軽量深層学習を用いた画像に基づく
高精度マルウェア分類

防衛大学校理工学研究科後期課程

電子情報工学系専攻　情報知能メディア学教育研究分野

ダオ・ヴァン・トゥアン

令和６年３月

目次

図目次

表目次

第 1 章

序論

この章では，まず研究背景として，本研究のテーマであるマルウェア分類に対しこれまで取られてきたアプローチについて紹介する．これらを踏まえ，本研究が想定する状況や取るべき立場を説明した後，本研究の目的とともに，得られた研究成果についても述べる．最後に本論文の構成を示す．

1.1 研究背景

1.1.1 マルウェア被害の現状

デジタル時代において，悪意のある行為者は，脆弱性を悪用しシステムにマルウェアを感染させる新たな方法を絶えず模索している．マルウェアはユーザーの個人情報，銀行情報，クレジットカード情報などを盗み (情報窃取)，ランサムウェアなどを使用してユーザーから身代金を要求し (金銭的利益)，システムやネットワークを破壊し，サービスを妨害する．マルウェアは，政府や企業の機密情報を盗むために使用されるスパイ活動，大規模なサイバー攻撃を実行するためのツールなど様々な目的で作成されている．マルウェアは，インターネット上で容易に作成できるツール [1, 2] を利用したり，また既知のマルウェアを修正することで簡単に作成できる [3]．マルウェアは有効な攻撃手段となっており，そのため進化と変異を続け，個人，企業，政府に対し容赦のない課題を突きつけている [4].

マルウェアによるダメージは深刻なため，様々な対策が講じられているが，新たに出現するマルウェアの数は一向に減る気配がない．その理由として，少なくとも次の三つが考えられる．一つ目はマルウェア分類技術の遅れである. 二つ目は IoT デバイスなどでも動作する軽量でありながら高性能なマルウェア分類手法の欠如である．三つ目は未知のマルウェア分類を可能にする手法の欠如である．ここで，マルウェア分類とは，検出したマルウェアの種類を予測する課題である．

マルウェアを正しく分類することによって，被害の範囲や影響をより正確に把握できる．特定のマルウェアファミリーが特定の脆弱性を標的にしている場合，その脆弱性を修正することで対処できる可能性がある．迅速にマルウェアを分類することで，被害の拡大を防ぎ，対処策を迅速に講じることができ，特定のタイプの攻撃に対する防御策を開発することができる．アンチウイルスソフトウェアベンダーやセキュリティエンジニアがセキュリティ対策や侵入検知に取り組む際の重要な情報源でありながらセキュリティポリシーを開発し，リスクを管理するのに役立つ．

AV-TEST 研究所 [5] によると，毎日 45 万以上の新しい悪意のあるプログラムや潜在的に望ましくないアプリケーションが登録されており，2022 年 5 月には低パワーの IoT デバイスから巨大なサーバまで 1 億 3,561 万以上のマルウェアが発見されている．この数は 10 年前の 7 倍である．幅広いデバイスが対象になっている中，低パワーデバイスにも脅威は存在する．従って，様々な環境で様々なマルウェアを処理できる適切なモデルが求められている．

マルウェアには，ウイルス，トロイの木馬，ランサムウェア，ルートキット，スパイウェアなど，さまざまな形態がある．世界のサイバー犯罪コストは，2025 年には約 10 兆 5000 億ドルになると予測されている [6]．マルウェアは機能，行為及び権限昇格などにより様々な角度で分類されている．マルウェアを正しく分類することは，マルウェアがコンピュータやデバイスにどのように感染するか，脅威レベルはどの程度かなど，マルウェアから保護する方法を理解するための重要なステップである．限られたリソース，環境の中でマルウェアを適切に分類できるモデルが求められる．

1.1.2　アプローチ１：シグネチャベース

これまでマルウェアの検出と分類には，主にシグネチャベースの手法とヒューリスティックな手法が用いられてきた [7]．シグネチャベースの手法は，特定のマルウェアを一意に識別するアルゴリズムまたはハッシュであり，アンチウィルスベンダーによって数十年にわたって使用されている．シグネチャに基づく方法は，検出のスピード，実行の効率性，幅広いアクセス性に優れている [8]．この方法の利点は，特定のシグネチャを検索することで，高速かつ正確であることである．しかし，ディジタルシグネチャのパターンは攻撃者によって簡単に抽出され，マルウェアのシグネチャを混乱させるために実装される [9]．また，ハッカーは常に，ウイルスシグネチャに一致しないポリモーフィックやメタモーフィックなマルウェアを作成することで，アンチウィルスベンダーに検出されないようにマルウェアを隠そうとしている [10, 11]．ポリモーフィック・コードはポリモーフィック・エンジンを使って，元のアルゴリズムをそのままに変異させる．暗号化はコードを隠す最も一般的な方法である．メタモーフィック・ウィルスは，自分自身のバイナリ・コードを一時的な表現に変換し，自分自身の一時的な表現を編集し，編集された形式を再びマシン・コードに戻す．レジスタの機能を変更したり，

マシン命令を同等のものに変更したり，操作命令を挿入しなかったり，独立した命令を並べ替えたりすることは，マルウェアを変異させる．これらのテクニックを使用し，数十万件ならともかく，毎日数百件の新しいシグネチャが生成される場合，対応するのは困難である [12]．つまり，データベースに依存せず，マルウェアの特徴を捕まえられるような，有効な手法が求められている．

マルウェア分類に使用されるもう一つの手法は，専門家によって決定されたルールに基づくヒューリスティック・ベースの手法である．シグネチャベースの手法と比較して，ヒューリスティックベースの手法は未知のマルウェアを分類できるという利点がある．しかし，ルールを満たせば正常なファイルでもマルウェアと判断してしまうため，誤検知率が高いという欠点がある．ここでは迅速な対応及びマルウェアの特徴を有効的に扱うことが求められる．

1.1.3 アプローチ２：静的・動的解析

従来のマルウェア分類では，マルウェアを解析してその挙動や特徴を理解する必要があった．マルウェアの解析には，静的解析と動的解析という 2 つの方法が一般的に用いられている．静的解析とは，マルウェアを実行せずに悪意のある解析を行うことである [13]．動的解析とは，サンドボックスや仮想マシンのようなシミュレートされた環境で実行される悪意のある解析を意味する [14]．静的解析では，通常，解析前に悪意のあるコードを解凍し，復号する．次に，IDA pro や OllyDbg などの逆アセンブラまたはデバッガツールを使用して，実行可能ファイルをインテル x86 アセンブリ命令のシーケンスを持つアセンブリファイルに逆アセンブルする．これらのアセンブリファイルや，API コール，ファイルのエントロピー，ファイル内の文字列などの情報を分析することで，マルウェアの有効的な特徴を得ることができる．半面，この手法は攻撃者が使用するさまざまなコード難読化技術に対しては対応が困難である [15]．動的解析では，SysAnalyzer，Process Explorer，ProcMon，RegShot，Wireshark，TCPview などのツールを使用して，実行中のマルウェアの動作を監視する [16]．静的解析と比較して，動的解析では実行ファイルを逆アセンブルする必要がない．マルウェアが難読化されていもリアルタイムでマルウェアの行為を観察することができる [17]．しかし，動的解析にはいくつかの制限がある．マルウェアの挙動を監視するためには，各サンプルを安全な環境で一定時間実行する必要がある [18]．監視プロセスには時間がかかり，実行されたマルウェアがプラットフォームに感染しないようにしなければならない [19]．さらに，多くのマルウェアは仮想環境での実行を検知し，対抗することができるため，動的解析手法は機能しない [20]．これらの静的解析と動的解析より得られた特徴を活かす様々な研究がある．その中で，より容易にコストを減少できる方法が機械学習である．

静的解析で取得可能な特徴は API システムコール，レジストリ，ファイル システムなど様々がある．統計的に機械学習は低レベルの特徴を使用し， Decision Trees [21]，kNN[22],

Naïve Bayes, Random Forest, and SVM[23, 24] を用いて高い精度を達成することができるが，低レベルの特徴を用いた統計的な手法だけでは，より巧妙な攻撃を検知するには不十分である [25, 26]．半面，様々な環境で豊富な情報を保つ高レベルの特徴を扱えるには工夫が必要である．

1.1.4　アプローチ３：自然言語処理

これに対し，高レベルの特徴を用いた機械学習の手法である，画像処理（CV）と自然言語処理（NLP）のアプローチが広く応用されている．CV および NLP のどちらも，マルウェアから得られた最も関連性の高い特徴を最大限に活用している．NLP 手法は，API コールシーケンス [27]，Windows レジストリ [28]，文字列など，マルウェアから抽出する際に得られる貴重な情報を活用する．しかし，自然言語処理ベースのアプローチは，パックされたり難読化されたりしているマルウェアの特徴量を設計することの難しさに直面する．最近では，マルウェアの最大 80% がパッカーや圧縮技術によって難読化されている [29]．これらの環境の中で有効的な情報を抽出できるような手法が求められている．

1.1.5　アプローチ４：画像化＋深層学習

マルウェアの作成者は，オリジナルのソースコードの一部を変更し，新しいマルウェアを作り出すことが多い [30]．作成されたマルウェアはバイナリ形式で保存されることが多いが，バイナリイメージを画像として見た場合，その中のさまざまなセクションは，画像内のパターンとして識別することができる．画像は小さな変化を捉えながらも，大局的な構造を保持することができるため，同じファミリに属するマルウェアの亜種同士のバイナリイメージは，画像として非常によく似ているように見える．これらの画像は，他のマルウェア・ファミリの画像とも容易に区別される．この性質を利用することで，マルウェアファミリの分類をバイナリイメージから行うというアプローチが生まれた．Conti et al. [31] はマルウェアのバイナリをグレースケール画像に可視化する方法を提案し，マルウェアのバイナリを視覚的に分析することで，画像から様々なデータ領域を区別できることに気づいた．

人間がマルウェアの特徴を視覚的に分析できる画像ベースの手法は，難読化されたマルウェアファミリにも対応できることが多い．この理由を以下に示す．攻撃者は難読化技術（暗号化，パッキング，メタモーフィズム，ポリモーフィズムなど）を用いてなりすましを行うが，同じファミリに属するマルウェアはコードやデータの順序が類似しており，なりすましたイメージが類似の傾向を持つことがある．マルウェアのパッキングに関しては，以下のような報告がなされている．同じパッカーでパックした後，異なるファミリに属するマルウェア亜種のイメージは異なっている [32]．また，異なるファミリのアンパックされた亜種は，パックされ

た亜種とは全く異なっている [33].

マルウェア画像化の利点は，マルウェアの機能を把握するためにデコンパイルや動的実行環境を使用する必要がなく，また分類のために意図的に特定の統計的特徴を計算せずとも同等の分析ができることである [34]．また，画像化することによって，様々な画像処理のテクニック及び深層学習モデルを使用することができる．

深層学習は，その強力さから各分野におけるゲームチェンジャーとなっている．マルウェア分類における深層学習の利用においては，他のアプローチに比べ歴史が浅く，これからの発展が期待されるため，本研究ではこのアプローチに基づいたマルウェア分類に取り組む．

1.2 本研究が想定する状況

教師あり学習に関する多くの既存研究においては，GPU の利用を前提とした，大規模で複雑なニューラルネットワークモデルの使用がますます一般的になっている．このため，GPU が非力であったり，CPU しか使えないような，ローパワーデバイス環境での使用は困難であるが，本研究ではそのような場所にこそマルウェア対策が必要であると考える．

本研究では，サーバだけでなく様々なクライアントやエッジ環境においても動作可能なシステムを想定している．このような場所で働くシステムは軽量であることが求められる．ここでいう「軽量」とは，分類時に軽量であることはもちろん，将来的にはエッジで新たなマルウェアファミリを補足する場合でも扱えることを目指すため，学習時にも軽量であることを意味している．

GPU が使える環境では，畳み込みネットワークがよく用いられる．他の画像抽出方法（HOC，GIST など）より精度が高いためである．多くの畳み込み層を用いたネットワークは，局所的な特徴の関連性を高めることができるがコストが大きい．一方，少数の畳み込み層で少数のパラメータを用いるとコストは削減できるが，精度が落ちてしまう．つまり，このトレード・オフを解決するために，適切なモデルが必要となる．一方，畳み込みネットワーク (CNN) の性能を向上させるために，カーネル（フィルタサイズ），パディング，ストライド，チャネル数など，多くのハイパーパラメータを最適化する必要があることも CNN の問題である [35]. いくつかの研究 [36, 37, 38, 39] では，最先端の CNN モデルの適用が試みられているが，その性能はまだ十分ではない．

最近の研究では，畳み込みネットワークを使用しない他のニューラルネットワークモデルへの移行も徐々に進んでいる [40, 41]．また，CPU 環境では，多層パーセプトロン（MLP）がよく使われているが，畳み込みと同じく，深層なアーキテクチャほど精度が高くなる．それにつれてコストがかかる半面，畳み込みに比べ精度の差は大きい．単純な多層パーセプトロンをより効果的に扱う MLP-mixer が提案されている．MLP-mixer は，様々な場面で画像の特徴を抽出が可能で，畳み込みネットワークの代表的な Resnet50 モデルと匹敵する結果が得られたが，

MLP-mixer には改善の余地がまだある．

1.3　本研究の目的と成果

本研究の目的は，画像に基づく深層学習により軽量でありながら高精度なマルウェア分類器を作成することである．この目的に対し具体的には以下の 2 つの課題に取り組んだ．

1.3.1　既知のマルウェア分類の性能向上

本研究は，まず限られたリソースを前提とした軽量な教師あり学習モデルのアーキテクチャを対象とする．軽量化によって推論時間を短縮することで，計算コストが削減できる [42]．限られた GPU が使える環境においては，少数の畳み込み層に変分オートエンコーダ及びアテンションメカニズムを加えることによって，マルウェアの特徴をさらに抽出できるモデルを提案する．このモデルは，シンプルなアーキテクチャであり，少数のパラメータで高い分類精度が得られる．CPU のみの環境においては，MLP-mixer よりオートエンコーダを導入することによって得られた画像特徴をより精錬させることで重要な特徴を取り出すことができるモデルを提案する．その結果，少数のパラメータ及び軽量なアーキテクチャで高い分類精度が得られた．

1.3.2　未知のマルウェア分類への対応

上記の提案手法はラベル付きの教師あり学習であるため，未知のマルウェア分類には対応できないが，未知のマルウェアは，毎年数多く発見されている．Sonicwall のレポートによると 2023 年前半だけで，1 日平均 956 個，合計 172,146 個の既知ではないマルウェアを特定した [43]．そのため，未知のマルウェア分類タスクも益々重要になってくる．

教師あり学習では，マルウェア分類の精度が向上しているが，未知のマルウェアを分類することは困難である．N-shot 学習の中でフューショット学習がいくつか研究されていたが，少なくとも一つのサンプルが必要である．サンプルがなくても画像特徴とラベルの関係で活用し，適用できるゼロショット学習があるが，モデル作成及び精度の問題もあり，研究としては少ない．既存研究では複雑な CNN モデルと一つのレイヤーで画像特徴からラベル空間への写像を行うが，過学習の危険性とマッピングの多重化不足が問題となる．

そこで，本研究では写像の妥当性を考慮したモデルによりゼロショット学習のそれぞれ課題を解決できるモデルを提案した．このモデルは，軽量でありながら既存研究より高い精度を得た．

1.4　本論文の構成

本論文は以下の章により構成される.

第 1 章: 序論
本論文における研究の背景と目的を述べた.

第 2 章: 畳み込み層を用いるマルウェア分類：CNN-AVAE の提案
軽量化 CNN を用いたマルウェア分類についての研究結果を紹介する.

第 3 章: 畳み込み層を用いないマルウェア分類：MLP-Mixer-Autoencoder の提案
軽量化 MLP を用いたマルウェア分類についての研究結果を紹介する.

第 4 章: 未知のマルウェア分類：ZSL-SLCNN の提案
軽量化 CNN 及びゼロショット学習を用いた未知のマルウェア分類についての研究結果を紹介する.

第 5 章: 結論と展望
本論文の結論を述べ, 残された課題を示した上で今後の展望について述べる.

第 2 章

畳み込み層を用いるマルウェア分類：CNN-AVAE の提案

この章では，特殊な畳み込みネットワークに基づく軽量なモデルを使用した悪意のあるコードを分類する方法を提案する．多層の畳み込み層を使用する代わりに変分オートエンコーダとアテンションを導入したモデルを考案し，実験によりその有効性を確認した．

高性能コンピューティングデバイスによって巨大な CNN アーキテクチャーモデルが動作可能となり，これまでより複雑なレベルで画像を処理することが可能になった．また，最近の研究 [34] によると，パラメータの少ない単純な CNN 構造でマルウェアを分類したところ，深い層からなる畳み込みネットワークより劣るものの，ある程度の性能を達成できることが分かる．これは，畳み込み層の有用性を示すものであるが，一方で求められる機能に適した層構造を用いることができれば，性能を維持しながら軽量化することができることを示唆している．

畳み込みの特徴と言えば，浅い畳み込み層ではローレベル (線，角度など) なローカル特徴しか抽出することができない，一方で深いニューラルネットワークほどハイレベルな物体のローカル特徴を得られる（顔の画像だと目，耳，鼻など）．このように，深い畳み込みネットワークを用いることによってより良いグローバル特徴を得られるため，様々な分類タスクの性能を向上できる．少数の畳み込み層で性能を向上させるためには，グローバルな特徴を改善する必要がある．高度な画像処理を考えると，(1) グローバル特徴の抽出には多層の畳み込み層が必要であり，(2) 特徴間の関係を抽出するにはフィルタが用いられている．これらが CNN の巨大化に繋がっている．

本研究では，まず上記の (1) に対応するため，オートエンコーダ（AE: AutoEncoder）を用いる．AE は，独自のニューラルネットワーク構造を持つ教師なし深層学習アルゴリズムである [44]．エンコーダとデコーダの間に生成される潜在空間は，入力を再生するために必要十分なグローバル特徴を教師なし学習で得られる．AE は基本的にはエンコーダ，デコーダ及び潜在空間の層のみで構成されるため，軽量なアーキテクチャであり，最小限の再構成誤差で入力

を出力に変換し，小さなデータでも処理できる．しかし，AE はオーバーフィッティングに陥ることが多いため，より良いグローバル特徴を得るためには潜在空間を改善する必要がある．このような手法の一つに，変分オートエンコーダ（VAE）がある．変分オートエンコーダは，学習が正則化されたオートエンコーダとして導入され，オーバーフィッティングを回避し，潜在空間が生成処理を可能にする適切な特性を持つことを保証する．

本研究では，次に (2) に対応するため，アテンションメカニズムを導入する．アテンションメカニズム [45] は，深層学習における重要なブレークスルーとなっており，自然言語処理をはじめ，画像認識，音声認識などに広く使われている．視覚システムにおいて，アテンションメカニズムは，入力の重要度に応じて特徴量を適応的に重み付けすることによって実現される動的選択プロセスとして扱うことができる．アテンションメカニズムは，画像分類，物体検出，セマンティック・セグメンテーション，画像生成，マルチモーダル課題など，非常に多くの視覚的課題にメリットをもたらしてきた．しかし，コンピュータビジョンにおけるアテンションメカニズムに基づいたマルウェア分類に関する研究は限られている．

これまでの研究では，主にニューラルネットワークの深さと幅，特徴量の増加に焦点が当てられてきたが，オブジェクトの特徴量の充実さにはまだ焦点が当てられていない．本章では，“AVAE”と呼ぶアテンションメカニズムによって強化された新しいタイプの変分オートエンコーダを CNN と組み合わせることで，小さなモデルアーキテクチャを維持しながら，可能な限り多くの価値ある特徴を収集することを目的とする．AVAE は，より識別性の高い特徴を提供し，元の特徴空間を潜在表現にマッピングし，精錬することができる．以下のセクションでは，AE，VAE，画像ベースのアテンション，提案手法，実験結果をそれぞれ紹介する．

2.1 関連研究

本節では，本研究に関連する研究と本研究の相違点について述べる．本研究はマルウェアを画像として分類し，軽量でありながら効果的なモデルを目的としている．本研究では大きく分けて畳み込みニューラルネットワーク，オートエンコーダ，注意機構の 3 つの技術を利用した研究との相違点について説明する．

2.1.1 画像に基づくマルウェア分類の誕生

画像処理技術を用いてマルウェアを可視化・分類するための新しいアプローチを初めて提案したのは，Nataraj et al.である [46]．著者らは，同じクラスの画像はレイアウトとテクスチャが非常に類似しているという観察に基づいて，マルウェアをグレースケール画像として可視化した．著者らは，特徴抽出器として画像のウェーブレット分解に基づく GIST 記述子を，分類器として k-nearest neighbor(kNN) を利用した．この論文は，導入した Malimg データセットで 97.18% の精度を達成した．HOG や HOC+GIST といった他の特徴記述子も適用されているが，この方法は計算コストが高いため，大量のマルウェアの処理には適していない．Naeem et al.[47] は，局所的特徴ベクトルとグローバル特徴ベクトルを組み合わせてバランスをとることで，新しいタイプの特徴記述子を利用した．その結果，Malimg データセットにおいて 98 % という高い分類率を達成した．しかしながら，この段階では，CNN はまだ使われておらず、高い分類精度まで至らなかった。

2.1.2 CNN を用いたマルウェア分類

現在の研究は，深い畳み込みネットワークを用いた複雑なネットワークモデルの構築に注力している．例えば，10 層以上の畳み込み層:VGG16[48]，VGG19[49]，複数の CNN アーキテクチャの組み合わせなどである [50]．一方，Çayır et al.[51] はパラメータを最小化することで学習を高速化する．提案モデルは，比較セクションにおいて，最良モデルの学習可能なパラメータ数を 99.7% 削減することで，最先端モデルと比べ，約 1% 以下の精度を達成した．

Rezende et al.[52] は，ImageNet 上の ResNet-50 の最初の 49 層をマルウェア分類タスクに移植した．凍結層は学習された特徴抽出層と見なすことができる．著者らは最後の層を，Malimg データセットのクラス数に応じて 25 個の全結合ソフトマックスと 1000 個の全結合ソフトマックスに置き換えた．750 エポックの後，この論文は 10 分割クロスバリデーションで平均 98.62% の精度に達した．また，同じ kNN 分類器を用いて，Deep CNN (DCNN) から抽出された特徴量と GIST の特徴量を比較した結果，ResNet-50 は 98.00%，GIST は 97.48% と，0.52% の差で GIST を上回った．

Vasan et al.[53] は CNN のアンサンブルを利用した．彼らは，異なる CNN が画像の異なる意味表現を提供すると仮定しており，そのため従来の手法よりも高品質な特徴が抽出される．ImageNet[54] で事前に訓練された VGG16 と ResNet-50 がマルウェア画像用に微調整された．このアンサンブル手法は，低い誤検出率で高い検出精度を達成した．

Anandhi et al.[55] は Densely connected networks（DensNet）を用いた別のタイプの DeepCNN を紹介した．DensNet は密なブロック，合成関数，遷移層から構成される [56]．このアーキテクチャは，深いネットワークを通じて勾配が縮小するため，消失勾配問題を解決した．著者は，201 層の深さを持つ DenseNet201 を利用し，Malimg データセットで 98.97%，類似ファミリの C2LOP と Swizzor を組み合わせて 99.36% の精度を達成した．

