19] Adrien Bardes, Jean Ponce, and Yann LeCun. "VICReg: Variance-Invariance-Covariance Regularization for Self-Supervised Learning". In: *International Conference on Learning Representations (ICLR)*. 2022. arXiv: 2105.04906 [cs.CV] (cit. on p. 16).

[20] Randall Balestriero and Yann LeCun. *LeJEPA: Provable and Scalable Self-Supervised Learning Without the Heuristics*. 2025. arXiv: 2511.08544 [cs.LG] (cit. on p. 16).

[21] Quentin Garrido, Randall Balestriero, Laurent Najman, and Yann LeCun. "RankMe: Assessing the Downstream Performance of Pretrained Self-Supervised Representations by Their Rank". In: *Proceedings of the 40th International Conference on Machine Learning (ICML)*. 2023. arXiv: 2210.02885 [cs.LG] (cit. on pp. 23, 40).