Çayır et al.[51] は複雑な CNN アーキテクチャの代わりに，Random CapsNet フォレストエンジニアリングと呼ばれるシンプルなアーキテクチャを提案した．このモデルはオートエンコーダに似たカプセルを含み，各カプセルは与えられたクラスのインスタンスを表現する方法を学習する．提案手法は，データ増強，データ再サンプリング，転移学習，重み付き損失関数を用いないが，それでも 98.72% の精度で許容できる結果を達成した．

Nisa et al.[57] は，事前に訓練された AlexNet[58] と Inception-V3[59] から抽出された特徴を組み合わせている．これらの融合特徴量は，SVM，kNN，決定木（DT）などの異なる分類器を用いて分類される．彼らは，Malimg データセットで 98.7% の精度を達成した．この結果は，Malimg をバランスの取れたデータセットにするために補強を適用すると，99.3% まで改善された．

これらの既存の研究では深い CNN を用いることによってより良いグローバル特徴を得られるが，そのためにはより多くの計算能力を必要とする．一方，Abijah Roseline et al.[34] は，軽量 CNN モデルを提案したが Nataraj et al.[46] と比べ、精度は 0.31% のみしか改善しない. つまり，浅い畳み込み層でパラメータが少ないだけでは，必ずしも対象物の特徴を十分に抽出しているとは言えない．

本研究では軽量でありながらより良い精度を実現するためにはグローバル特徴抽出できるようにローカルな特徴に加え、変分オートエンコーダ（VAE）を用いてグローバル特徴を整理して補足させる．

2.1.3　オートエンコーダを用いたマルウェア分類

Lee and Lee[40] は，複数の AE を適用することで，オートエンコーダの有効性を示している．各 AE モデルは，1 種類のマルウェアのみを分類し，対応するファミリのサンプルのみを使用してトレーニングされる．その結果，同じ AE ネットワーク構造のシステムで 94.03%，様々な AE を適用したシステムで 97.75 %の精度を達成した．さらに，類似クラスを組み合わせることで，97.75% から 98.21% に 0.46% 改善した．しかし，依然として誤分類が多く，AE

が画像ベースのマルウェアの特徴抽出に有効でないことが示された．AE は再現ロスを最小限しながら学習を行い、潜在空間に重要な情報を残すが，潜在空間内の配置的な関連性を考慮していない．VAE は AE より潜在空間の特徴配置を良くするために提案された．Burks et al.[60] は，ただ Resnet を結合するため，精度は改善されていない．本研究では VAE の特徴の品質向上のため、注意機構（Attention）を適用する．

2.1.4 注意機構を用いたマルウェア分類

Awan et al.[49] は，動的空間畳み込みと呼ばれる空間畳み込みアテンションを VGG19 ネットワーク [61] に適用した．このアテンションは，グローバル平均プーリング（GAP）メカニズムを利用し，ラムダ層によって GAP の出力を再スケールし，完全連結層の前に 0.25 の割合でドロップアウトに供給し，著者は CNN の伝統的な分類器としてソフトマックスを利用した．性能は Malimg データセットで評価され，97.68% の精度を達成した．

Ma et al.[62] は，アテンションメカニズム [45] を応用し，メカニズムを適用し，5 つの部分からなる手作りのアーキテクチャを採用した： 入力層，ローカルアテンション層，グローバルアテンション層，デンス層，出力層である．他の手法と比較して，注意機構と CNN 機構の組み合わせは，Microsoft's Kaggle データセットで 96.09% という最高の分類精度を達成した．

これらの研究は深い畳み込み層においてアテンションメカニズムを導入した結果，若干の精度向上が見られたが，よりモデルを複雑にしてしまう．そこで，本研究ではシンプルな注意機構 [63] を VAE のエンコーダを導入することによって，マルウェアの浅い特徴に焦点を当てる．

2.2　マルウェア画像化

マルウェアのバイナリファイルを図 2.1 に示す．図中の各行を見ると最初はメモリアドレスのオフセットで次には 16 進数のペアである．各 16 進数の組は，画像のピクセル値となる 1 つの数値として扱われる．得られた配列は 2 次元配列として構成されなければならず，値は [0,255] の範囲にある（0：黒，255：白）．図 2.2 に示すようにそれぞれのペアを変換すると最終的なマルウェアの画像が得られる．図 2.3 は，Dontovo.A マルウェアを画像化した例である．

```
00000000  4D 5A 90 00 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 00 00 00 00
00000018  40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000030  00 00 00 00 00 00 00 00 00 00 00 00 E8 00 00 00 0E 1F BA 0E 00 B4 09 CD
00000048  21 B8 01 4C CD 21 54 68 69 73 20 70 72 6F 67 72 61 6D 20 63 61 6E 6E 6F
00000060  74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65 2E 0D 0D 0A
00000078  24 00 00 00 00 00 00 00 DA ED 02 7E 9E 8C 6C 2D 9E 8C 6C 2D 9E 8C 6C 2D
00000090  64 AF 75 2D 9C 8C 6C 2D 8D 84 31 2D 9C 8C 6C 2D 9E 8C 6C 2D 9B 8C 6C 2D
000000A8  1D 84 31 2D 93 8C 6C 2D 9E 8C 6D 2D CB 8C 6C 2D 9B 80 0C 2D 9C 8C 6C 2D
000000C0  9B 80 36 2D 9F 8C 6C 2D 52 69 63 68 9E 8C 6C 2D 00 00 00 00 00 00 00 00
000000D8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 03 00
000000F0  47 3D DA 4D 00 00 00 00 00 00 00 00 E0 00 0F 01 0B 01 07 0A 00 0C 00 00
00000108  00 18 00 00 00 00 00 00 71 16 00 00 00 10 00 00 00 20 00 00 00 00 A0 2A
00000120  00 10 00 00 00 02 00 00 04 00 00 00 00 00 00 00 04 00 00 00 00 00 00 00
00000138  00 50 00 00 00 04 00 00 00 00 00 00 02 00 00 00 00 00 10 00 00 10 00 00
00000150  00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 00 00 00 00 00 00 00 00 00
00000168  94 18 00 00 78 00 00 00 00 40 00 00 A8 03 00 00 00 00 00 00 00 00 00 00
00000180  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000198  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 10 00 00 7C 00 00 00
000001C8  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000001E0  2E 74 65 78 74 00 00 00 98 0B 00 00 00 10 00 00 00 0C 00 00 00 04 00 00
```

図 2.1: バイナリファイルの構造

深層学習が他の機械学習技術よりも優れている点の一つは，手作業やドメイン固有の特徴工学を必要とせずに，バイナリデータに直接適用できることである．これは，マルウェアのシグネチャを特定・抽出するための専門知識や時間のかかるプロセスを必要とすることなく，マルウェアを効率的に分類できる能力であり，研究の重要な動機となっている．

バイト列は通常，円形バッファを保持するために使用される基本的なバイト配列を隠すための精錬されたインターフェースである．このバイト配列は，テキストファイルや画像を表すこともあり，誰が読み取るかは文脈によって異なる．そのため，マルウェアのサンプルを画像として可視化するには，すべてのバイトを画像の 1 ピクセルとして解釈する必要がある．バイナリファイルは，図 2.1 のマルウェアの Portable Executable（PE）の 16 進数表現である．

モデルのトレーニングとテストの計算を効率的に並列化するために，深層学習アプローチでは，各ファイルが標準的なサイズである必要がある [64]．同じサイズであることに加え，標準的なハードウェアを使用してモデル学習プロセスを実用的な時間で行うためには，本深層学習

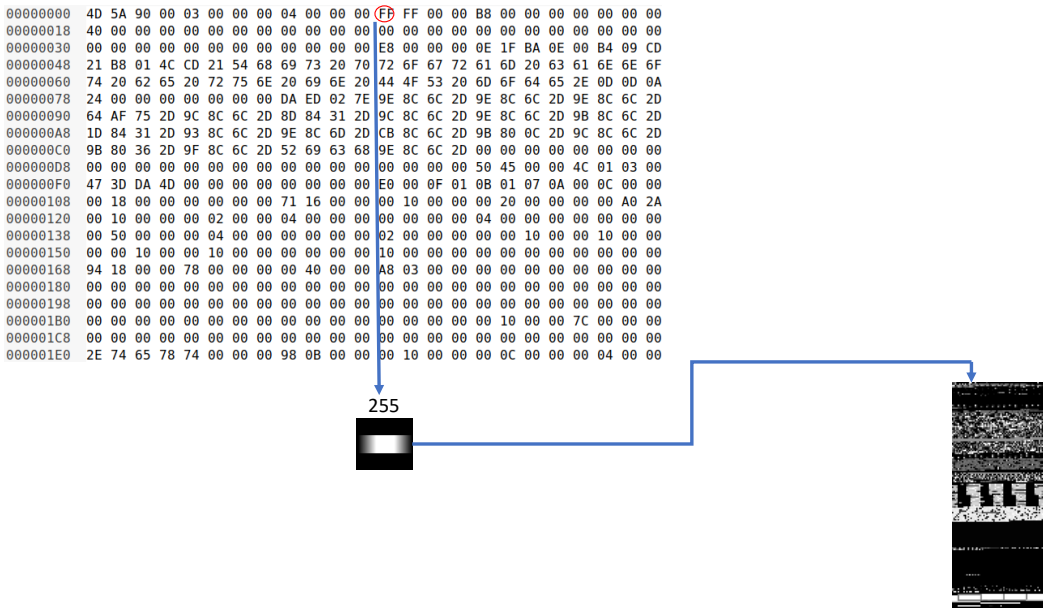


図 2.2: バイナリファイルからの画像化

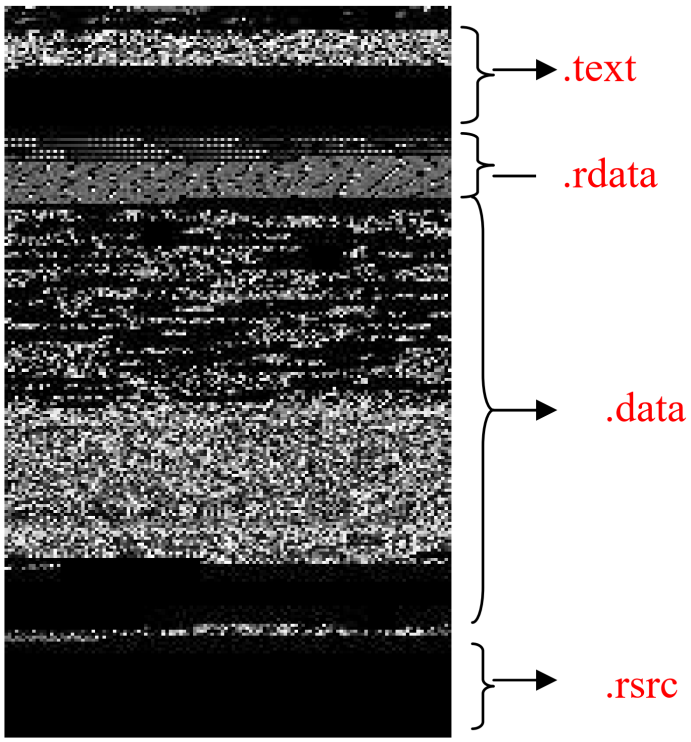


図 2.3: Dontovo.A マルウェアの画像化 [46]

手法は，計算量の観点からこのサイズを制約する必要がある．

パディングや切り捨てなど，ファイルサイズを標準化するために取り得るオプションは数多

くあるが，本研究は，マルウェアのファイルデータ内の共通のパターンや構造を識別・検出するために深層学習モデルを設計している．Nataraj et al.[46] で実装されたのと同じ手順に従って，汎用的な画像スケーリングアルゴリズムを使用した．画像のサイズはバイナリファイルのサイズに依存する．表 2.1 は，画像の幅をファイルのサイズによって固定し，画像の高さはファイルサイズによって異なる．また，マルウェアをグレースケール画像に変換するのに長時間を要しないことも示している．1Mb 未満の一般的な悪意のあるコードであれば，変換にかかる時間は 0.001 秒以下である．

表 2.1: マルウェアのファイルサイズに応じた画像の幅

ファイルサイズ	画像の幅	時間変換 (ms)
<10 kB	32	0.105
10kB-30kB	64	0.312
30kB-60kB	128	0.428
60kB-100kB	256	0.571
100kB-200kB	384	0.748
200kB-500kB	512	0.665
500kB-1Mb	768	0.814
>1Mb	1024	2.85

バイナリのさまざまなセクションは，画像として見ることができる．**“.text”**セクションには実行コードが含まれている．図 2.3 から，**“.text”**セクションの最初の部分には，テクスチャが細かいコードが含まれていることがわかる．残りはゼロ（黒）で塗りつぶされており，このセクションの最後にゼロのパディングがあることを示している．続いて **“.data”**セクションには，初期化されていないコード（黒いパッチ）と初期化されたデータ（細かい粒状のテクスチャ）の両方が含まれている．最後のセクションは **“.rsrc”**セクションで，モジュールのすべてのリソースが含まれている．これらには，アプリケーションが使用するアイコンが含まれることもある．

図 2.4 は，Nataraj et al.[46] が作成した Malimg データセットからのマルウェアプロットの一部である．特定のファミリの画像は類似しているものの，別のファミリの画像とは異なっていることが観察できる．

新しいバリアントは，多くの場合，コードのほんの一部を変更することで作られるため，画像化された結果は非常によく似ている [65]．さらに，マルウェアを画像に変換することで，同じファミリに属するサンプルの包括的な構造を維持したまま，小さな変更を検出することが可能になる [66]．図 2.5 からわかるように，ファイルサイズにばらつきはあるものの，全体的な構造は画像から確認できる．

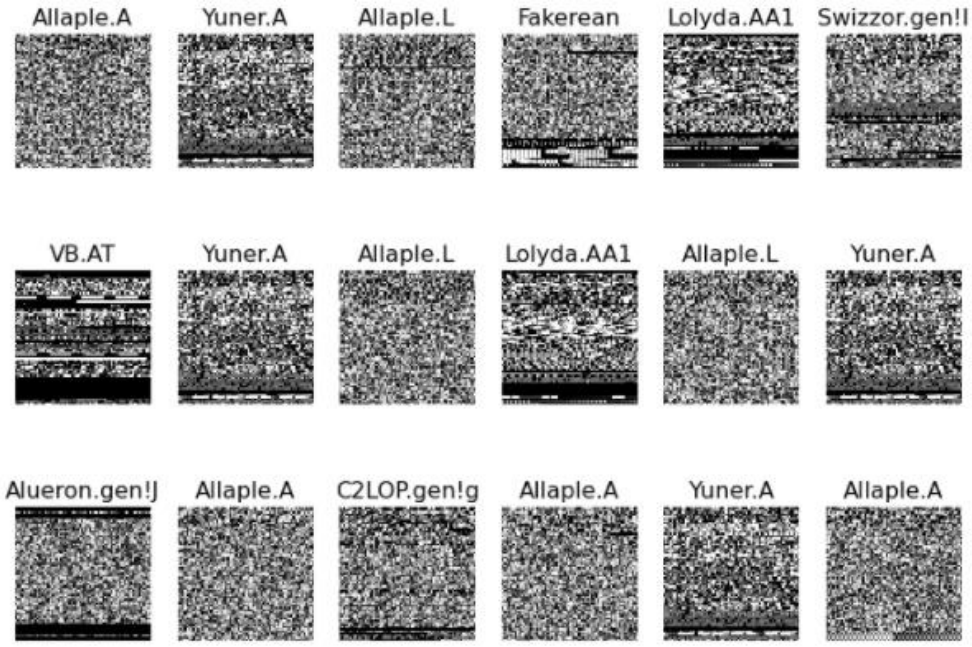
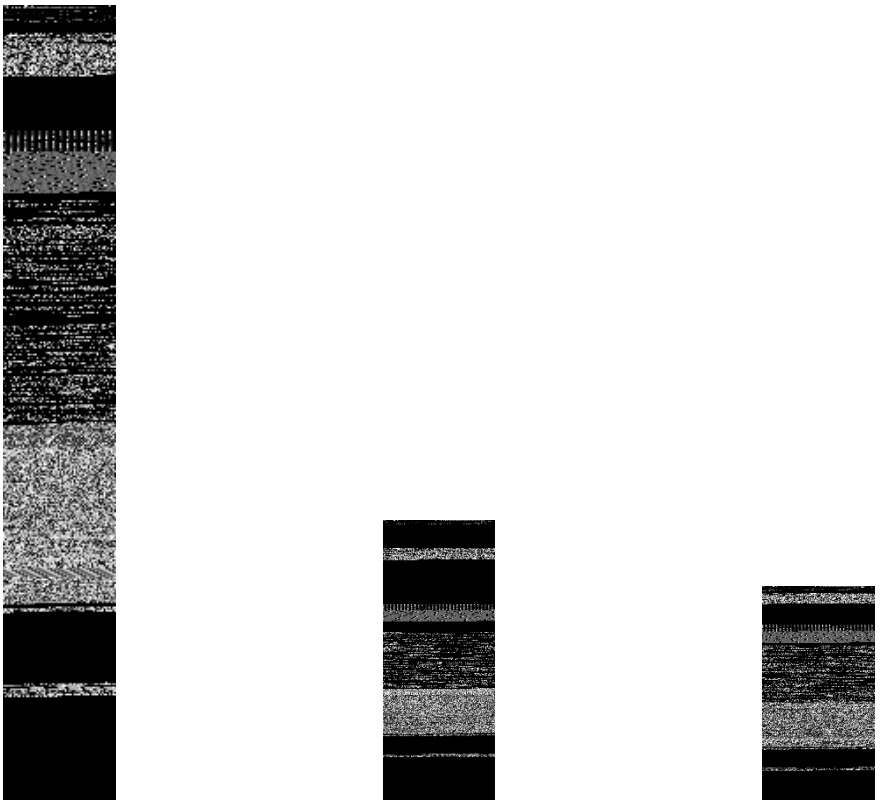


図 2.4: Malimg データセットからのサンプル

図 2.6 及び図 2.7 はそれぞれのパック方法によるイメージの違いを表している．アンパックマルウェアファミリが同じパックタイプでパックされた場合，新しくパックされたマルウェア（同じファミリ）のイメージも類似している傾向があるため，画像手法を用いてパックされたマルウェアとパックされていないマルウェアの間の相関関係を分析することが可能であると考えられる．



(a) Dontovo.A マルウェアのサンプル 1　(b) Dontovo.A マルウェアのサンプル 2　(c) Dontovo.A マルウェアのサンプル 3

図 2.5: Dontovo.A 様々なマルウェアのサンプル

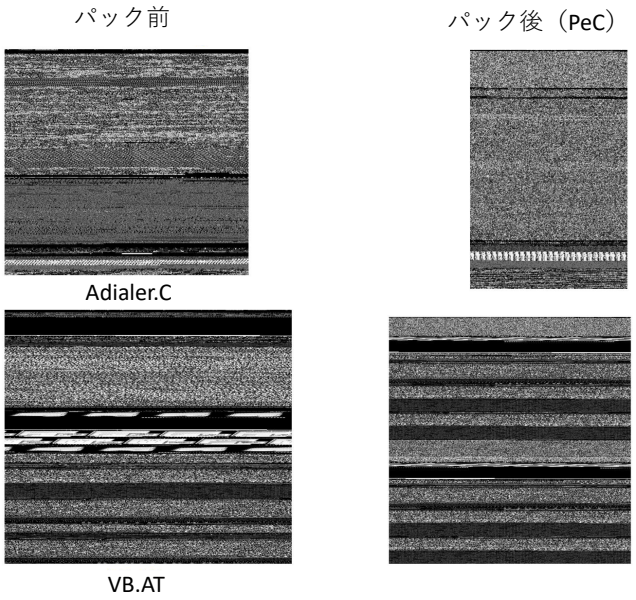


図 2.6: PeC パッカーよる画像の影響

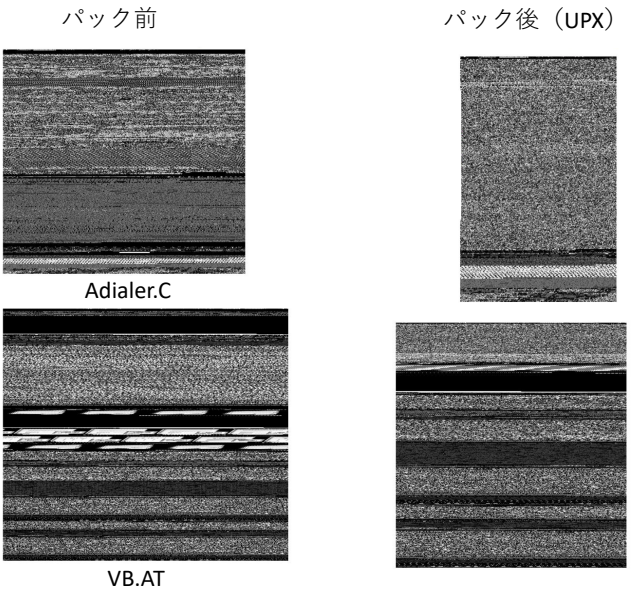


図 2.7: UPX パッカーによる画像の影響

2.3　オートエンコーダ

機械学習における次元削減とは，あるデータを記述する特徴の数を減らすプロセスのことである．この次元削減は，選択または抽出のいずれかによって行われ，低次元データを必要とする多くの状況（データの可視化，データストレージ，重い計算など）で有用である．次元削減には様々な手法があるが，これらの手法のほとんどにマッチするグローバルなフレームワークを設定することがでる．まず，「古い特徴」表現から「新しい特徴」表現を生成するプロセスを（選択または抽出によって）エンコーダと呼び，逆のプロセスをデコーダと呼ぶことにする．次元削減は，エンコーダが（初期空間から潜在空間へ）データを圧縮し，デコーダがそれらを復元する．初期データの分布，潜在空間の次元，エンコーダの定義によっては，この圧縮は損失発生になる可能性がある．つまり，情報の一部が符号化処理中に失われ，復号化時に回復できなくなる．

次元削減法の主な目的は，与えられたファミリの中から最良のエンコーダ／デコーダのペアを見つけることである．言い換えれば，与えられたエンコーダとデコーダの可能な集合に対して，エンコード時に最大の情報を保持し，デコード時に最小の再構成誤差を持つペアを探すことである．エンコーダとデコーダをニューラルネットワークとして設定し，最適化プロセスを繰り返しながら最適な符号化・復号化方式を学習する．つまり，各反復において，オートエンコーダ・アーキテクチャにデータを与え，エンコード-デコードされた出力を初期データと比較し，その誤差を誤差逆伝播法してネットワークの重みを更新する．したがって，直感的には，全体的なオートエンコーダアーキテクチャは，情報の主要な構造化された部分のみが通過し，再構成されることを潜在空間を作り出す．

図 2.8 に示すように一般的なフレームワークを見ると，エンコーダのファミリはエンコーダネットワークアーキテクチャによって定義され，デコーダのファミリはデコーダネットワークアーキテクチャによって定義され，再構成誤差を最小化するエンコーダとデコーダの探索は，これらのネットワークのパラメータに対する勾配降下によって行われることによってより潜在表現 $\mathbf{z}$ を生成することができる．

エンコーダとデコーダの重み行列とバイアスベクトルをそれぞれ $\mathbf{W}_e, \mathbf{b}_e, \mathbf{W}_d, \mathbf{b}_d$ とする．$\mathbf{x} = \left\{x^1, x^2, ..., x^n\right\}$ を訓練データセットとする．$\varphi = (\mathbf{W}_e, \mathbf{b}_e), \theta = (\mathbf{W}_d, \mathbf{b}_d)$ をそれぞれエンコーダとデコーダの学習用パラメータセットとする．q_φ はエンコーダとし，z^i は入力サンプル x^i の潜在変数となる，エンコーダは入力 x^i を潜在変数 z^i にマッピングする．デコーダ p_θ は，潜在変数 z^i から入力 $\mathbf{x}$ を復元するニューラルネットワークである．

$$z^i = q_\varphi(x^i) = a_e(\mathbf{W}_e x^i + \mathbf{b}_e) \tag{2.1}$$

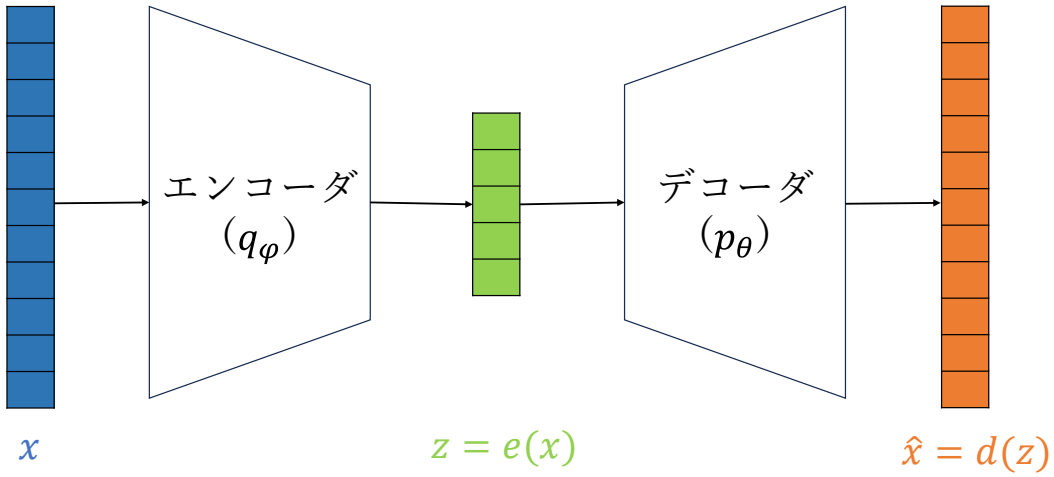


図 2.8: オートエンコーダの構造

$$\widehat{x^i} = p_\theta(z^i) = a_d(\mathbf{W}_d z^i + \mathbf{b}_d) \tag{2.2}$$

ここで，a_e と a_d はそれぞれエンコーダとデコーダの活性化関数である．$\widehat{x^i}$ はオートエンコーダの出力になる．x^i の単一サンプルに対して，オートエンコーダの損失関数は $\widehat{x^i}$ と x^i の差である．データセットに対するオートエンコーダの損失関数は，下式のようにデータサンプル全体の平均 2 乗誤差（MSE）として計算されることが多い [67]：

$$l_{AE}(\mathbf{x}, \theta, \varphi) = \frac{1}{n}\sum_{i=0}^{n}(x^i - \widehat{x^i})^2 \tag{2.3}$$

2.4　変分オートエンコーダ（VAE）

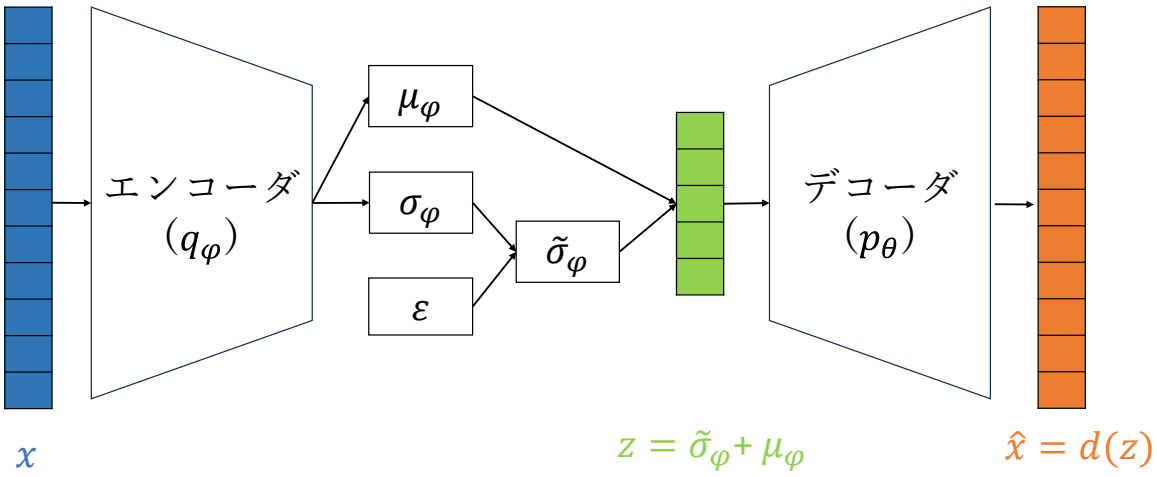


図 2.9: 変分オートエンコーダの構造

VAE はオートエンコーダの一種で，エンコーダとデコーダから構成される [68]．オートエンコーダは，潜在空間がどのように構成されていても，できるだけ損失（再構成損失）を少なく符号化・復号化するようにのみ学習される．したがって，エンコーダが潜在空間を位置的な関連性に整理することを保証するのは難しい．ここで似たようものを近くに配置するため，潜在空間は正規分布として扱うことを考えられる．

まず，入力は潜在空間上の分布として符号化される．次に，潜在空間からの点がその分布からサンプリングされる．サンプリングされた点がデコードされ，再構成誤差が計算される．最後に，再構成誤差はネットワークを通じて逆伝播される．

実際には，符号化される分布は正規分布として選択され，エンコーダはこれらのガウシアンを記述する平均と共分散行列を返すように学習することができる．入力が一点ではなく，ある分散を持つ分布としてエンコードされる．エンコーダによって返される分布は，標準正規分布に近くなるように強制される．したがって，VAE を学習する際に最小化される損失関数は，符号化-復号化方式の性能を可能な限り向上させる傾向のある「再構成項」（最終層）と，エンコーダによって返される分布を標準正規分布に近づけることによって潜在空間の構成を正則化する傾向で構成される．

この正則化項は，生成された分布と標準ガウス分布との間の KL ダイバージェンス [69] として表され，生成プロセスを可能にするために潜在空間から期待される規則性は，2 つの主要な特性によって表現することができる：連続性（潜在空間内の 2 つの近接した点は，一度デコードされると 2 つの全く異なる内容を与えるべきではない）と完全性（選択された分布に対して，潜在空間からサンプリングされた点は，一度デコードされると「意味のある」内容を与えるべきである）．

VAE が入力を単純な点ではなく分布として符号化するという事実だけでは，連続性と完全性を保証するには不十分である．うまく定義された正則化項がないと，モデルは再構成誤差を最小化するために，分布が返されるという事実を「無視」して，ほとんど古典的なオートエンコーダのように振る舞う（オーバーフィッティングにつながる）ことになる．すると，エンコーダは，小さな分散を持つ分布を返すか，非常に異なる平均を持つ分布を返すかのどちらかを行うことになってしまう．どちらの場合も，分布は間違った方法で使われ，連続性や完全性は満たされない．そこで，これらの影響を避けるために，エンコーダから返される分布の共分散行列と平均の両方を正則化する必要がある．実際には，この正則化は，分布が標準正規分布に近くなるように強制することによってエンコードされた分布が互いに離れすぎないようにする．

この正則化項により，モデルが潜在空間内で離れたデータをエンコードするのを防ぎ，可能な限り返された分布が「重なる」ように促し，期待される連続性と完全性の条件を満す．当然ながら，正則化項と同様に，これは訓練データにおけるより高い再構成誤差という代償という代償を払うことになる，再構成誤差と KL ダイバージェンスのトレードオフは調整必要である．その結果，VAE の損失関数は以下に挙げられる：

$$l_{VAE}(x^i, \theta, \varphi) = -\mathbf{E}_{q_\varphi(\mathbf{z}|x^i)}\left[log\ p_\theta(x^i \mid \mathbf{z})\right] + D_{KL}(q_\varphi(\mathbf{z}|x^i \parallel p(\mathbf{z})) \tag{2.4}$$

第一項は i 番目のデータ点の期待負対数尤度である．この項は VAE の再構成誤差とも呼ばれ，入力データの再構成をデコーダに学習させるためである．モンテカルロ法によってこの $\mathbf{E}_{q_\varphi(\mathbf{z}|x^i)}\left[log\ p_\theta(x^i \mid \mathbf{z})\right]$ の勾配を計算する場合，$q_\varphi(\mathbf{z} \mid \mathbf{x}) \sim \mathcal{N}(\mathbf{z}; \mu(\varphi), \sigma(\varphi))$ に従う乱数を振ってしまうと，あとからパラメータ φ によって微分しようとしても微分できないという問題があった．これを解析するため，Reparameterization trick と呼ばれる決定的な式変形を提案された [70]．図 2.9 のように潜在変数 $\mathbf{z}$ は平均 μ と標準偏差 σ の正規分布に従うと仮定しているので，潜在変数は以下の式，

$$\mathbf{z} = \mu(\varphi) + \sigma(\varphi) \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1) \tag{2.5}$$

で書くことができる．これにより変分パラメータ φ で微分することが可能になる．従って，再構成誤差は以下の式で書き換えることができる．

$$\mathbf{E}_{q_\varphi(\mathbf{z}|\mathbf{x})}\left[log\ p_\theta(\mathbf{x} \mid \mathbf{z})\right] \simeq \frac{1}{K}\sum_{k=1}^{K} log\ p_\theta(\mathbf{x} \mid \mathbf{z}) \tag{2.6}$$

上記の式 2.6 を用いて，VAE の損失関数の式 2.4 は以下のようになる．

$$l_{VAE}(x^i, \theta, \varphi) = -\frac{1}{K}\sum_{k=1}^{K} log\ p_\theta(x^i \mid z^{i,k}) + D_{KL}(q_\varphi(\mathbf{z}|x^i \parallel p(\mathbf{z})) \tag{2.7}$$

ここでは $z^{i,k} = g_\varphi(\epsilon^{i,k}, x^i)$, g は決定的関数, ノイズ ϵ^k が $\mathcal{N}(0,1)$ を示す. 第二項は，エンコーダの分布 $q_\varphi(\mathbf{z}|\mathbf{x})$ と期待分布 $p(z)$ の間の KL ダイバージェンスである．このダイバージェンスは q と p の関係を測るものである [70].

2.5　注意機構 (アテンション・メカニズム)

アテンション・メカニズム（Attention Mechanisms）は人間の知覚において重要な役割を果たすことはよく知られている [71, 72]．この手法は，機械翻訳のエンコーダ・デコーダ・モデルの性能を向上させるために導入された．アテンション・メカニズムの背後にある考え方は，デコーダが入力シーケンスの最も関連性の高い部分を柔軟に利用できるようにすることである．アテンション・メカニズムは自然言語処理（NLP）の領域で最初に一般的に紹介されたのは，Vaswani et al.[45] が提供したものである．アテンション・メカニズムによって NLP の分野では，リカレント・ニューラル・ネットワーク（RNN）をアテンション・ベースのネットワークに置き換える動きがある．この論文では，三つの主要コンポーネントを用いて注目メカニズムを計算した：クエリー，キー，バリューである．これらの主要を使用して文内の単語間の相関関係を見つけ出す．

自然言語処理のアイデアを活かして Zhang et al.[73] によって初めてコンピューター ビジョンにも適用された Self-Attention GAN モジュール（SAGAN）を図 2.10 に示す．アテンション・メカニズムを利用することによって，ネットワークは顕著な領域に注意を向け，画像合成を成功させるために必要な大域的属性のいくつかを完成させるように学習することで，画像の大域的構造をよりよく捉えることができる．

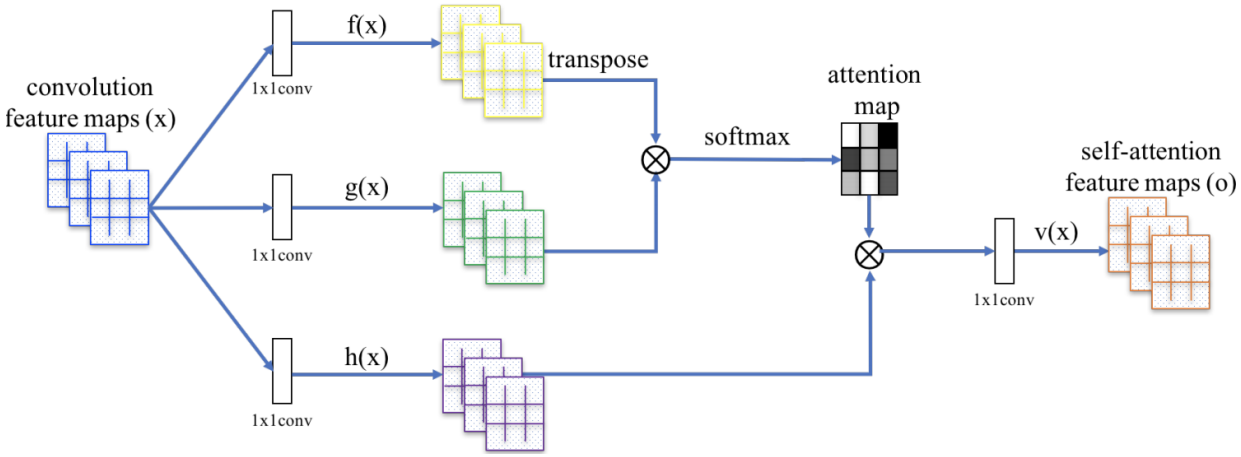


図 2.10: コンピュータビジョンのために提案された自己注意のメカニズム [73]

モジュールでは，前の隠れ層 $\mathbf{x} \in \mathbb{R}^{CxN}$ からの画像特徴が二つの特徴空間に変換される，f と g, で注目度を計算する．それぞれ行列 $\mathbf{W}_f$ 及び $\mathbf{W}_g$ で定義される特徴空間は，入力特徴マップ

に 1 × 1 畳み込みを適用することで得られる．

特徴空間 f と g を用いて注目スコア $\beta_{j,i}$ を計算する．注目スコアは，特定の部分を合成する際に，画像の異なる領域に必要とされる相対的な強調の程度を決定する．$\beta_{j,i}$ は次式で計算される：

$$\beta_{j,i} = \frac{\exp\left(s_{ij}\right)}{\sum_{i=1}^{N} \exp\left(s_{ij}\right)} \tag{2.8}$$

アテンション層の出力は $\mathbf{o} = \left(o_1, o_2, \ldots, o_j, \ldots, o_N\right) \in \mathbb{R}^{CxN}$ である．ここで，o_j は画像の j 番目の領域に対する出力であり，次のように得られる：

$$o_j = v\left(\sum_{i=1}^{N} \beta_{j,i} h\left(x_i\right)\right) \tag{2.9}$$

$v\left(x_i\right)$ の値は，画像の各部分に与えられる重要度の重みを意味し，入力特徴マップの位置 x_i に学習された行列 $\mathbf{W}_v$ を乗算することで得られる．同様に，$h\left(x_i\right)$ は各ロケーションの重要度を測定し，獲得行列 $\mathbf{W}_h$ との行列乗算によって計算される．

SAGAN モジュールの最終出力は，注意層の出力にパラメータ γ を乗算し，それを入力特徴マップに足し戻すことで得られる．この定式化は次式で与えられる．

$$y_i = \gamma o_i + x_i \tag{2.10}$$

導入されたパラメーター γ によって，ネットワークはまず局所的な近傍の手がかりに依存するようになり，その後徐々に非局所的な証拠により多くの重みを割り当てるように学習する．

コンピュータビジョンにおけるアテンションメカニズムの異なる形態として，畳み込みブロック注意モジュール (CBAM) が知られている [63]．CBAM は，特に画像分類と物体検出タスクにおいて，モジュールの幅広い応用可能性を示すことに成功した最初のものである．CBAM は，潜在的な多層的注意 (空間アテンションとチャンネルアテンションの組み合わせ [74]) を利用する．ここで，空間アテンションの「空間」は，各特徴マップに内包される領域空間を指す．空間アテンションは特徴マップまたはテンソルの単一断面スライス上のアテンション・メカニズムを表す．特徴マップを精錬することで，後続の畳み込み層への入力を強化し，モデルの性能を向上させることができる．

一方，チャンネルは基本的にテンソルに積み重ねられた特徴マップであり，各断面スライスは基本的に $(H \times W)$ 次元の特徴マップである．水平および垂直エッジの学習に特化したものもあれば，より一般的で画像内の特定のテクスチャを学習するものもある．チャンネル・アテンションは，基本的に各チャンネルに重みを与え，その結果，学習に最も貢献する特定のチャンネルを強化し，モデル全体の性能を向上させる．チャネルアテンションは，どの特徴マップ

が学習に重要であるかを絞り込み，強化する．一方，空間アテンションは，特徴マップ内の何が学習に不可欠であるかを伝える．両者を組み合わせることで，特徴マップをロバストに強化することができ，その結果，モデル性能の大幅な向上が得られる [63].

2.5.1　チャネル空間における注意機構 (CAM)

チャンネル・アテンション・モジュール（CAM）は機構では特徴マップの意味があるものに焦点を当てる働きがある．CAM は，チャネルの冗長性を減らし，特徴のチャネル間の関係を捉えることによってチャネル・アテンション・マップを構築することに重点を置いたアテンション・メカニズムである [63].

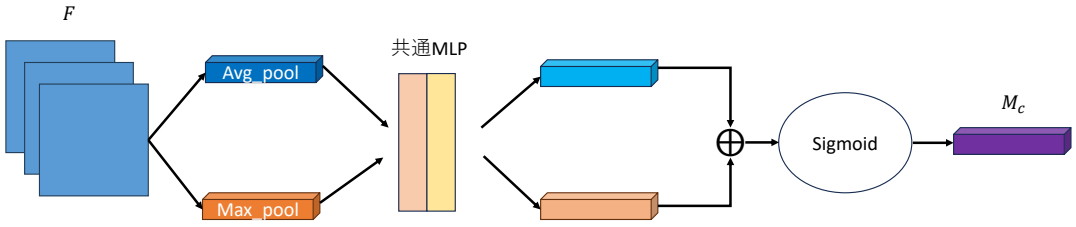


図 2.11: チャンネルアテンション (CAM)

図 2.11 より，特徴マップの中間層 **F** が与えられ，特徴を絞り込んで集約するために，平均プーリングと最大プーリングが同時に実行され，二つの異なる特徴マップが生成される：最大値プーリングされた特徴 $\mathbf{F}^{c}_{max}$ と平均値プーリングされた特徴 $\mathbf{F}^{c}_{avg}$.

そして，それぞれの $\mathbf{F}^{c}_{avg}$ 及び $\mathbf{F}^{c}_{max}$ は，共有ネットワークの小さな多層パーセプトロン（MLP）に転送される．MLP は一つの隠れ層であり，パラメータのオーバーヘッドを減らすため，隠れ活性化サイズは $\mathbb{R}^{1\times1\times C/r}$ に設定されている [75]，r は縮小率である．縮小率が高いほど，ボトルネックのニューロン数は少なくなり，逆もまた然りである．そして，その非線形性のために ReLU 活性化関数が選択される．得られたそれぞれの特徴 $\mathbb{R}^{1\times1\times c}$ を合計した結果のベクトルはシグモイド活性化層に渡され，チャンネル重みが生成される．要約すると，チャネルのアテンションは次のように計算される．

$$\mathbf{M_c} = \sigma(MLP(Avg\_Pool(\mathbf{F})) + MLP(Max\_Pool(\mathbf{F}))) \tag{2.11}$$

ここで，σ はシグモイド関数を示す．

CAM は，スクイーズ・エキサイテーション・レイヤー [75] とよく似ているが，若干の変更が加えられている．グローバル平均値プーリング (GAP) によって特徴マップを 1 ピクセルに縮小する代わりに，入力特徴を二つの次元 $(1 \times 1 \times c)$ のベクトルに分解する．これらのベクトルの一方は GAP によって生成され，もう一方はグローバル最大値プーリング（GMP）によって生成される．平均プーリングは主に空間情報を集約するために使用されるが，最大プーリングは画像内のオブジェクトのエッジという形で，より豊かな文脈情報を保存するため，より細かいチャンネルに注意を向けることができる．平均プーリングはスムージング効果があり，最大プーリングはよりシャープな効果があるが，オブジェクトの自然なエッジをより正確に保存する．

2.5.2　画面空間における注意機構 (SAM)

スペーシャル・アテンション・モジュール (SAM) は，物体がどの位置にあるかについて焦点を当てる働きがある．画面空間における注意機構の概要を図 2.12 に示す．

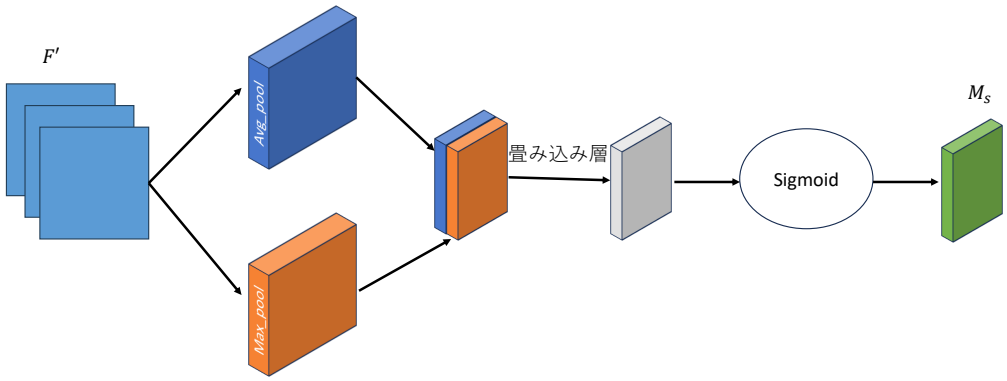


図 2.12: 空間アテンション (SAM)

空間アテンション・モジュール（SAM）は，3 重の逐次操作で構成される．最初の部分はチャネルプールと呼ばれ，入力次元 $(H \times W \times C)$ を二つのチャネルに分解する $(H \times W \times 2)$．それぞれはチャネル間の最大プーリング $\mathbf{F^s_{max}}$ と平均プーリング $\mathbf{F^s_{avg}}$ を表す．二つの特徴を合

わせて，フィルタサイズ 7x7 の畳み込み層を通して，出力の次元は $(H \times W \times 1)$ を得る．この出力は次にシグモイド活性化層に渡される．シグモイドは確率的活性化で，すべての値を 0 から 1 の間の範囲にマッピングする．この空間アテンションマスクは，次に単純な要素ごとの積を使用して，入力のすべての特徴マップに適用される．要約すると，空間のアテンションは次のように計算される．

$$\mathbf{M_s} = \sigma(f^{7\times7}(Avg\_Pool(\mathbf{F'}); Max\_Pool(\mathbf{F'}))) \tag{2.12}$$

ここで，σ はシグモイド関数を示す，$f^{7\times7}$ はフィルタサイズ 7×7 の畳み込み層である．

2.5.3　画像処理における注意機構 (CBAM)

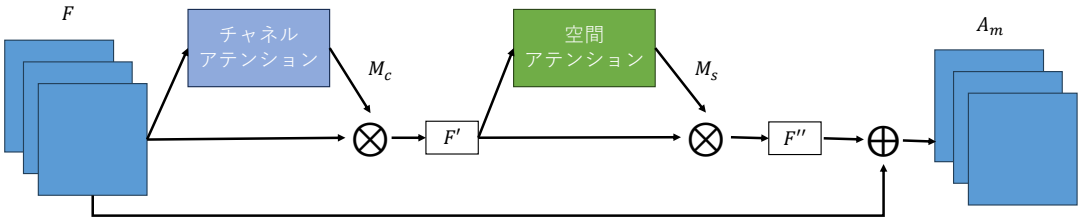


図 2.13: CBAM の概要図

中間特徴マップ $\mathbf{F} \in (H \times W \times C)$ が入力として与えられると，CBAM は図 2.13 に示すように，1 次元チャネルアテンションのマップ $\mathbf{M_c} \in (1 \times 1 \times C)$ と 2 次元空間注意マップ $\mathbf{M_s} \in (H \times W \times 1)$ を順次推論する．全体的な注意のプロセスは次のように要約できる：

$$\mathbf{F'} = \mathbf{M_c}(\mathbf{F}) \otimes (\mathbf{F}) \tag{2.13}$$

$$\mathbf{F''} = \mathbf{M_s}(\mathbf{F'}) \otimes (\mathbf{F'}) \tag{2.14}$$

$$\mathbf{A_m} = \mathbf{F} \oplus \mathbf{F''} \tag{2.15}$$

⊗ はアダマール積を表す．空間アテンションで得られた特徴マップ **F”** は元の特徴マップ **F** を足す最終的特徴マップ $\mathbf{A_m}$ が得られた．この手法はスキップ接続 (skip connection) と呼ばれる．ディープニューラルネットワークにおいて，途中の複数層を N 層分スキップして先の層へとつなげる迂回パスにより，離れた層間で順伝搬・逆伝搬を行えるようにする機構である．勾配消失や勾配爆発を防ぐための効果的手法である [76].

2.6　提案手法 CNN-AVAE

これらを踏まえ，本研究ではオートエンコーダとアンションをもつ畳み込みニューラルネットワークモデルである CNN-AVAE を提案する．

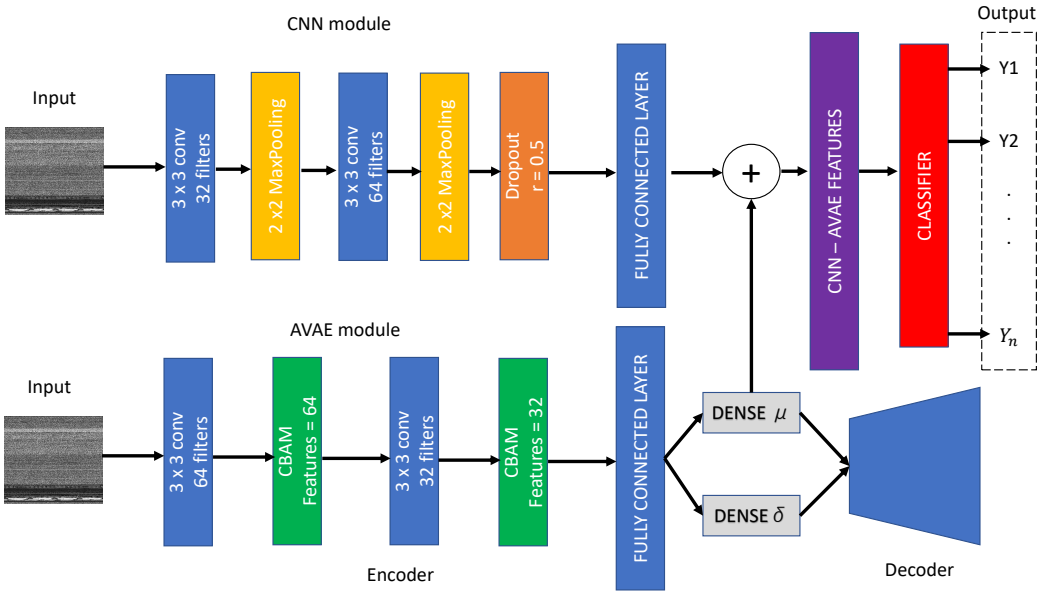


図 2.14: CNN-AVAE のトレーニングフェーズ

図 2.14 は提案モデルのトレーニング時におけるシステムアーキテクチャを示している．上部での CNN モジュールは入力画像に対しするそれぞれの畳み込み層，（カーネルサイズは 32，次いで 64）及びプーリング層（フィルタサイズ 2 × 2）を通して，浅い畳み込み層ではローレベル (線，角度など) ローカル特徴を抽出する．プールされた特徴マップを平坦化する（全結合層）前に，オーバーフィッティングを避けるために，0.5 の割合でドロップアウトを適用する．さらに，最小学習率=0.001 の微調整オプティマイザとして Adam[77] を用いる．

下部の AVAE モジュールでは，畳み込み変分オートエンコーダーのエンコーダの畳み込み層の間に順番に CBAM を挿入する．デコーダを通して，学習を行い，潜在空間を生成する．学習後の平均 μ を用いる．スキップ接続からインスピレーション [76] を得て，上部の抽出された特徴量を足し合わせして総合的特徴 CNN-AVAE FEATURES を得る．図 2.14 右上の CLASSIFIER は，この総合的特徴を入力として，典型的な分類アルゴリズムを使用することで入力された画像の分類を行う．CNN モジュールも AVAE モジュールも，64 × 64 サイズの低解像度画像を学習し，エポック数は 50 である．

本研究は 5 エポック後に改善されずに学習を終了する早期終了を利用する．早期終了とは，モデルが訓練データを過剰に学習し汎化性能が低下する過学習の状態に陥る前に学習を早期に

終了させる手法である．機械学習では，早期終了は，勾配降下法などの反復法で学習を訓練する際に，オーバーフィッティングを避けるために使用される正則化の一形態である [78].

図 2.15 はテスト時のアーキテクチャを示す．テストでは，学習済の重みを用いて，入力サンプルはそれぞれの CNN モジュールで得られた特徴量と AVAE モジュールのエンコーダの後に潜在空間の平均の特徴量を組み合わせ，総合的特徴 CNN-AVAE を用いて，評価を行う．

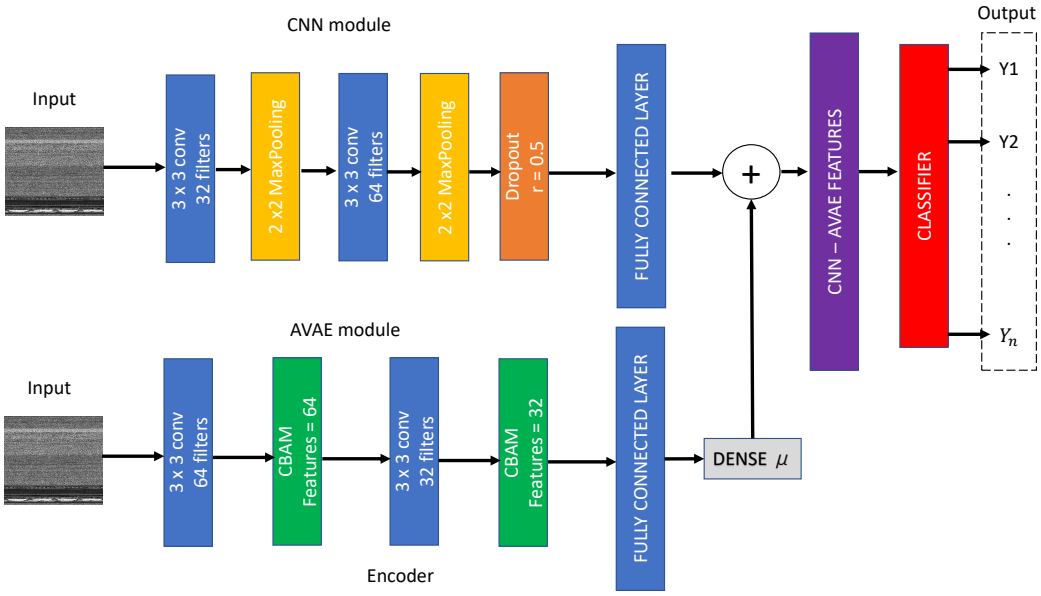


図 2.15: CNN-AVAE のテストフェーズ

提案手法を評価するために，10 分割クロスバリデーションを利用する．10 個のサブサンプルのうち 1 個を検証データとして持ち出し，残りの 9 個のサブサンプルを学習データとして使用する．このプロセスを 10 回繰り返し，10 個のサブサンプルをそれぞれ検証データとして使用する．10 回の結果の平均が手法の性能となる．

表 2.2: Malimg データセットの詳細

クラス	ファミリ名	サンプル数	比率 (%)
0	Adialer.C	122	1.31
1	Agent.FYI	116	2.12
2	Allaple.A	2949	31.58
3	Appaple.L	1591	17.04
4	Alueron.gen!J	198	2.12
5	Autorun.K	106	1.14
6	C2LOP.gen!g	200	2.14
7	C2LOP.P	146	1.56
8	Dialplatform.B	177	1.89
9	Dontovo.A	162	1.73
10	Fakerean	381	4.08
11	Instantaccess	431	4.62
12	Lolyda.AA1	213	2.28
13	Lolyda.AA2	184	1.97
14	Lolyda.AA3	123	1.32
15	Lolyda.AT	159	1.70
16	Malex.gen!J	136	1.46
17	Obfuscator.AD	142	1.52
18	Rbot!gen	158	1.69
19	Skintrim.N	80	0.86
20	Swizzor.gen!E	128	1.37
21	Swizzor.gen!I	132	1.41
22	VB.AT	408	4.58
23	Wintrim.BX	97	1.04
24	Yuner.A	800	8.57

2.7 実験結果

2.7.1 データセット

提案モデルを三つのマルウェアデータセットで評価した： Malimg[46]，Microsoft's BIG 2015[79]，MaleVis[80]．表 2.2, 表 2.3, 表 2.4 は，異なるデータセットにおける異なるファミ

表 2.3: BIG2015 データセットの詳細

クラス	ファミリ名	サンプル数	比率 (%)
0	Ramnit	1541	14.18
1	Lollipop	2478	22.80
2	Kelohos_ver3	2942	27.07
3	Vundo	475	4.37
4	Simda	42	0.39
5	Tracur	751	6.91
6	Kelihos_ver1	398	3.66
7	Obfuscator.ACY	1228	11.3
8	Gatak	1013	9.32

リのサンプルを示している．

Malimg データセットには，グレースケール画像として提示された 9339 個のマルウェアサンプルが存在する．データセットの各マルウェアサンプルは，25 のマルウェアファミリの一つに対応している．Microsoft データセットには，9 つのマルウェアファミリに由来する 10,868 個のラベル付きサンプルがある．

Microsoft データセットのマルウェアサンプルはアンパックされている．マルウェアを表 2.1 に示すルールに従って画像に変換する際には，各マルウェアのバイナリファイルのみが使用される．

MaleVis データセットは，26 のファミリ（25 のマルウェア＋ 1 のクリーンウェア）のいずれかに割り当てられた 9100 個の画像から構成される．

2.7.2　評価指標

本研究の提案手法を評価するため，既存の研究コミュニティで広く使われている 4 つの標準的な性能指標を適用した：正解率，適合率，再現率，F 値 (F-measure)．この 4 つの指標は，表 2.5 の 4 つのパラメータを援用して説明されており，評価対象のクラスは正，残りのクラスは負である．

正解率は，正しく分類されたサンプル数とテストデータセット全体の比率として定義される．

$$\text{正解率} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.16}$$

表 2.4: Malevis データセットの詳細

クラス	ファミリ名	サンプル数
0	Adposhel	350
1	Agent	350
2	Allaple	350
3	Androm	350
4	Autorun	350
5	Autorun.K	350
6	BrowseFox	350
7	Dinwod	350
8	Elex	350
9	Expiro	350
10	Fasong	350
11	HackKMS	350
12	Hlux	350
13	Injector	350
14	InstallCore	350
15	MultiPlug	350
16	Neoreklami	350
17	Neshta	350
18	Other	350
19	Regrun	350
20	Sality	350
21	Snarasite	350
22	Stantinko	350
23	VBA	350
24	VBKrypt	350
25	Visel	350

適合率は，正に分類されたサンプルの TP の比率として定義される．

$$\text{適合率} = \frac{TP}{TP + FP} \tag{2.17}$$

表 2.5: パフォーマンス測定パラメータ

パラメータ	説明
True Positive (TP)	正しく分類された正のクラスサンプルの数
True Negative (TN)	負のクラスは正しく負のクラスに分類される
False Positive (FP)	正クラスに誤分類された負クラスのサンプル数
False Negative (FN)	正のクラスが負のクラスに誤分類されたサンプルの数

再現率は，TP のうちに正に分類されたサンプルの比率と定義する．

$$\text{再現率} = \frac{TP}{TP + FN} \tag{2.18}$$

F **値** *(F-measure)* は，モデルの性能を評価するために使用される．

$$F\text{ 値} = 2 \times \frac{\text{適合率} \times \text{再現率}}{\text{適合率} + \text{再現率}} \tag{2.19}$$

2.7.3　CLASSIFIER の違いによる分類性能の違い

提案手法においては，CNN-AVE により得られた特徴 (CNN-AVE FEATURES) を，何らかの分類器 (CLASSIFIER) への入力とすることで分類を行う．本研究は三つのデータセットに対していくつかの標準的な分類器を利用して，分類性能の比較を行った．

その結果を表 2.6 に示す．提案モデルの CLASSIFIER 部分に Random Forest を使用した場合に最も高い精度を得られた．また，この時には他の既存研究で提案されたモデルよりも高い精度を得られた [81, 82]．既存研究との精度比較は表 2.7 を示す．Wong et al.[50] は Malimg データセット及び Malevis データセットとも圧倒的に適合率が高い割には再現率は低い．再現率は，実際に発見するマルウェアのうち，どれだけを正しく分類できるかを示す指標である．再現率が低いと，実際のマルウェアの一部を見落としていることになる．その結果，総合的 F は低下してしまう．Wong et al.[50] は Malimg データセットにおいて，0.25% の精度の方が高いが，再現率は 2.06% 下回る．その結果，提案手法の方が 2.21% 上回った．Malevis データセットにおいても，Wong et al.[50] は 0.1% の精度の方が高いが，再現率は 4.11% 下回る．その結果，提案手法の方が 4.78% 上回った．

また，表 2.7 から，モデルの畳み込み層の数と学習に関わるパラメータ数を見ると提案手法は他のモデルと比べ，大幅にシンプルで軽量であることが分かった．軽量化 CNN の Abijah Roseline et al.[34] と比べ，匹敵するものである．軽量化 CNN の Abijah Roseline et al.[34] は少数のパラメータ (0.83M) で学習したが正解率は 97.49% のみである．初めて Malimg データセットを紹介した Nataraj et al.[46] より 0.31% のみしか上がらなかった [83].

このことは，少数のパラメータを用いるだけでは，必ずしも対象物の特徴を十分に抽出できていないことを証明している．一方，ResNet-50[84] や VGG19[49] などの膨大なパラメータを持つモデルを用いると，若干の精度の向上が見られたが，計算量が多くなる．それにもかかわらず，軽量なアーキテクチャ及び十分な数のパラメータを使用することで，提案手法は大幅に向上させ，計算コストを節約できる．さらに，各悪意のあるコードを分類する時間は平均 0.01 秒しかかからない．

既存研究 [36, 81, 84, 85] のような複雑なアーキテクチャは，高い画質と計算処理能力を必要とする．複雑なネットワークを使用する理由は，深い層は人間に関係する画像処理タスクにおいて，耳や目のような特定の特徴を抽出することが期待されるからである．一方，浅い層は物体のエッジなど画像全体の特徴に注目する．例えば，図 2.4 では，マルウェアのサンプルの単純なグレースケールを観察することで，多くの複雑でない要素を見つけることができる．そのため，64×64 の小さな画像サイズで十分な特徴を抽出し，なおかつ高い精度を確保するために，最初のレイヤーに着目する．Malimg データセットには，暗号化やパッキングなどの難読化技術によって処理されたサンプルが多数含まれている．その中でも，Adialer.C，Autorun.K，Lolyda.AT，Malex.gen!J，VB.AT，Yuner.A に属するマルウェア サンプルは，同じパッカー（UPX）でパッキングされているため，類似した構造とパターンを持っている．その結果，分析者はしばしばこれらを区別することが困難となる．しかし，我々の手法では，これらのサンプルを解凍することなく，高い精度で直接処理することができる．実験の結果，我々の手法がこれらの難読化攻撃に対して頑健であることが示された．

さらに，高い分類精度を達成したにもかかわらず，多くの研究が二つのファミリ変異型の分類で障害に遭遇している： Swizzor.gen!E と Swizzor.gen!I は非常に類似しており，区別が難しい．両ファミリの精度を他の著者と比較した結果を表 2.8 に示す．提案手法はそれぞれ 87.5%，87.9% の精度で最高の性能を達成している．

表 2.6: 三つのマルウェアデータセットにおける CNN-AVAE モデルの分類器部分を変えた場合の性能比較

モデル	Malimg データセット				BIG2015 データセット				Malevis データセット			
	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
Decision Tree	98.12	98.02	97.59	97.80	98.76	98.43	98.16	98.29	98.06	97.99	97.39	97.69
k-Nearest Neighbors	99.30	99.21	99.03	99.12	99.05	98.78	98.52	98.65	96.12	95.92	95.79	95.85
Naive Bayes	98.16	97.65	96.89	97.27	97.15	97.02	96.75	96.88	87.18	87.11	86.78	86.95
Nearest Centroid	98.82	97.73	96.52	97.12	96.79	96.66	96.32	96.49	88.65	88.52	88.58	88.58
Random Forest	**99.40**	**99.25**	**99.12**	**99.18**	**99.21**	**99.15**	**98.89**	**99.02**	**99.32**	**99.65**	**98.94**	**99.29**
SVM	98.23	97.24	96.91	97.07	98.43	98.36	98.10	98.23	98.27	98.20	97.94	98.07

表 2.7: 三つのマルウェアデータセットに対する様々な CNN アーキテクチャの性能比較

モデル	畳み込み層数	パラメータ数	Malimg データセット				BIG2015 データセット				Malevis データセット			
	(個)	(M)	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
軽量化 CNN [34]	**3**	**0.83**	97.49	97.00	97.00	97.00	95.67	96.00	96.00	96.00	-	-	-	-
VGG16 [81]	13	134.36	97.44	97.54	97.42	97.48	88.61	88.72	88.61	88.66	96.18	96.44	95.76	96.10
VGG19 [85]	16	139.67	95.76	97.84	97.76	97.69	-	-	-	-	93.70	93.16	93.06	93.42
VGG19 + Spatial Attention [49]	16	139.67	97.62	97.68	97.50	97.20	-	-	-	-	-	-	-	-
Inception-v3 [81]	48	21.82	97.65	98.70	98.64	98.67	93.29	93.36	993.28	93.32	95.32	96.58	94.99	95.33
Resnet-18 + VAE [60]	16	11.20	85.00	83.00	83.00	83.00	-		-	-	-	-	-	-
Resnet-50 [84]	49	23.59	99.23	98.30	97.88	98.08	98.41	98.33	93.00	94.78	-	-	-	-
Xception [36]	36	20.86	98.03	97.96	98.03	97.99	96.78	96.80	96.73	96.76	97.49	97.57	97.38	97.47
DensNet201 [55]	101	20.00	98.97	-	-	98.88	98.52	-	-	98.53	-	-	-	-
ShuffleNet + DenseNet201 [50]	149	20.20	98.87	**99.50**	97.06	96.97	-	-	-	-	93.91	**99.75**	94.83	94.51
提案手法	8	3.62	**99.40**	99.25	**99.12**	**99.18**	**99.21**	**99.15**	**98.89**	**98.55**	**99.32**	99.65	**98.94**	**99.29**

表 2.8: 最も誤分類の多かったファミリの比較

既存研究	正解率 (%)	
	Swizzor.gen!E	**Swizzor.gen!I**
Yajamanam et al. [86]	51.0	36.0
Naeem et al. [47]	30.0	50.0
Roseline et al. [34]	70.0	45.0
Çayır et al. [51]	56.3	68.8
Verma et al. [87]	**87.5**	81.8
Awan et al. [49]	48.0	56.0
V. Anandhi et al. [55]	84.2	52.5
提案手法	**87.5**	**87.9**

2.8　第 2 章のまとめ

本章では限られた GPU の環境でも活用できる軽量なネットワークに基づくモデルを提案し，代表的なデータセットに対し，巨大な畳み込みモデルより高い精度でマルウェアの分類ができた．複雑かつ巨大なモデルを使用しなくても軽量なモデルでも画像ベースマルウェアを高い精度で分類できたことは，マルウェア分類タスクに対する期待が持てる新たな方法性を示唆できたと言える．

本章は 8 節からなる．2.1 節では，畳み込みネットワークに基づくマルウェア分類の既存研究を紹介し，巨大なアーキテクチャが数多く提案されている一方で，軽量なモデルは少ないことが分かった．2.2 節ではマルウェアを画像に変換する仕組みを紹介し，マルウェアの特徴は画像でも可視化することができるにもかかわらず，変換時間は僅かであることから既存のシステムに適用可能であることを示した．2.3 節及び 2.4 節では，変分オートエンコーダ及びアテンションメカニズムについて説明し，それを 2.5 節の提案手法へ活かした．提案手法はシンプルかつ畳み込み層を数枚しか用いない軽量なアーキテクチャである．詳細な図を用いてトレーニングとテストフェーズをそれぞれ説明した．次に複数のデータセットに対し標準的な評価方法を用いた計算機実験を行い，提案手法の有効性を示した．他のアーキテクチャと比較した結果，提案手法は十分軽量なモデルであり既存研究より良い精度が得られたことが分かった．例えば精度がほぼ同等の ResNet50 に比べパラメータ数を 6 分の 1 以下にすることができた．

実験結果によれば，提案手法はマルウェアファミリを効率的に分類できることが示された．提案手法は，Random Forest を分類器として用いる時に，三つのデータセットにおいて最高の性能を達成している．

モデルの組み合わせには様々なパターンが考えられる，組み合わせによっては必ずしも改善するばかりではない．それぞれのブランチの強みを活かして，工夫していく必要がある．画像に基づくタスクにおいてもローカル特徴のみ，またグローバル特徴のみに着目することでは精度が改善されない．より良いグローバル特徴を生成するためには，特徴数を増やすだけでなく品質を改善する必要がある．本章において，グローバル特徴をより絞り込む必要があることが分かった．

そこで，本研究は AVAE を導入した．VAE の潜在空間では，AE よりもグローバルな特徴がより計画的に組織されているが，要素の重要性は考慮されていない．この研究では，VAE においてより重要な特徴を獲得するために，アテンションメカニズムを特徴選択や重要な部分を強調に利用している．同時に，特徴の多様性を確保するために，また勾配消失や勾配爆発を防ぐため軽量な CNN を組み合わせて低レンジの特徴を捉えることができた．

悪意のあるコードによって生成された画像データと比較して，ImageNet データのような顔画像や動物画像など，深い CNN ネットワークを必要とする複雑な要因はそれほど多くない．

これらの二つのモデルからの補完的な方法は，対象物の豊かで異なる特性を獲得するのに役立っている．また人間の目で区別ができないほどの類似するマルウェアファミリも高い精度で分類できる．これらの結果から，既存のシステムに適用可能であると考えている．

畳み込みネットワークは少なくとも少数の GPU 上に実装されているが，CPU のみのデバイスなどでマルウェア分類するモデルを求められることもあるため，次の章ではそれに対応したモデルを提案する．

第 3 章

畳み込み層を用いないマルウェア分類：MLP-Mixer-Autoencoder の提案

第 2 章では，少数もしくは非力な GPU しか使用できない環境のためのモデルを提案したが，この章では更に厳しい環境である，CPU しか使用できない環境を対象とする．ここでは，低パワーの IoT デバイスまた組み込みデバイスのマルウェアセキュリティのためのモデルを想定している．そのため畳み込み層を使わないマルウェアを分類する新しい方法を提案する．

本章では，畳み込み層及びアテンション・メカニズムを使用しない代わりに，軽量で新たな多層パーセプトロンモデル（MLP-mixer）とオートエンコーダを用いる．MLP-mixer を用いたマルウェア分類手法は提案されていなかったが，事前の調査によって単独では性能が十分ではないことが分かった．そこで，性能向上のため，MLP-mixer にオートエンコーダを組み入れて IoT デバイスでも高精度なマルウェア分類ができることを示す．

2016 年 10 月，Dyn（米国の大手 DNS サービスプロバイダー）は，マルウェアファミリ「Mirai」による近年史上最大かつ最も強力な DDoS 攻撃を受けた．このマルウェアは 120 万台以上の IoT デバイスに感染した [88]．2020 年までに，全サイバー攻撃の 25% が IoT 機器を標的にすると推定されている [89]. 世界の IoT 市場は，ここ数年で大きな成長を遂げている．対策のため GPU を利用したいが多くの IoT デバイスでは GPU を持たない．そのため，畳み込みネットワークを使わないモデルを求められる．そのようなニューラルネットワークモデルで，現在有望視されているものの一つはオートエンコーダであり，もう一つは MLP-mixer である．オートエンコーダに関してはセクション 2.3 で紹介したため，以下では MLP-mixer について説明する．

MLP-mixer は，従来の MLP を改良したものであり，コンピュータビジョン分野で再び注目されている．Tolstikhin et al.[90] は，完全に MLP に基づいたシンプルなアーキテクチャの MLP-mixer を紹介した．

MLP-mixer は入力画像を各パッチに分割し，それぞれパッチ間及びチャンネルの情報共有

することによって画像の部分的関連性を高めるため，通常の MLP より性能が改善された．とはいえ，多くの入力情報の中には，必要でない特徴も含まれているため，より重要な特徴を取り出す手法が求められる．

本章でも特徴抽出の性能向上のために軽量なオートエンコーダを用いる．オートエンコーダーは，独自のニューラルネットワーク構造を持つ教師なし深層学習アルゴリズムである．次元削減，分類のための前処理，入力データの本質的な要素のみの識別など，いくつかの用途がある．オートエンコーダは畳み込み層を使わないため，IoT デバイスでも扱える．

本章では，マルウェアサンプルのより豊かでより選択的な特徴を得るために，AE と連結することで MLP-mixer を強化することを提案する．具体的には，MLP-mixer のグローバル平均プーリング層（GAP）は，モデル内のパラメータ総数を減らし，パッチ間の各チャンネルのローカル空間をすべて集約することにより，オーバーフィッティングを最小化するために使用される．オートエンコーダは潜在的な次元空間を作成し，情報の主要な構造化部分のみを確実に再構成できるようにする．本提案手法では，MLP-Mixer で処理した後，オートエンコーダが各サンプルの特徴空間を精緻化する役割を果たす．

本章の主な貢献は，MLP-mixer と AE からの特徴合成を通じて，画像に基づく軽量なマルウェア分類システムを提供することである．画像に依存した処理を行うだけであるため，マルウェアの挙動を判定するためにマルウェアや環境に関する深い知識を必要としない．AE を用いることで MLP-mixer 自体の中間層のノードを 4 倍以上減少することができる．その結果，モデルの軽量化が実現できる．

3.1 関連研究

本セクションでは，畳み込みネットワークを使用しない画像ベースのマルウェア分類に関する様々な新しい研究について調査する．

前章で述べたように，マルウェアの分類に畳み込みニューラルネットワークを適用したいくつかの研究は，Malimg や Malheur などの標準的なデータセットで高い精度を達成している．提案されたモデルの性能を向上させるために，事前に訓練されたモデルを利用するものもある [36, 50]. ハイパフォーマンス・コンピューティングの発展は，巨大な畳み込みアーキテクチャと相まって，より複雑なレベルでの画像処理を可能にした．しかし，最近の研究によると，単純なネットワーク構造でより少ないパラメータが比較的満足のいく結果をもたらし，IoT[33, 84, 91] やスマートフォン [92] のような薄型のデバイスにも適用できることが示されている．一方，CNN の性能を向上させるために，カーネル（フィルタサイズ），パディング，ストライド，チャネル数など，多くのハイパーパラメータを最適化することも CNN の問題である [35]．いくつかの研究 [36, 37, 38, 39] では，最先端の CNN モデルの適用が試みられているが，その性能はまだ十分ではないため，最近の研究では，畳み込みネットワークを使用しない他のニューラルネットワークモデルへの移行も徐々に進んでいる [40, 41].

Barros et al.[85] は，マルウェア識別のために設計された新しい類似性尺度に基づき，データペアからペア情報を取得する新しい手法を開発した．著者らは，新しい画像をトレーニングデータセットの全ての画像とペアリングすることにより，シャムネットワークの入力を生成する．この方法は，Garbor[93]，Histogram of Oriented Gradients(HOG)[94]，Generic Invariant Shape-Templates(GIST)[95] といった従来の画像特徴抽出を上回る性能を持つが，SoTa 畳み込みネットワークモデルよりは劣る．

Naeem et al.[33] は，SIFT[96] 及び GIST お組み合わせして Combined SIFT-GIST Malware（CSGM）を用いてマルウェアベース画像の特徴抽出する，計算時間を短縮して分類精度を向上させた．CSGM 特徴量はマルウェア画像の局所的特徴量と大域的特徴量から構成されており，従来の手法や画像の局所的特徴量を抽出する単純な CNN モデルよりも情報量が多い．Malimg データセットと Malheur データセットの両方で，それぞれ 98.40% と 97.50% の精度で高い性能を達成した．

Son et al.[97] は，マルウェア画像の主なテクスチャは縦方向に広がっているため，計算の複雑さや学習時間をあまりかけずに，マルウェア画像の横方向のサイズを小さくできることを発見した．正規化された入力画像の次元を減少させる著者らの提案手法は，GIST ベースのマルウェア分類モデルよりも高く，計算複雑度を低減し，学習時間を大幅に節約する．著者らは，SVM 分類器を用いて，Malimg と Malheur データセットにおいて，それぞれ 98.49% と 95.79% の最高精度を達成した．

Vu et al.[98] は, バイナリファイルのバイトをエンコードして画像に配置する新しいアプローチを提案している．著者らは意味情報を考慮し，それをカラー画像として表現する．生成される画像のサイズを 256 × 1024，8 ビット RGB カラーに制限し，分類器として XGBoost[99] を使用する．その結果，提案されたカラー符号化方法であるグレースケールで，精度が 84.20% から 86.36% にそれぞれ 2.16% 向上した．

Narayanan et al.[100] は，あるファミリに属する悪意のあるプログラムは，それぞれ異なるパターンを持っていると宣言している．著者らは，線形次元削減は計算時間を節約し，いくつかの貴重な情報を失うというトレードオフさえ可能であるとして，主成分分析（PCA）[101] を使用する．その結果，得られた性能はまだ CNN に遠く及ばない．

これらの既存研究は従来の MLP を用いて，あらゆるデバイスに広く使用することが可能であるが，高い性能を達成することはできない．本研究ではオートエンコーダを用いて MLP-mixer を改善することで，モデルを複雑にすることなく軽量かつ従来の CNN モデルより良い精度を実現する．

3.2 MLP-Mixer

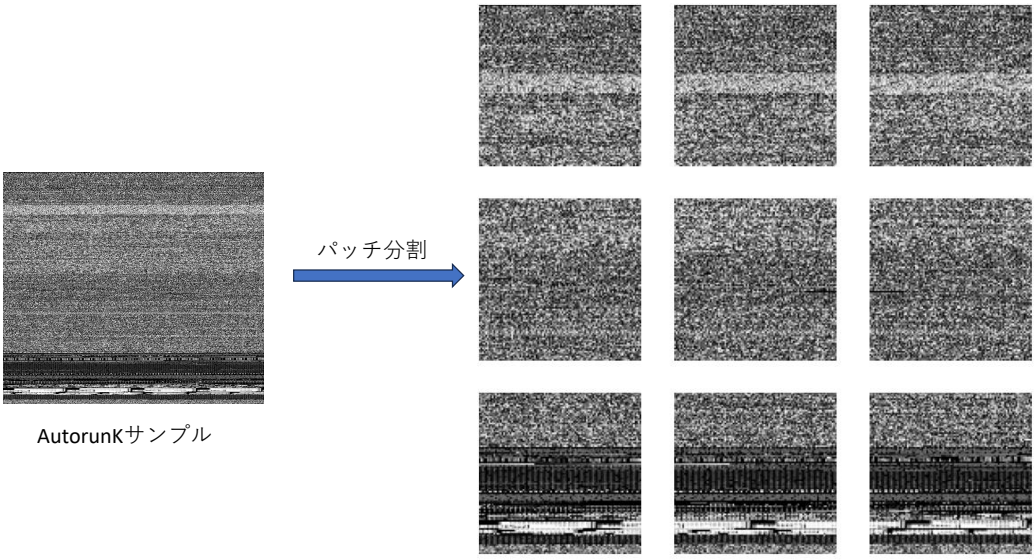


図 3.1: パッチ作成

図 3.1 に示すように MLP-mixer は入力画像を各パッチに分割する．入力サンプルのサイズは（W,H,1）とし，W は画像の幅，H は画像の高さ．分割するパッチのサイズは（P,P,1），P は画像の幅，あるいは，高さ．パッチの数は S とし，は以下に計算される：

$$S = (W \times H)/(P^2) \tag{3.1}$$

パッチの次元は（S,P,P,1）となる．処理の流れを図 3.2 に示す．4 次元パッチをまとめ 2 次元に変更され，その後，次元削減をするため，線形変換を行い．新しいチャンネル次元は C とする（$C \ll P \times P$）．変換されたそれぞれのパッチは複数の Mixing レイヤーを通して，特徴を抽出する．チャンネル数は変換せず，最終的の出力のチャネル次元も C である．

ここで，Mixing レイヤーを詳細を見てみる．各 Mixing レイヤーを図 3.3 に示す．MLP-mixer には，二つの Mixing レイヤー — Token-mixing 及び Channel-mixing — が存在する．それぞれの Mixing レイヤーは二つの隠れ層を持つニューラルネットワークである．隠れ層の間は活化関数 GELU を使われている．GELU 活性化関数は，Gaussian Error Linear Unit functions の略称であり [102]，OpenAI GPT や BERT などの有名なモデルで使われている活性化関数である．GELU の関数形は以下で定義される：

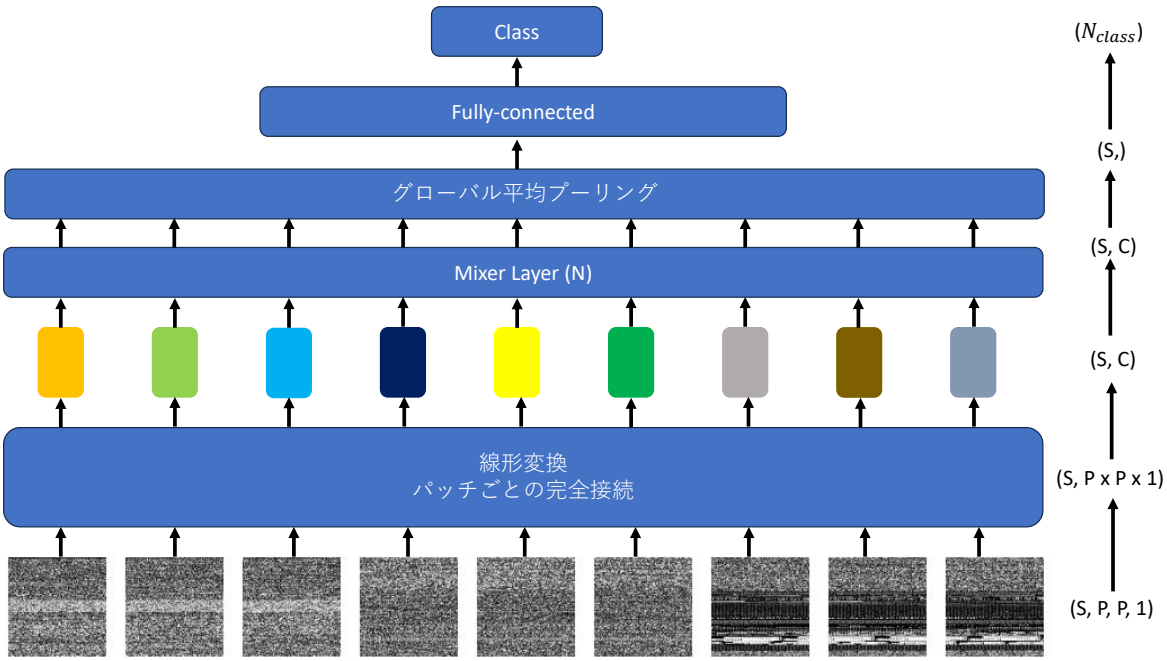


図 3.2: MLP-mixer の構造

$$GELU(x) = \Phi(x) \times Ix + (1 - \Phi(x)) \times 0x = x\Phi(x) \tag{3.2}$$

ここで $\Phi(x)$ は標準正規分布の分布関数とする．ReLU はインプットの値によって確定的に 0 もしくは 1 を掛けるものに対し，GeLU では，インプットの値に依存するように，その確率に正規分布の分布関数を使う．しかし，標準正規分布の分布関数や誤差関数は解析的に計算できないため，以下のように近似する．

$$GELU(x) \simeq x\sigma(1.702x) \tag{3.3}$$

Token-mixing 及び Channel-mixing は同じ構造であるが，入力対象が異なる．図 3.3 に見られるように，入力パッチをテーブルとして，各行はパッチ，列はチャンネルとみなす．Token-mixing は各列ごとの情報を共有することに対し，Channel-mixing は各行ごとの情報を共有する．つまり，Token-mixing は各パッチの情報を共有するため生成されたニューラルネットワークであり，Channel-mixing の方では，チャンネルの情報を共有するためのものである．Token-mixing の処理は以下の数式で表現される．

$$\mathbf{U}_{*,i} = \mathbf{X}_{*,i} + \mathbf{W}_2\sigma(\mathbf{W}_1 LayerNorm(\mathbf{X})_{*,i}) \tag{3.4}$$

ここでは，入力パッチを正規化（Layernorm）を行い，MLP ニューラルネットワークで処理するため，テーブルを転置する必要がある．Token-mixing の MLP ブロック内にそれぞれ重

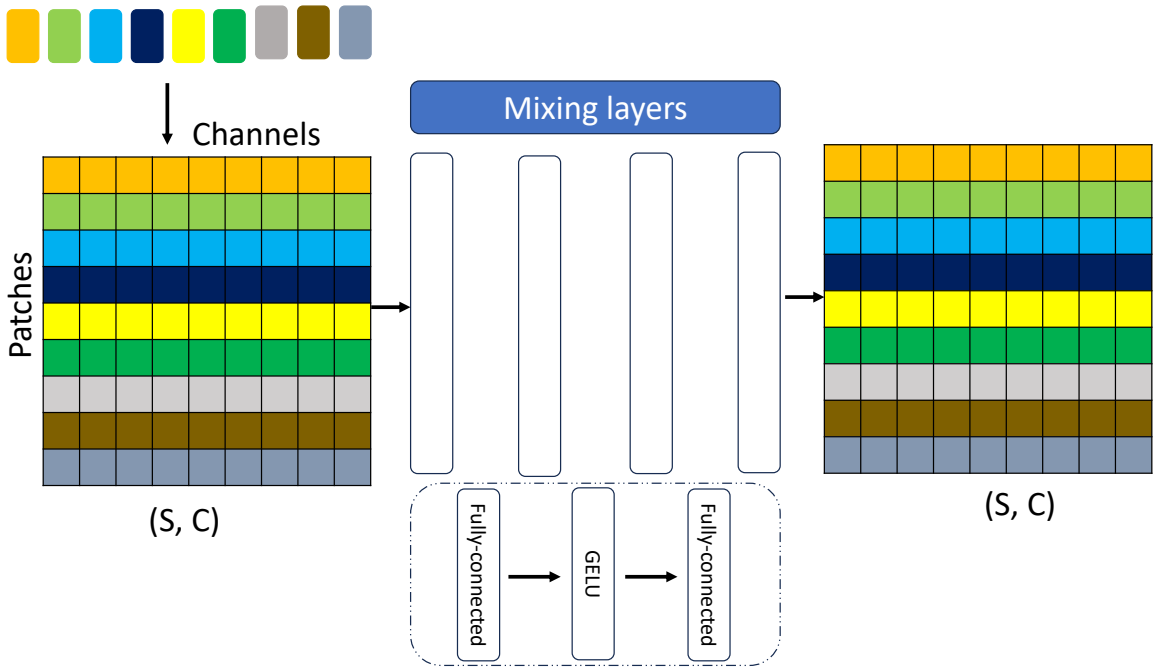


図 3.3: MLP-mixer レイヤーの構造

み $\mathbf{W}_1, \mathbf{W}_2$ とし GELU 関数の σ を用いて学習を行う．次に，Skip-connection を行い，元の入力を補足する．Token-mixing の MLP から得られた出力 $\mathbf{U}_{*,i}$ をまた転置し，元の入力の形になる．これを用いて，Channel-mixing の入力対象とする．CNN では各チャンネルで別々に処理がなされるのに対し，Token-mixing は各チャンネルは MLP のパラメータを共有することによってC及びSを値を大きくしてもモデルが肥大化していない（パラメータが急激に増加しない）という特徴を持つ．その結果，空間計算量を節約することができる．

Channel-mixing の処理は以下の数式で表現される．Token-mixing と同じく，正規化した後，MLP ブロックでそれぞれ重み $\mathbf{W}_3, \mathbf{W}_4$ 及び GELU 関数の σ を用いて学習を行う．

$$\mathbf{Y}_{j,*} = \mathbf{U}_{j,*} + \mathbf{W}_4\sigma(\mathbf{W}_3 LayerNorm(\mathbf{U})_{j,*}) \tag{3.5}$$

CNN はフィルタを画像の領域をかけることによって重み共有することが可能であるため，MLP とは異なり，物体が画面上の違う位置にあっても認識することができる．つまり，位置的不変性（Positonal Invariance）の能力を持つ．更に，空間計算量を節約することもできる [103]．一方，Channel-mixing では，パッチ間は MLP のパラメータを共有する．これは CNN と似ている，画像の領域はすべて同じパラメータのセットを使用する．その結果，CNN と同時に位置的不変性の能力を向上することができる．

ニューラル ネットワークは，トレーニング中に，勾配消失または勾配爆発に関連する問題に頻繁に遭遇する．そのため，レイヤーの出力に入力を追加し，モデルの勾配流を増やすこと

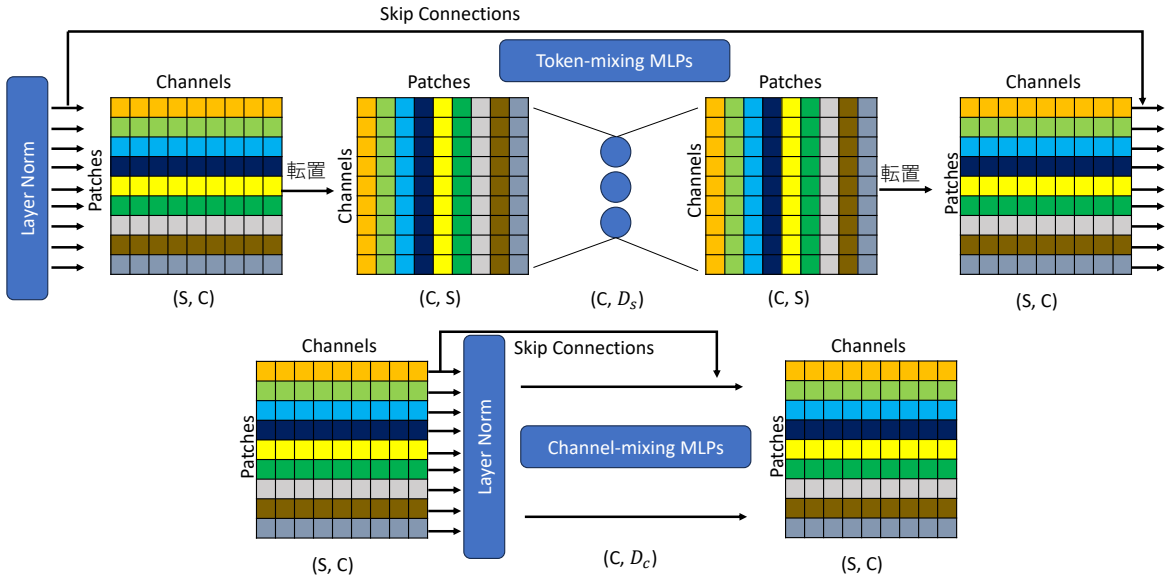


図 3.4: MLP-mixer レイヤーの学習過程

で，情報があまり変更されず，レイヤー間を完全に循環することができる [76].

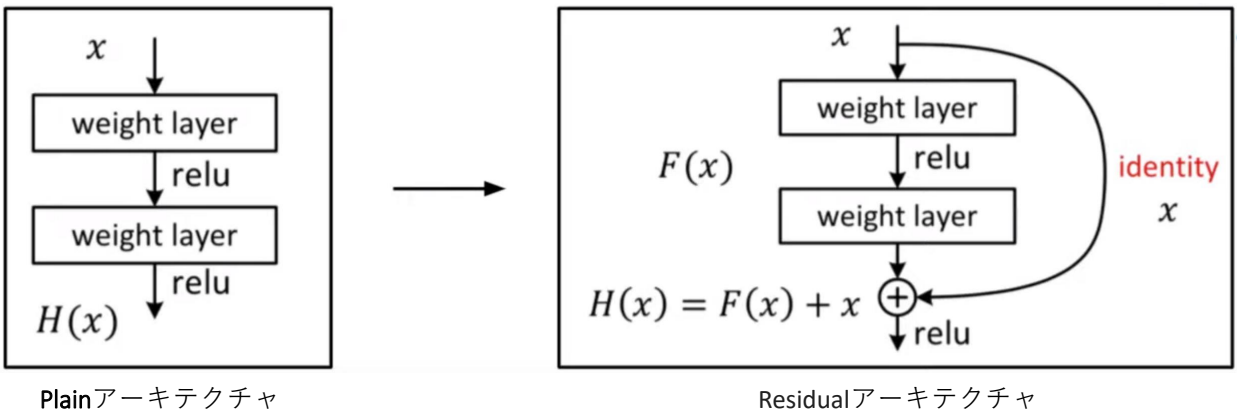


図 3.5: Skip connection [76]

MLP-mixer ブロックが N 回繰り返された後，GAP（Global Average Pooling）を用いて次元を縮小させるとともにサンプルの特徴を集約する．そして分類タスクの前に全結合層が続く．

3.3 MLP-Mixer-AE

MLP-mixer は Token-mixing と Channel-mixing ブロックを通じてニューラルネットワークの構造を最適化しているが，悪意のあるコードを識別するために必要な機能は付与されていない．Tolstikhin et al.[90] では，GAP を使用してすべてのパッチを介してチャンネルを要約したものを全結合層に入力するだけであったが，わずかなアテンションを加えるだけで，モデルがより効率的になることが Liu et al.[104] で発見された．アテンションを用いる理由は，入力から重要な特徴を再選択するためである．ここでは，複雑なアテンションを使用する代わりに，教師なし学習でおなじみのニューラルネットワークであるオートエンコーダを選択し，MLP-mixer を経由した後に特徴を絞り込む．エンコーダとデコーダの構造は，画像から最も重要な特徴をデータの形で抽出し，ネットワーク内の様々な入力間の価値ある相関関係を確立するのに役立つ．

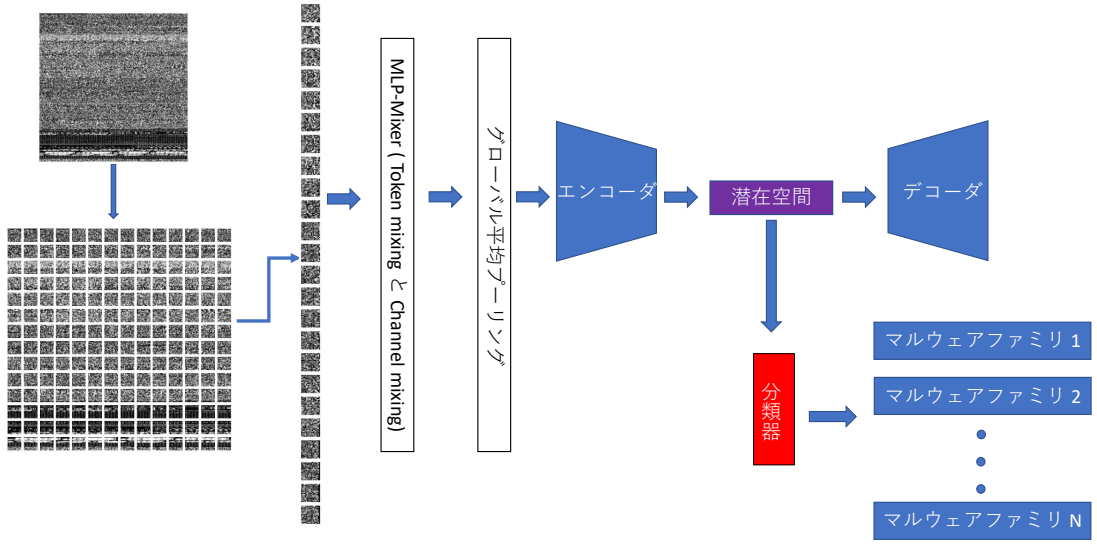


図 3.6: MLP-mixer-AE のトレーニングフェーズ

図 3.6 はトレーニングにおけるシステムのアーキテクチャを示している．前処理でマルウェア画像をリサイズした後，S 個のパッチに分割する（パッチサイズを 8×8 とした）．その後，LayerNorm でパッチが正規化され，マルウェア画像はパッチとチャンネル（S，C）のテーブルに変換される．このテーブルは，GELU 活性化とともに隠れノード D_s を持つ Token-mixing に入力するために (C, S) 転置される．その後，Channel-mixing に投入する前に，テーブルを転置して初期フォームに戻し，初期テーブルと集約する．これらのブロックでは，Channel-mixing の隠れノードは D_c であり，活性化は GELU である．Token-mixing ブロックと同じように，

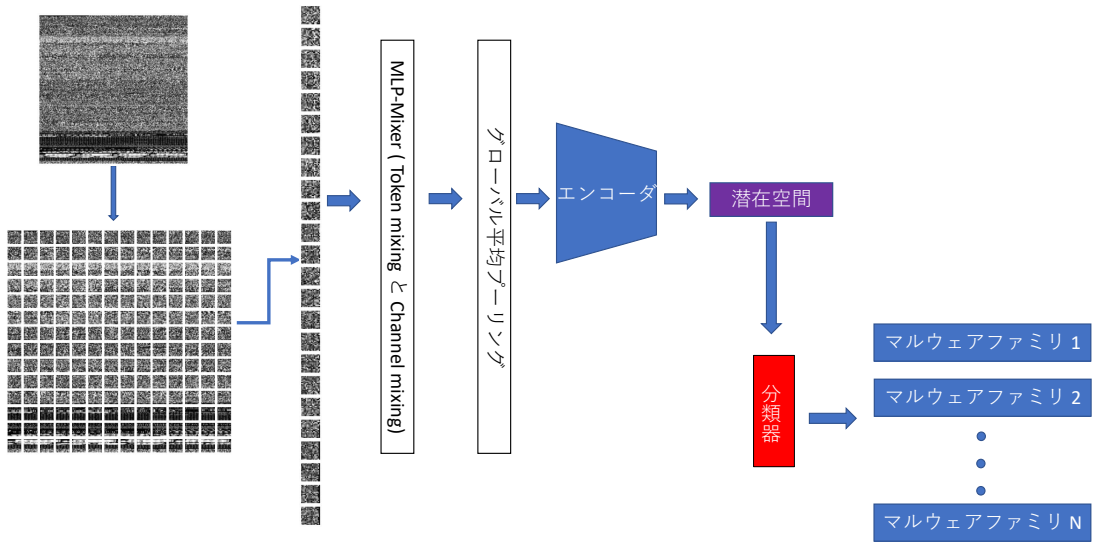


図 3.7: MLP-mixer-AE のテストフェーズ

勾配の消滅を避けるために skip connection が使われる．MLP-mixer ブロックは n 回繰り返され，その後に GAP 層が続く．我々の実装では，C = 128, D_s = 64, D_c = 512 の軽量 MLP-mixer を利用し，原論文の最小モデルスケールである C = 256, D_s = 256, D_c = 2048 に比べて計算コストを削減している．本研究では，エポック数は 50 とした．

提案手法では，MLP-mixer モデルから GAP 層以降の特徴ベクトルを，二つ層のエンコーダーと二つ層のデコーダーを持つ単純なオートエンコーダーモデルに連続的に取り込む．エンコーダの第一層のノードは 2 * C チャンネルで，C チャンネルと C/2 が潜在空間に続く．平均二乗誤差 (MSE) 損失関数と 50 エポックを用いて AE を訓練し，決定木 (Decision Tree)，k-最近傍 (k-Nearest Neighbors)，ナイーブベイズ (Naïve Bayes)，最近傍セントロイド (Nearest Centroid)，ランダムフォレスト (Random Forest)，SVM といった機械学習の典型的な分類アルゴリズムを用いて潜在空間を分類し，システムを 10-分割交差検証を用いて評価する．

図 3.7 にテストのアーキテクチャを示す．テストでは，学習済の重みを用いる．入力サンプルに対して軽量な MLP-mixer モデル及びエンコーダで得られた特徴量を用いて，分類を行う．

3.4 実験結果

3.4.1 データセット

提案モデルを二つのマルウェアデータセットで評価した： Malimg[46]，Malheur Dataset [105] は，表 3.1 に示すように 24 の異なるファミリの 3133 個のマルウェアバイナリから構成され，表 2.1 のルールに従って画像化される．

表 3.1: Malheur データセットの詳細

クラス	ファミリ名	サンプル数	比率 (%)
0	Adultbrowser	262	8.36
1	Allaple	300	9.58
2	Bancos	48	1.53
3	Casino	140	4.47
4	Dorfdo	65	2.07
5	Ejik	168	5.36
6	Flystudio	33	1.05
7	Ldpinch	43	1.37
8	Looper	209	6.67
9	Magiccasino	174	5.55
10	Podnuha	300	9.58
11	Posion	26	0.83
12	Porndialer	98	3.13
13	Rbot	101	3.22
14	Rotator	300	9.58
15	Sality	85	2.71
16	Spygames	139	4.44
17	Swizzor	78	2.49
18	Vapsup	45	1.44
19	Vikingdll	158	5.04
20	Vikingdz	68	2.17
21	Virut	202	6.45
22	Woikoiner	50	1.59
23	Zhelatin	41	1.31

3.4.2 評価結果と考察

提案手法が持つ分類器の候補として，決定木，k 近傍法，ナイーブベイズ，Nearest Centroid，ランダム・フォレスト，サポートベクターマシンなどの一般的な分類器を用い，両データセットに利用した．表 3.2 はパラメータ数の比較である．提案手法は既存研究より少ないパラメータ数であることが分かる．

表 3.2: 既存研究と学習可能パラメータの比較

既存研究	学習可能パラメータ数（M）
Barros et al.[85]	139.67
Roseline et al.[81]	134.36
Nisa et al.[57]	88.26
Lee and Lee[40]	23.81
Hammad et al.[106]	4.00
提案手法	**2.05**

表 3.3, 表 3.4 は，様々な画像サイズに応じた両データセットの性能を示している．入力画像サイズがパフォーマンスに大きな影響を与えることがわかる．最小の画像サイズ 32×32 でも，提案モデルは，オリジナルの MLP-mixer と ResNet50 より 10% 近く精度が良い．他の分類器と比較すると，SVM はほとんどの評価指標で上回っており，文献 [97] と同じ結果となっている．さらにこの結果は，画像の圧縮が分類性能に悪影響を与えないことを示している．

一方，表 3.3 及び表 3.4 の結果から，MLP-mixer 単体では，ResNet などの CNN モデルを性能的に上回っていないという，原著論文 [90] と同じ結果が得られてた．ほとんどの場合，MLP-mixer 単体の精度は ResNet50 より低く，Malimg データセットでは 0.5% から最大 1.93%，Malheur データセットでは 4.04% であった．

CNN は位置的不変性（Positonal Invariance）を利用し，画像の他の場所でも物体情報を得ることができる．これにより，CNN モデルは従来の MLP よりも優れた性能を発揮する．しかし，バイナリファイルや悪意のあるコードは，セクションの比率が異なるだけで，アーキテクチャは固定されている [111]．その結果，CNN はマルウェアから生成された画像の処理において，その強みを発揮することができない．表 2.7 に示すように，CNN は入力画像から属性を学習するために多くのパラメータを結びつける．同時に，MLP-mixer は（典型的な ResNet50 モデルと比較して）1/20 以下の学習可能パラメータしか必要としないが，それでも CNN と同等の高いパフォーマンスを発揮する．

提案手法の性能を評価するために，正解率，適合率，再現率，F 値を，Malimg データセット

（表 3.5）および Malheur データセット（表 3.6）で行われた既存研究の結果と比較した．提案手法では，軽量 MLP-mixer モデルから作成された潜在空間を，シンプルで効果的なオートエンコーダーモデルで精錬する．より少ない十分なパラメータを使用することで，提案手法は，与えられたデータセットに対し，これまでの CNN フリーの既存研究よりも高い性能を達成した．Malheur データセットに関して，Kim et al.[110] は提案手法より 2.16% 高い再現率を達成したが，適合率は 8.08% と明らかに低い．その結果，F 値で計算した総合性能は提案手法の方が 0.23% 高くなった．

3.5　第 3 章のまとめ

本章では CPU しか使用できない環境でも動作できる軽量なニューラルネットワークを提案し，高精度なマルウェア分類を実現した．現在，畳み込みネットワークはポピュラーなツールとなっているが，最近の研究では，MLP や Vision Transformer のアテンションメカニズムである単一記述子や複数記述子のような，畳み込みネットワークを使わないモデルに置き換えて規模を小さくする試みがなされている．とはいえ，この分野の研究は始まったばかりで，最先端の CNN モデルのような期待される結果はまだ得られていない．本研究の結果，複雑なかつ巨大なモデルを使用しなくても軽量なモデルでも画像ベースマルウェアを高い精度で分類できた．このことは，画像に基づくマルウェア分類タスクに期待できる新たな方法を示唆できたと言える．

本章は 5 つの節から構成された．3.1 節では，ニューラルネットワークを用いた画像に基づくマルウェア分類の既存研究を紹介し，挙げられたアーキテクチャは規模が大きいにもかかわらず畳み込みネットワークより精度を上回らないことが分かった．3.2 節では，MLP-mixer を紹介した．様々な特徴の抽出の仕方により，従来の多層パーセプトロンより性能が向上することが分かった．これを踏まえて 3.3 節では，それぞれ MLP-mixer 及びオートエンコーダを活かしたモデルを提案し，3.4 節では性能を確認するため様々実験を行った．それらの結果を考察し，提案手法の有効性を示した．

MLP-mixer の誕生により，現在のコンピュータビジョンにおける畳み込みネットワークと比較して，多層パーセプトロンの強さが改めて確認された．悪意のあるコードから作られた画像を処理する問題では，悪意のあるコードの構造的特徴により，畳み込みネットワークは位置的不変性能力を活用できない．パッチやチャネルを介してパラメータを共有するニューラルネットワークでは，多層パーセプトロンと CNN の性能差は大きく縮まっている．特徴ベクトルの再精錬の問題は，シンプルだが非常に効果的なオートエンコーダーネットワークによって解決された．実験の結果，提案手法は畳み込みネットワークを用いないモデルよりも優れており，純粋な畳み込みネットワークの典型的なモデル ResNet50 と比較して性能が向上していることが示された．さらに，提案した軽量なアーキテクチャは，オリジナルの MLP-mixer モデルや畳み込みネットワークの複雑な組み合わせアーキテクチャ，アテンションメカニズムを持つ AVAE[83] と比較して，より少ない十分なパラメータ数であるにもかかわらず，様々な実験を通して，高い性能を達成している．なお，本研究では，計算時間を節約するために，Ghouti and Imam[112] のようなグリッドサーチハイパーパラメータチューニングは用いていない．

本章までは，画像に基づくラベル付きの教師あり学習であるモデルを提案したが．教師データの無い未知のマルウェア分類タスクには対応していない．これを解決するために，次の章でさらに新しいモデルを提案する．

表 3.3: Malimg データセットにおける入力画像による性能比較

入力サイズ	手法	分類機	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
32 × 32	MLP-Mixer-AE	Decision Tree	76.88	69.79	72.10	70.96
		k-Nearest Neighbors	89.75	91.14	85.36	86.92
		Naïve Bayes	79.53	79.70	75.79	77.09
		Nearest Centroid	89.72	91.58	93.31	92.09
		Random Forest	88.67	90.36	79.88	81.67
		SVM	**94.67**	**95.19**	**93.42**	**94.12**
	MLP-Mixer	Softmax	84.62	-	-	-
	ResNet50	Softmax	84.94	-	-	-
64 × 64	MLP-Mixer-AE	Decision Tree	91.61	82.33	82.29	81.50
		k-Nearest Neighbors	97.27	95.22	93.06	93.72
		Naïve Bayes	96.04	91.38	91.45	91.24
		Nearest Centroid	98.27	95.79	96.17	95.91
		Random Forest	97.45	95.48	93.51	94.17
		SVM	**98.66**	**97.06**	**96.61**	**96.77**
	MLP-Mixer	Softmax	95.82	-	-	-
	ResNet50	Softmax	98.11	-	-	-
96 × 96	MLP-Mixer-AE	Decision Tree	92.98	84.99	85.76	85.24
		k-Nearest Neighbors	98.52	96.79	96.26	96.45
		Naïve Bayes	96.41	72.74	93.23	92.85
		Nearest Centroid	98.49	96.59	96.83	96.66
		Random Forest	98.31	96.45	95.67	95.97
		SVM	**99.05**	**97.82**	**97.58**	**97.66**
	MLP-Mixer	Softmax	97.25	-	-	-
	ResNet50	Softmax	98.43	-	-	-
224 × 224	MLP-Mixer-AE	Decision Tree	95.41	90.19	90.34	89.78
		k-Nearest Neighbors	99.06	97.85	97.73	95.75
		Naïve Bayes	98.21	96.18	96.19	96.09
		Nearest Centroid	98.68	97.15	97.29	97.16
		Random Forest	99.12	97.95	97.84	97.91
		SVM	**99.34**	**98.38**	**98.26**	**98.29**
	MLP-Mixer	Softmax	97.75	-	-	-
	ResNet50	Softmax	99.14	-	-	-

表 3.4: Malheur データセットにおける入力画像による性能比較

入力サイズ	手法	分類機	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
32 × 32	MLP-Mixer-AE	Decision Tree	89.08	83.71	83.20	82.96
		k-Nearest Neighbors	98.15	97.46	95.96	96.43
		Naïve Bayes	97.32	95.24	95.89	95.33
		Nearest Centroid	98.18	96.92	96.28	96.40
		Random Forest	97.83	97.56	95.36	96.22
		SVM	**98.37**	**97.78**	**96.44**	**96.71**
	MLP-Mixer	Softmax	94.47	-	-	-
	ResNet50	Softmax	91.38	-	-	-
64 × 64	MLP-Mixer-AE	Decision Tree	85.37	76.24	75.46	74.72
		k-Nearest Neighbors	96.74	94.03	94.03	94.61
		Naïve Bayes	95.82	94.35	94.35	93.32
		Nearest Centroid	97.35	95.88	95.88	95.55
		Random Forest	96.52	93.37	93.37	94.14
		SVM	**97.89**	**96.70**	**96.70**	**96.70**
	MLP-Mixer	Softmax	93.09	-	-	-
	ResNet50	Softmax	96.06	-	-	-
96 × 96	MLP-Mixer-AE	Decision Tree	87.61	80.38	80.29	79.73
		k-Nearest Neighbors	97.64	96.92	95.13	95.62
		Naïve Bayes	96.36	93.65	95.02	94.02
		Nearest Centroid	97.92	96.64	**96.23**	96.18
		Random Forest	97.51	96.74	94.48	95.36
		SVM	**98.02**	**97.05**	96.05	**96.31**
	MLP-Mixer	Softmax	93.30	-	-	-
	ResNet50	Softmax	97.34	-	-	-
224 × 224	MLP-Mixer-AE	Decision Tree	88.50	84.10	82.63	81.98
		k-Nearest Neighbors	97.92	**97.44**	95.71	96.13
		Naïve Bayes	97.03	94.89	95.37	94.90
		Nearest Centroid	98.05	97.03	**96.43**	96.42
		Random Forest	97.70	97.22	95.17	95.98
		SVM	**98.15**	97.24	96.38	**96.50**
	MLP-Mixer	Softmax	94.79	-	-	-
	ResNet50	Softmax	97.87	-	-	-

表 3.5: Malimg データセットにおけるマルウェア分類 CNN フリーモデルとの比較.

モデル	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
GIST feature + kNN [46]	97.18	-	-	-
Combined SIFT-GIST [33]	98.40	-	-	-
SFTA + Cosine kNN [57]	98.70	-	97.0	-
Multiple Autoencoders [40]	97.75	95.0	94.0	93.0
Feature Extraction Tamura [106]	95.42	-	-	-
Feature Extraction GoogleNet [106]	96.48	-	-	-
CliqueNet + Multiscale Attention [107]	99.2	98.0	97.9	97.9
Dimension Reduction + SVM [97]	98.51	-	-	-
DNN + DGAN [41]	95.63	95.34	95.30	94.98
提案手法（MLP-mixer-Autoencoder）	**99.34**	**98.38**	**98.26**	**98.29**

表 3.6: Malheur データセットにおけるマルウェア分類 CNN フリーモデルとの比較.

モデル	正解率 (%)	適合率 (%)	再現率 (%)	F 値 (%)
Euphony [108]	-	90.06	83.86	86.85
Combined SIFT-GIST [33]	97.50	-	-	-
AV labels [109]	-	90.81	88.45	89.61
Multiple AV [110]	-	89.70	**98.60**	93.94
Dimension Reduction + SVM [97]	95.79	-	-	-
提案手法（MLP-mixer-Autoencoder）	**98.37**	**97.78**	96.44	**96.71**

第 4 章

未知のマルウェア分類：ZSL-SLCNN の提案

本章では，未知ラベルに対応したゼロショット学習を用いて未知のマルウェアを分類するための新たな手法を提案する．ここで「未知ラベル」とはマルウェアファミリの名前は付いているが，サンプルが一つもないケースを指すラベルであるとする．ゼロショット学習を活用することによって，ラベル付きデータがなくとも未知のマルウェアを検出でき，また軽量なモデルでありながら，既存手法より精度が向上したことを示す．

第 2 章及び第 3 では，教師あり学習における軽量なモデルを用いて高精度でマルウェア分類ができたが，教師あり学習ではサンプル及びラベル付きデータが必要である．そのため，未知のマルウェアに対応できない事案が近年増加している．これは現在のマルウェア分類手法のほとんどは教師あり学習に基づく手法であり，ラベルが得られていない新しいマルウェアを分類することが原理的に難しいためである．そこで，未知のマルウェアを分類できる新たな方法が求められている．

ゼロショット学習 (ZSL: Zero-Shot Learning) は，マルウェアのサンプルがこれまでなかったラベルに属している場合でも，モデルがサンプルをマルウェアとして認識して分類できるため，マルウェア分類に対する有望なアプローチを提供する [85]．普通の教師あり学習では不可能と思えることをゼロショット学習が行えるのは，ゼロショット学習が事前に学習したモデルに基づいて，異なるラベル間の共通の特徴や関係性を考慮できるためである．

例えば，図 4.1 に示すように学習された馬，自動車，犬を空間に表示できたとする．トレーニングデータ (画像) の特徴を高次元のラベル空間（セマンティック埋め込み空間）にマッピングする．この空間では類似のオブジェクトや概念が近接して配置されるため，未知のデータが既知のデータに近い位置にあれば，適切にそのラベルを推測できる．ここで未知の猫とトラックのラベルはセマンティック埋め込み空間でばそれぞれの類似のものに近くように配置される．未知の画像はその特徴に基づき空間にマッピングされる．マッピングされた点から最も近

いラベルを選ぶことで，未知の画像に対するクラス推定が行える．

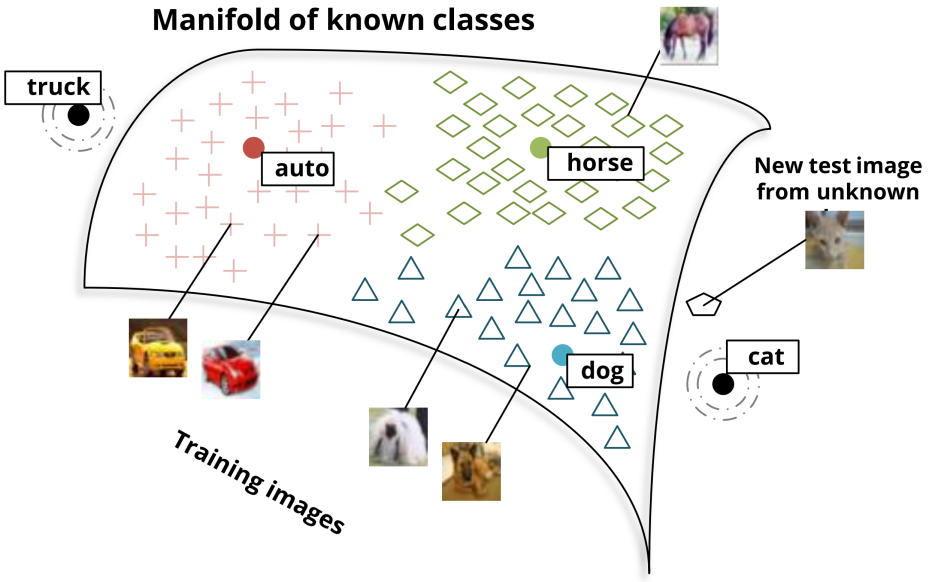


図 4.1: ゼロショット学習 [113]

既存研究 [113] ではこのようなマッピング手法で，一つの画像化したマルウェアの特徴空間から一つのセマンティック埋め込み空間へマッピングすることで，未知のマルウェアを推定可能としたものの，性能が性能が十分ではない．その理由として，少なくとも次の二つが考えられる．一つ目は [113] は Word2vec を用いてラベル空間を生成するため，学習済の辞書にない単語は利用できない (1)．二つ目はラベル空間へ様々な対象画像をマッピングできることが重要であり，すべての対象をカバーするのに十分適切であることを保証する上で限界が生じやすい，つまり，マッピング自体の妥当性の欠如 (2) である．

まず (1) に関しては，巨大な Transformer で新たな単語に対応することが可能である CLIP モデル [114] があるが，Transfomer 自体は軽量なモデル向けではない．そこで，本研究は Word2vec，Transformer の代わりに Fasttext[115] を使用する．Fasttext は軽量でありながら未知の単語の特徴ベクトルを推定する機能がある．

(2) に関しては，写像の妥当性を高めるため，本章ではもう一つの画像空間とセマンティックラベル空間を生成する二層のマッピングを行い，それぞれの層の関連性を総合的損失関数で強調する手法を提案する．ここで，総合的損失関数とは，それぞれの層に写像する際に用いた損失関数をまとめたものである．ここで用いる畳み込みネットワークは，単語ベクトル空間でのラベルを扱うため「セマンティックラベル畳み込みネットワーク」(SLCNN) と名付けた．

未知のマルウェアに対する対策の難しさにはいくつかの理由が挙げられる．一つには，シグネチャベース手法の限界があるる新しい署名を作成するのは時間がかかるため，即座に対応で

きない．二つには，マルウェア制作者は，ポリモーフィックおよびメタモーフィックな技術を使用して，マルウェアのコードを動的に変化させる．これにより，同じマルウェアでも異なるバージョンが作成され，シグネチャベース手法を回避することができる．三つには，ゼロデイ攻撃は，まだ修正プログラムや対策策がないセキュリティ上の脆弱性を利用する攻撃である．マルウェア制作者はこれらの脆弱性を見つけ，それを悪用して対策が存在しない状態で攻撃を仕掛けることがある．これらの要因により，未知のマルウェアに対する完璧な対策が難しくなっている．これに対し，ZSL は，各マルウェアファミリの大規模なラベル付きデータセットの必要性を低減できるため期待が大きい．

とはいえ，この方法はまだ期待された程のパフォーマンスを達成していない．現在の研究では，CNN の全結合層以降のレイヤーである画像特徴空間からセマンティック空間へのマッピングのみを学習し，最も近いラベル埋め込みベクトルを探索する複雑なモデルを開発することに主眼が置かれている [113, 116, 117, 118, 119, 120]．本章では，一つのマッピングレイヤーとして上記のアプローチを挙げる（図 4.2）．入力マルウェアは画像を ResNet50 モデルで特徴を抽出し，得られた特徴をセマンティック空間への投影し，ラベル空間とのコサイン距離を測り，一番類似度を高いものが出力となる．このアプローチは，マッピング空間が重要であり，すべてのケースをカバーするのに十分適切であることを保証する上で限界が生じやすい．さもないと，オーバーフィッティングの問題につながる．一方では，上記の欠点を克服するために，マッピングを正しい方向に向けるための追加属性を提供するものもある [118, 119, 120]．とはいえ，特にマルウェアの分類タスクでは，マルウェアファミリ全体に共通する属性を見つけ，構築するには，大量のラベルを消費する必要がある．その結果，マルウェア分類への ZSL の適用はまだ限定的であり，不足している．

デジタル時代において，悪意ある行為者は脆弱性を悪用し，システムをマルウェアに感染させる新たな方法を絶えず模索している．マルウェアの作成者は，その攻撃手法や標的アプローチを絶えず革新している．新たなマルウェアの出現数は一向に減少する気配を見せず，このままでは，マルウェアの脅威がますます拡大してしまう可能性がある．Mandiant のレポートによると，新たに確認されたマルウェアは，2021 年の月間 45 件から，2022 年には月間約 49 件に上った [121]．

ここで，図 4.3 に示すように提案手法で追加したもう一つの層によってマッピングの次元を多様化することで，意味空間の対応するラベルと比較して，分類されるオブジェクトの複数の視点を受け取ることができ，オーバーフィッティングを防ぐことができると考えられる．

本章では，視覚的特徴と意味空間との対応付けをより効果的に行うことができる新しい手法を提案する．Malimg データセットを用いて，別のマルウェアデータセットの未知のラベルを用いた学習とテストの実験を行った．本章の主な貢献は，包括的な損失関数を用いた二層マッピングの処理に関する新しい視点を提供することである．浅いアーキテクチャは，学習する特徴がそれほど複雑でない比較的単純な画像認識タスクに適しており，より深遠で複雑な畳み込

図 4.2: 従来手法のゼロショット学習における一つのマッピングレイヤー [113]

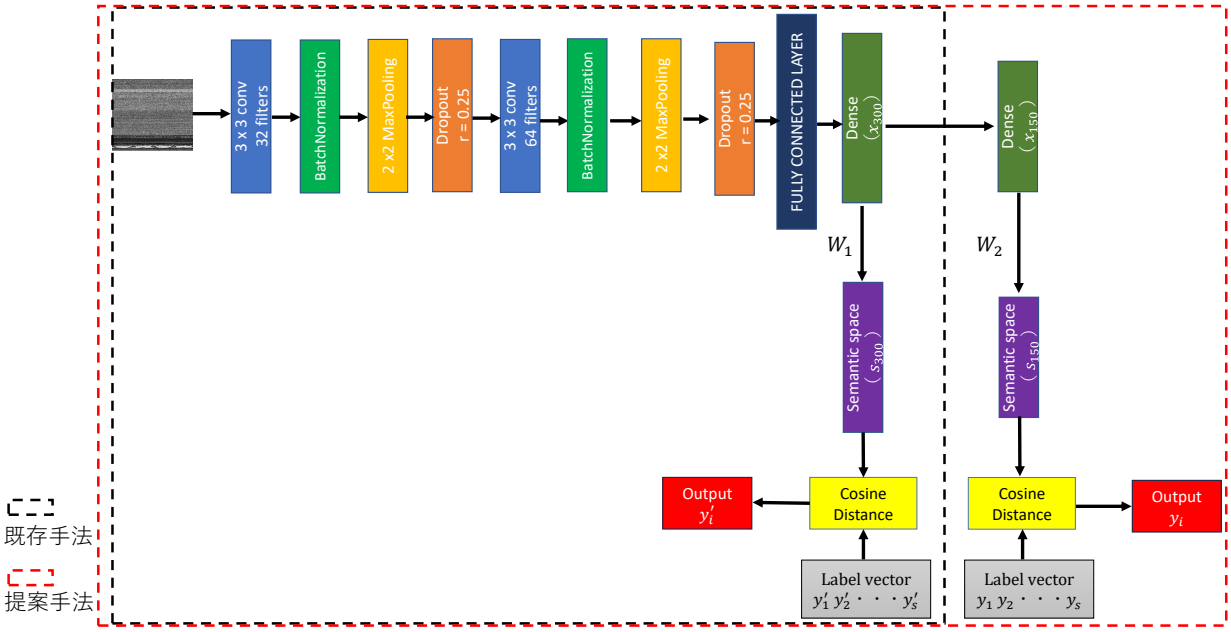


図 4.3: 提案手法と従来手法の比較

みネットワークのアーキテクチャよりも，GPU メモリや学習時間などの計算資源が少なくて済む．実験結果は，提案手法が CNN の畳み込みの二層と事前に訓練されたラベルだけで複雑なモデルを凌駕できることを示している．

4.1 関連研究

ZSL は，視覚的に見かけたラベルから得た知識を未知のラベルに転送する形式の転移学習の一形態と考えることもできる．この際，手動で作成された属性やテキストから学習された word2vec[122] などの補助情報を提供する [123]．そして，ゼロからトレーニングする代わりに，さまざまな自然言語処理アプリケーションで広く使用されている Fasttext [115] からの事前トレーニング済みの単語埋め込みを使用する．FastText は，フェイスブックの AI リサーチ（FAIR）ラボによって開発された人気のオープンソースライブラリであり，リサーチモデルである．これらの単語埋め込みは，個々の単語だけでなく文字レベルの情報も考慮に入れて作成されており，モデルが未知の単語や稀な用語を効果的に処理できるようにしている．

自然言語処理と画像処理の手法を活用することに加え，このモデルはマルウェアサンプルのマルチモーダルな理解を獲得し，より正確な分類につながる可能性がある [114]．一方，同じラベルの画像は，セマンティック空間に埋め込まれた後，そのラベルの意味的埋め込みを中心にクラスター化する [116]．画像に基づくマルウェアの特徴と，すべてのラベル（既知と未知のもの）との間のリンクを確立するために，モデルは視覚的特徴空間からセマンティック空間への射影関数を学習する，モデルは視覚的特徴空間からセマンティック空間への投影関数を学習する [113, 124, 125]．図 4.4 に示すように，マルウェアを画像化したものから畳み込みネットワークなどのモデルを用いて特徴を抽出し，その視覚的特徴空間とラベル空間の関連性を活かして，未知のラベルを推定することができる．

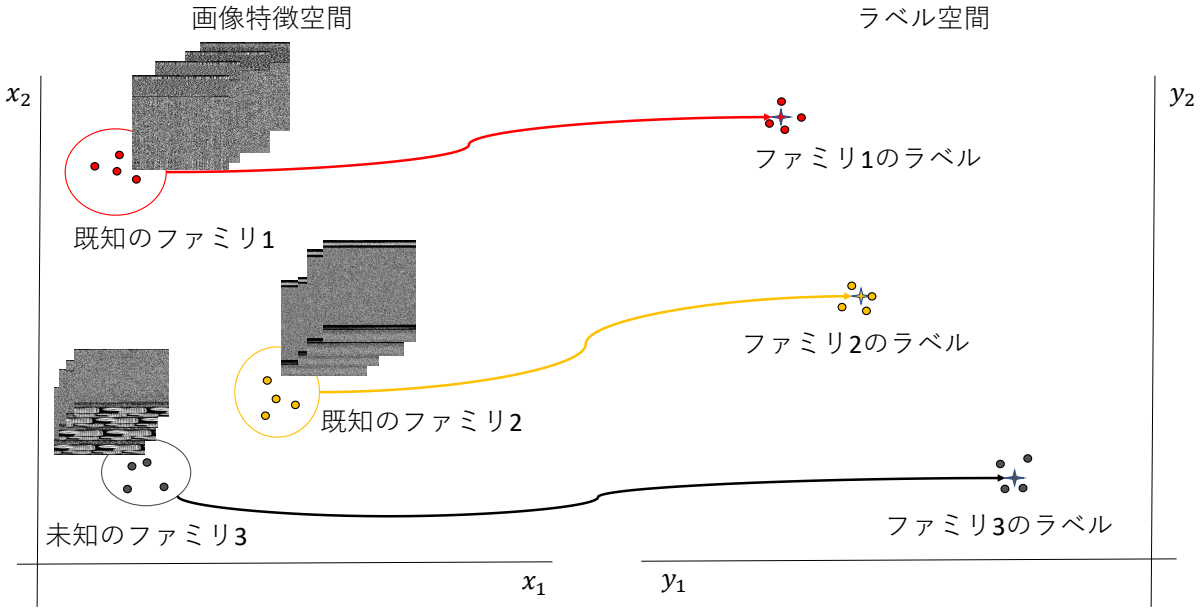


図 4.4: 画像ベースマルウェアに応用するゼロショット学習

最近の研究では，高い分類精度を達成するために，いくつかの複雑なアーキテクチャ [126, 127, 128, 129] からシンプルなアーキテクチャ [81, 130] までが提案されている [131]．しかし，現在のマルウェア分類モデルは教師あり学習に基づくものであり，各ファミリのラベル内の様子の変化を把握するために，膨大な量の学習データを収集し，アノテーションする必要があり，また，未知のマルウェアファミリに対処することはできない．したがって，既知のマルウェアファミリと新しい未知のマルウェアファミリを扱うための新しい技術が必要である．本セクションでは，フューショット学習，ワンショット学習，ゼロショット学習を用いた画像に基づくマルウェア分類に関する様々な新しい研究を紹介する．

4.1.1　フューショット学習，ワンショット学習を用いた未知のマルウェア分類

Chai et al.[132] は，サンプル適応に基づく動的プロトタイプネットワーク DPNSA（Dynamic Prototype Network based on Sample Adaptation）を紹介した．彼らの提案するフューショット学習法は，動的畳み込み層を持つ軽量ニューラルネットワークを含み，サンプル適応による動的特徴埋め込みを容易にする．さらに，性能指標に対するサンプル間の無関係な特徴の影響を最小化するために，二重サンプル動的活性化関数を導入した．数ショットのマルウェアデータセットを用いた実験評価により，DPNSA が様々なシナリオにおいてベースラインモデルと比較して優れた性能を発揮することが実証された．

Conti et al.[133] は，画像に基づくマルウェアを利用して，限られたインスタンスでマルウェアを分類し，未知のマルウェアファミリに対する分類器の再トレーニングを不要にする，数ショットマルウェア分類技術を提案した．彼らは，CSNN と Shallow-FS という 2 つの異なるフューショット学習アーキテクチャを比較した．その結果，CSNN はデータ不足が顕著な場合に適しており，Shallow-FS はそれ以外の場合に優れた性能を発揮することがわかった．実験の結果，CSNN は，3 つの PE マルウェア・データセットにおいて，未知のマルウェア・ファミリの分類において 96.21%，94.99%，93.42% という高い精度を達成した．これらの結果は，データセットの約 10% のみを学習し，残りのデータを評価することによって得られた．

Tran et al.[134] は，Memory Augmented Neural Network (MANNWARE) を使用して，マルウェアの API シーケンスと組み合わせたワンショット学習によるマルウェア分類の有効性を提案した．著者らは，n-gram や word2vec などの NLP アプローチを用いて，明確な特徴を意味空間に変換する．MANNWARE の重要な革新性は，学習データが乏しいにもかかわらず，異なるマルウェア・ファミリの特徴的な特徴を捉える能力にある．メモリを増強したメカニズムにより，モデルはファミリ間のパターンと関係を記憶することができ，それにより分類性能が向上する．著者らの提案手法は，新領域のサンプルを訓練した場合，約 89.59% の精度に達する．

Khan et al.[135] は，マルウェアを良性に分類するためにワンショット学習を使用した．彼

らは Relation Network を実装する新しいアプローチを提案した．このネットワークは，クエリ特徴ベクトルとサポート特徴ベクトルを組み合わせるために，深い残差学習法を適用した．その結果，著者らの提案手法は，一つサンプルだけで，数ショット学習の性能を最大 94% 向上させた．

しかしながら，これらの既存研究は未知のラベルを推定するため，補足データとして，少なくとも一つのサンプルが必要となる．本章で対象とするサンプルが無くても分類可能な手法はゼロショット学習である．

4.1.2 ゼロショット学習を用いた未知のマルウェア分類

Radford et al.[114] が提案した CLIP（Contrastive Language-Image Pre-training）は，OpenAI が開発した強力で革新的な機械学習モデルである．CLIP モデルは，オンラインで見つけた画像と対になったテキストを豊富に利用できるソースを活用する．CLIP モデルはテキストエンコーダと画像エンコーダから構成され，テキスト情報と視覚情報をマルチモーダル埋め込み空間にエンコードする．このモデルの目的は，実際に関連付けられた画像とテキストのコサイン類似度スコアを増加させることである．一方，このモデルはまた，一緒に発生しない画像とテキストの類似度を最小化しようとする．著者らは，画像エンコーダのバックボーンに 2 つの異なるバックボーン（Resnet50 と Vision Transformer（ViT））を使用し，テキストエンコーダのバックボーンに Transformer を使用した．CLIP モデルは，複数の異なるデータセットでスコアを検証することで，現行の SOTA よりも大幅に柔軟性を高めている．

Frome et al.[113] は，NLP と CV を組み合わせた先駆的な手法として，DeViSE モデルを提案した．このモデルは，視覚的類似性と意味的類似性を活用して，未見のカテゴリラベルを正しく予測する．著者らは，ウィキペディアの 570 万文書（54 億語）からなる膨大なコーパスを用いてテキストモデルを訓練しており，これも DeViSE の成功の要因となっている．さらに，著者らの手法の重要な革新性は，埋め込み空間において，類似の画像とテキストのペアを近づける一方で，非類似のペアを遠ざけるように埋め込みを最適化する，その損失関数にある．この損失関数は，視覚的・意味的に関連するアイテム同士を近づけるようにモデルを促す．

これらの既存研究のほとんどは単一マッピングを行うため、妥当性が欠如している．本研究はこれを解決するため，二層マッピングを提案し，それぞれの関連性を総合的損失関数で強調する．

4.2　シンプル CNN

最近の研究では，画像からの特徴抽出における CNN の優位性が示されている．このモデルは，単純な CNN ネットワークでも 90% 以上の高い結果を達成した [81, 34]．本研究では，複雑な CNN モデルを利用するのではなく，画像ベースのマルウェアの特性を抽出するためにカスタマイズされた単純な CNN を利用した．

MaxPooling 層と Dropout 層のドロップ率は 0.25 で，完全連結層の後に，それぞれラベル次元に沿った 300 ユニットと 150 ユニットの二つの層で，ReLU 活性化を持つ．バニラ (通常)CNN では，Cross-Entropy が損失関数とともにソフトマックス関数として使われる．

4.3　Fasttext

単語は単なる One-hot ベクトルで表現することができるが，単語の意味を考慮しないため，自然言語処理の課題として挙げられた．これに対応するために，は 2013 年に Mikolov et al.[122] は大量の非構造化テキストデータから単語のベクトル表現を学習する効率的な方法である Word2vec を紹介した．似たような言葉が似たような文脈で出てくることを注意し，単語ベクトル化することで自然言語処理にて大きいな影響を与えた．しかし，Word2vec には二つの問題点がある．

一つ目は各単語に対して埋め込みが作成されるため，学習中に存在しないない単語は扱えないという問題である．例えば，“tensor ”や “flow ”といった単語は，Word2Vec の語彙の中に存在する．しかし，“tensorflow ”という複合語は語彙にないため，埋め込もうとすると，out of vocabulary エラーとなる．

二つ目は形態論という問題である．“do”と “does”のような同じ部首を持つ単語では，Word2Vec はパラメータを共有しない．各単語は，それが現れる文脈に基づいて独自に学習される．したがって，単語の内部構造を利用して，より効率的な処理を行う余地がある．

上記の課題を解決するために，Bojanowski et al.[115] は FastText と呼ばれる新しい埋め込み手法を提案した．FastText の重要なアイデアは，各単語の文字レベルの n-gram であるサブワードを単語表現の学習に使用できることである．その根拠は，似た形の単語は似た意味（形態素）を持つ可能性が高いということである．例えば，where，who，when，why はすべて 2-gram のサブワード wh を持っている．このような文字構成の類似性は，これらの単語が類似した意味を持っているという情報を持っている．サブワードの生成方法は表 4.1 に示すように，最初単語を取り，角括弧を付けて単語の最初と最後を表す．文字 n-gram は，角括弧の開始から終了角括弧に達するまで n 文字のウィンドウをスライドさせることで生成できる．ここでは，ウィンドウを 1 ステップずつずらす．こうして，単語に対する文字 n-gram のリストを

得る．

一意な n-gram は膨大な数になる可能性があるため，ハッシングを適用してメモリ使用量を抑える．一意な各 n-gram の埋め込みを学習する代わりに，B 個の埋め込みを学習する．[115] では 200 万バケットを用いている．各文字の n-gram は 1 から B までの整数にハッシュ化される．これは衝突を引き起こす可能性があるが，語彙のサイズを制御するのに役立つ．Fowler-Noll-Vo ハッシュ関数の FNV-1a[136] 変種は，文字列を整数値にハッシュするために使用される．最後に，n-gram であるサブワードの平均を取ることによって，訓練データに存在しない単語ベクトルでも表現できる．

表 4.1: 異なる長さの文字 n-gram

単語	長さ (n)	n-gram 文字
malware	3	<ma, mal, alw, lwa, war, are, re>
malware	4	<mal, malw, alwa, lwar, ware, are>
malware	5	<malw, malwa, alwar, lware, ware>
malware	6	<malwa, malwar, alware, lware>

4.4　提案手法 ZSL-SLCNN

本研究では，Fasttext [115] によって学習された単語埋め込みを利用し，デフォルトの次元を 300 とし，初期ラベル空間とする．次のレイヤーのノード数を前のレイヤの半分にして，オートエンコーダの符号化部と同じように次元削減を行い，第 2 のラベル空間を生成する．既知ラベルと未知ラベルは，それぞれ 150 と 300 の二つの空間にベクトル化される．画像ベースのマルウェアの特徴は，セクション 4.2 で紹介したシンプルな CNN によって抽出される．この空間がラベル空間に類似していることを満たす写像関数として，標準的な非線形性 tanh を用いて，画像特徴空間から意味空間への写像を学習する．

ここでは，$\mathbf{Y} = \{y_1, ..., y_s\}$ and $\mathbf{Z} = \{z_1, ..., z_u\}$ を 150 次元に対応する s 個の見たラベルと u 個の見たラベルのセットとし，$\mathbf{Y'} = \left\{y'_1, ..., y'_s\right\}$ を 300 次元とする．両者は分離している $\mathbf{Y} \bigcap \mathbf{Z} = \emptyset$ and $\mathbf{Y'} \bigcap \mathbf{Z'} = \emptyset$.

トレーニングフェーズを図 4.5 に示す．訓練では，既知ラベルデータセットを [115] に埋め込み，ラベル空間 **Y** を作成する．訓練過程を通じて，意味空間 **S** は，コサイン距離を損失として用いることで，ラベル空間 **Y** と近くなるように訓練される．これらの埋め込みと適切なセグメント空間の作成により，学習セットに対応するサンプル画像がない未知のラベルであっても，画像特徴空間とラベル空間の対応付けの互換性を計算することができる．したがって，このフレームワークはゼロショット学習にも適用できる．このマッピングを訓練するために，以下の目的関数を最小化するニューラルネットワークを訓練する：

$$L = \lambda * cosine(\mathbf{W}_1, s_{300}) + (1 - \lambda) * cosine(\mathbf{W}_2, s_{150}) \tag{4.1}$$

ここで，λ は，二つの測定値間の損失比を表すハイパーパラメーターである $0 < \lambda < 1$．二つのセマンティック空間とラベル空間は，それぞれ別のものであるが，式 4.1 は 2 次元空間間の相互作用を調整するものである．その結果，マッピングの多様化が進み，オーバーフィッティングを避けることができる．さらに，二つのセマンティック空間は，非線形関数 ReLU を用いて，密な 300 から密な 150 への画像処理を通じて間接的に相関する．

テスト段階で使われるモデルを図 4.6 に示す．未知のラベルデータセットに [115] を埋め込み，ラベル空間 Z を作る．単純な CNN の最後の Dense レイヤーは，事前に訓練された重みを持つ画像特徴に使われる；どのサンプルが Z のどのラベルに最も似ているかを決定するために，k=1 の kNN とコサイン距離を使う．

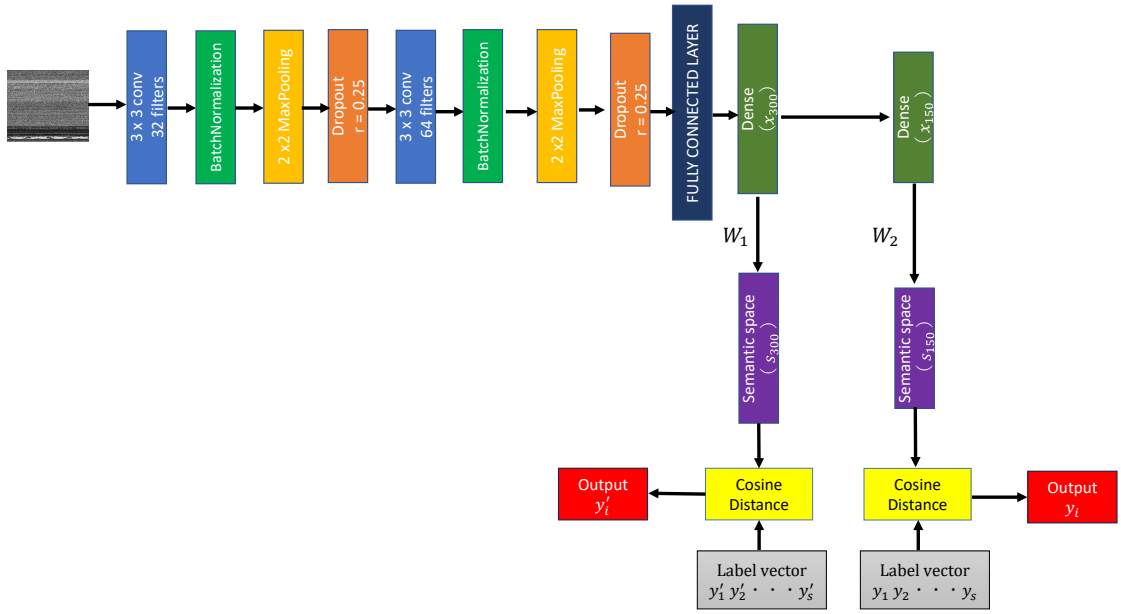


図 4.5: ZSL-SLCNN のトレーニングフェーズ

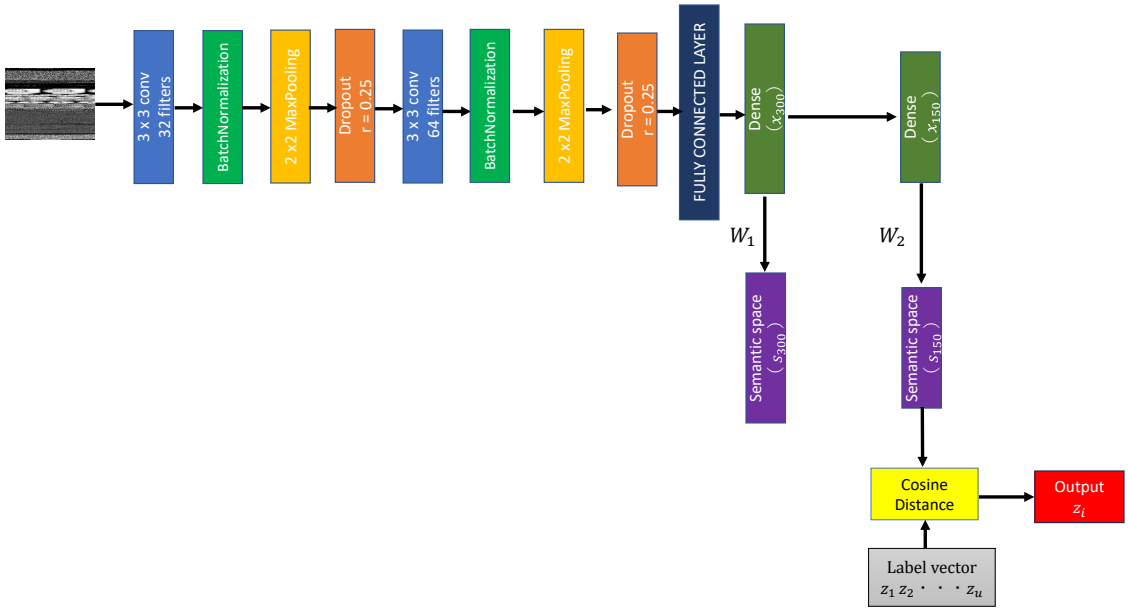


図 4.6: ZSL-SLCNN のテストフェーズ

4.5　実験結果

4.5.1　データセット

学習に関して, 本章では, 25 のファミリの 9,339 のマルウェア画像サンプルからなる Malimg データセット [46] を採用した．テストに関しては，二つのデータセットを利用する．一つ目は

[137] が提供する 5 つのマルウェアファミリと合計 8,940 のサンプルサイズを持つマルウェアデータセット．すべてのマルウェアファイルは，VirusShare[*1] と Malicia Project[138] から収集したものである．二つ目は [139] が提供する 8 つのマルウェアファミリと合計 1,803 のサンプルである．表 2.1 を基に画像変換を行う．各ラベルのマルウェア数を表 4.2 及び表 4.3 に示す．

表 4.2: マルウェアデータセット [137]

ラベル	ファミリ名	サンプル数	比率 (%)
0	Locker	300	3.36
1	Mediyes	1450	16.22
2	Winwebsec	4400	49.22
3	Zbot	2100	23.49
4	Zeroaccess	690	7.71

表 4.3: マルウェアデータセット [139]

ラベル	ファミリ名	サンプル数	比率 (%)
0	LockScreen	123	6.82
1	Reveton	522	28.95
2	TeslaCrypt	167	9.26
3	WannaCry	491	27.23
4	Win32_Crypt	146	8.10
5	Win32_Cryptor	123	6.82
6	Win32_FileCoder	27	1.50
7	Win32_Ransom	204	11.32

4.5.2　既知ラベルを持つマルウェアの分類結果

表 4.4 に示す結果で λ 値を微調整する．λ を 0.8 に設定したとき，99.47% の精度で最高の性能を得た．

図 4.7 に示すように，提案手法の二つのマッピング層を用いた学習は，一つのマッピング層よりも安定している．両マッピングレイヤーの相互作用により，パフォーマンスも大幅に向上している．通常の CNN は 46 エポック目まで安定していないものの、提案モデルは 32 エポッ

[*1] https://virusshare.com

表 4.4: 微調整時のパフォーマンス λ.

λ	λ=0.1	λ=0.2	λ=0.3	λ=0.4	λ=0.5	λ=0.6	λ=0.7	**λ=0.8**	λ=0.9
正解率 (%)	99.00	99.29	98.82	99.14	99.36	98.89	99.14	**99.47**	99.36

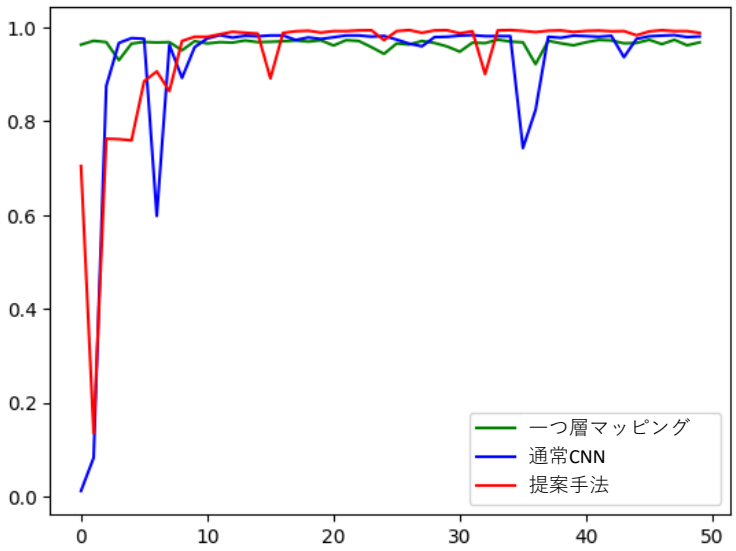


図 4.7: 性能比較

ク以降安定した．各モデルの正解率を表 4.5 に示す．提案手法は，一つのマッピング層からなるモデルより 2.04%，通常 CNN より 1.07% 改善する．表 4.6 は提案方法と先行研究を比較する結果である．軽量でありながら，既存研究より高い性能が得られた．

表 4.5: Malimg データセットでの性能比較

手法	**正解率 (%)**
一つ層マッピング	97.43
CNN	98.40
提案手法	**99.47**

4.5.3　未知ラベルを持つマルウェアの分類結果

Malimg データセットで学習した重みを利用して，未知のマルウェアファミリに属する各画像をマッピングする．次に，得られた特徴量を各ラベルベクトルと比較する．その結果，

表 4.6: Malimg データセットにおける他のマルウェア分類モデルとの比較.

モデル	正解率 (%)	パラメータ数 (M)
GIST feature + kNN [46]	97.18	-
Lightweight CNNs [34]	97.49	**0.83**
Attention-based Cross-model CNN [126]	99.09	-
Stacked Ensemble SE-AGM [127]	99.30	-
Global-Local Attention + Vision Transformer [128]	99.32	307
Stacked Depthwise Separable Convolutions + Attention Mechanism [129]	99.33	-
提案手法	**99.47**	5.09

どのサンプルがどのラベルと最も高い類似性を持つかを決定するために，k=1 の kNN とコサイン距離を利用する．CLIP[114] で得られた結果を，バックボーン ResNet50，ResNet101，Vit-B/16，Vit-L/14 と比較する．

二つのデータセットより得られた結果から見ると，一つのマッピング層を持つ CLIP モデルは，中規模から大規模のネットワークを使用しているにもかかわらず，期待される結果が得られないことが分かった． 表 4.7 では，分類結果が 0% のファミリが 二つあるが，大きいサイズの Vision Transformer を使用すると，一つのファミリが改善された．ただし，得られる正解率は依然として 30% 未満である． ResNet101 および Vit-B/16 を使用した CLIP は，それぞれ二つのファミリ Zeroaccess および Mediyes に対しては，提案方法よりも良い結果を得たが，その他のファミリと提案手法の性能差はかなり大きい．CLIP で用いたモデルの中の Vit-B/16 は，一番高い 26.38% 正解率を得た．提案手法は各ファミリの分類精度は 10% 以上で，平均は CLIP[114]Vit-B/16 より 9.28% 上回った．未知のデータセット [139] でも同じ結果が得られた（表 4.8）．CLIP で用いたモデルの中に Vit-B/16 は一番高い 18.75% の正解率である．しかし，各ファイルの分類精度の差は大きく，分類できないファミリは 3 つある．提案手法は一つのファミリ（Reveton）は 10% しか分類できない．平均は CLIP[114] Vit-B/16 と比べ 6.52% 上回った.

表 4.7: 未知のマルウェアデータセット [137] における他のマルウェア分類モデルとの比較.

マルウェアファミリ	CLIP[114] ResNet50	CLIP[114] ResNet101	CLIP[114] Vit-B/16	CLIP[114] Vit-L/14	提案手法
Locker	0.00	0.00	0.00	0.00	**31.42**
Mediyes	49.59	0.76	**91.93**	70.07	64.68
Winwebsec	29.98	0.00	0.00	5.25	**34.75**
Zeroaccess	7.83	**96.09**	29.42	1.76	32.97
Zbot	0.00	11.57	10.52	4.20	**14.48**
平均	17.48	21.68	26.38	16.26	**35.66**

表 4.8: 未知のマルウェアデータセット [139] における他のマルウェア分類モデルとの比較.

マルウェアファミリ	CLIP[114] ResNet50	CLIP[114] ResNet101	CLIP[114] Vit-B/16	CLIP[114] Vit-L/14	提案手法
LockScreen	**47.15**	0.00	17.07	0.00	38.33
Reveton	0.00	1.53	0.19	0.00	**2.85**
TeslaCrypt	80.84	**95.21**	91.02	48.50	45.76
WannaCry	0.61	0.00	0.00	0.00	**18.81**
Win32_Crypt	0.00	0.68	0.00	0.00	**19.32**
Win32_Cryptor	0.00	0.00	0.00	0.00	**16.35**
Win32_FileCoder	0.00	0.00	40.74	**51.85**	46.93
Win32_Ransom	0.49	0.49	0.98	5.39	**13.82**
平均	16.14	12.24	18.75	13.22	**25.27**

4.6 第 4 章のまとめ

本章では，未知マルウェアを分類するため新たな方法を提案し，軽量でありながら既存研究の手法より性能が上回ったことを，いくつかのデータセットに対して確認した．シンプルな畳み込みネットワークで非力な GPU でも適用できる未知のマルウェアを分類タスクに期待できる新たな方法を示唆できた．

本章は 6 節から構成された．4.1 節では未知のマルウェア分類についての既存研究を紹介した．フューショット学習が数多く研究されたが，ゼロショット学習を用いる研究は限られることが分かった．4.2 節と 4.3 節は，画像特徴を抽出するシンプル畳み込みネットワーク及びラベルベクトルを生成するための Fasttext を紹介し，それを 4.4 節での提案手法へ活かした．提案手法は新たな二層マッピングを提案し，それらの層を関連性を高めるため総合的損失関数を提案した．4.5 節では，既知のマルウェア分類また未知のマルウェア分類においても提案手法の有効性を示した．

既知ラベルによる分類は望ましい結果をもたらすが，未知のマルウェアにはへの適用は困難である．本章では，単語埋め込みと画像表現に基づく新しいゼロショット分類を提案した．提案手法は，単語や画像に対して手動で定義された意味的特徴や視覚的特徴を必要としない．マルウェアは画像に変換され，ラベルに対応する意味的な単語ベクトルに近くなるようにマッピングされる．画像ベースのマルウェアから抽出された局所情報に焦点を当てたシンプルな CNN ネットワークを使用し，FastText から事前に学習された単語埋め込みを利用した．

これまでの研究では，一つの画像特徴空間をラベル埋め込みにマッピングすることがほとんどで，オーバーフィッティングを招きやすかった．この問題を克服するために，二つの層によるマッピングを提案し，多重化により妥当性を高め，損失関数を最小化することによって，これら二つの空間の相互作用を最大化した．画像の特徴を処理する際には，損失等化と前の二つの高密度レイヤーから，より多くの情報が交換される．実験結果は，二つのマッピング層を使用することで，手頃なパラメータの数で，既知ラベルを持つマルウェアの分類及び未知ラベルを持つマルウェアの分類の両方でより効果的であることを複数のデータセットにおいて示していた．

ゼロショット学習は，データの特異性のため，全てのマルウェア処理に広く適用されてはいない．本稿では，この手法の高い適用可能性を示した．ゼロショット学習の応用は，これまで出現したことのないマルウェアの新ファミリの検出においてセキュリティ分野を促進し，専門家が詳細な分析にかかる時間をいくらか短縮するのに役立つ．しかし，性能にはまだ改善の余地がある．これはゼロショット学習が他の N ショット学習と比較して直面する共通の問題でもある．

今後の研究は製品化後の実情に即して，他の優れた，しかしシンプルな CNN モデルを適用

し低コストを確保する．さらに，ウィキペディアのような一般的なデータではなく，マルウェアに特化したデータでコーパスを構築することも考えられる．

第 5 章

結論と展望

5.1 結論

本研究では，IoT デバイスなどで増加しているマルウェア被害に対処が可能な，深層学習によって，軽量でありながら精度が得られるような，適切かつシンプルなネットワークを構築した．第 1 章では，研究背景及び研究目的を述べ．この章で指摘した三つの問題点：対策が不十分，良い手段が適用できない端末用のマルウェア，及び出現してまもないマルウェアへの対策については，以下の各章で対応することとした．

第 2 章及び第 3 章では，教師あり学習を対象とし，畳み込みネットワークを用いたモデル，畳み込みネットワークを用いないモデルの二つのマルウェア分類モデルをそれぞれ提案した．CNN を用いたモデルとしては，局所的特徴とアテンションを活かしたグローバル特徴を組み合わせた総合的特徴を持つ CNN-AVEA を提案し，他の CNN ベースモデルと比較し，モデル及び学習可能パラメータは少ないにも関わらず，複数のデータセットにおいて優れた結果を得た．CNN を用いないモデルとしては，新たに提案された MLP-mixer が，MLP のみであるがパッチ面及びチャンネル面両方を重視することによって，CNN に匹敵する性能に辿り着いた．本研究は MLP-mixer にオートエンコーダを導入することによって，シンプルなアーキテクチャかつ少数のパラメータでも CNN フリーのモデルと比較して，正解率が高い結果を得た．既存研究は CNN より性能的に下回った結果であるが，提案したモデルでは，大きく改善された．

第 4 章では新しいマルウェアファミリが続々と誕生していることを懸念して，サンプル数が少ない，もしくは無い状態でも学習可能なゼロショット学習を取り入れた未知のマルウェア分類手法を提案した．

本研究では，軽量でありながら妥当性の向上に焦点を当てたシンプルなモデルを提案した．マッピング手法について，従来は一つのレイヤーによる巨大なモデルで処理を行っていたところを，提案手法では軽量かつシンプルな二つのレイヤーで制御することによって，大きな精度

の改善が見られた.

5.2　研究倫理

本研究では自然言語処理，画像処理及び機械学習ライブラリなどの実験環境は，一般的に公開されており，実装が容易なものである．本研究では，公開されていないデータやライブラリは使用していない．したがって，結果の再現性は高いものと考えられ，様々な環境で使用可能である．今回対象とする，非力な GPU を使用可能なデバイスの例としては，スマートカメラとビジョンセンサー，ロボティクス・プラットフォームなどがある．また，CPU のみからなるデバイスの例としては，Raspberry Pi, microcontroller boards, 環境センサーがある．

5.3　研究の限界

本研究は様々な公開されたデータセットを用いて実験を行ったが，この検証は必ずしも現実の全実行ファイルの傾向を反映しているとは限らない．また最新の難読化技術はマルウェアの構造が変わっていくに連れて違ってくる。特に，パックタイプも 10 種類以上があるため，その結果，ファミリの数も多くなってしまう．既知分類の負担は大きくなってしまう．未知分類は対応可能であるが，実用的な精度までたどり着かない.

5.4　課題と展望

深層学習を用いた画像に基づくのモデルはマルウェアの中身や挙動を解析する必要がないことが利点であるが，反面この手法を回避する研究も盛んである．敵対的攻撃 [140] に対するこれらの提案手法のロバスト性を検討することが，今後も必要になってくる.

既知の様々な環境のシステムに導入するために，本研究で提案した手法がどこまでの範囲を受け持つかの検討が必要である．つまり，システムの既知な機能との適合性を維持することが今後の課題と展望になる．また，ゼロショット学習がマルウェア分類タスクに適用できる可能性を示したが，今後より高い精度を得るためには，モデルの改善も必要となる.

謝辞

本論文は防衛大学校理工学研究科後期課程において, 多くの方々のご指導とご協力をいただき, 本研究を行うことができました．本論文は, 多忙極まる状況下で, 指導してくださった佐藤 准教授, 久保 准教授及び松木 俊貴 助教の先生の方々のもとで, 研究成果をまとめたものです.

佐藤 准教授からは, 研究活動において様々な助言をいただくとともに, 知能情報分野に留まらず, ネットワークやセキュリティについても学ぶことができました.

久保 准教授からは, ゼミ内での議論を通じ論理的思考力を, 論文執筆を通じ的確に第三者に伝わる言語表現力や図解化能力を学ぶことができました．深く感謝を申し上げます.

また, 研究室のメンバーである航空自衛官の安井 大翼 一等空尉, 陸上自衛官の小鹿 凌 二等空尉, グェン・アイン・ナム学生, ブイ・ドク・ヴェト学生には, 公私共に多くの助言とサポートをいただきました. 厚く御礼申し上げます.

10 年間日本にいる間，いつも優しく見守ってくださったホストファミリのお父さん，お母さん，福井ファミリ，石川ファミリに感無量です.

最後に, 筆者の研究生活を支えてくれた, 茶道で色々慰めて下さった村瀬社中及び学校の職員茶道部を感謝申し上げます．村瀬先生は私のために裏千家の講師の資格まで取って下さいした．色々体験させて頂き日本の心をより深く理解ができました．大宗匠様，村瀬先生の恩に報いるため，日本の素晴らしい茶道の文化を幅広くベトナムで普及したく存じます.

この研究成果をもちましてこれから職場の部隊に戻り, 活かしたいと所存でございます．いつも気にかけてくれた母, そして支えてくださった全ての方に感謝申し上げます.

参考文献

[1] Github - screetsec/thefatrat: Thefatrat a massive exploiting tool. https://github.com/screetsec/TheFatRat, .

[2] Github - tor-0/arbitrium-rat: Arbitrium is a cross-platform, fully undetectable remote access trojan, to control android, windows and linux and doesn't require any firewall exceptions or port forwarding rules. https://github.com/ToR-0/Arbitrium-RAT, .

[3] Github - lief-project/lief: Lief - library to instrument executable formats. https://github.com/lief-project/LIEF, .

[4] Cybersecurity trends: Looking over the horizon — mckinsey. https://www.mckinsey.com/capabilities/risk-and-resilience/our-insights/cybersecurity/cybersecurity-trends-looking-over-the-horizon.

[5] Malware statistics & trends report — av-test. https://www.av-test.org/en/statistics/malware.

[6] 10.5 trillion reasons why we need a united response to cyber risk. https://www.forbes.com/sites/forbestechcouncil/2023/02/22/105-trillion-reasons-why-we-need-a-united-response-to-cyber-risk.

[7] Chandni Raghuraman, Sandhya Suresh, Suraj Shivshankar, and Radhika Chapaneri. Static and dynamic malware analysis using machine learning. In *First International Conference on Sustainable Technologies for Computational Intelligence: Proceedings of ICTSCI 2019*, pages 793–806. Springer, 2020.

[8] Hao Sun, Xiaofeng Wang, Rajkumar Buyya, and Jinshu Su. Cloudeyes: Cloud-based malware detection with reversible sketch for resource-constrained internet of things (iot) devices. *Software: Practice and Experience*, 47(3):421–441, 2017.

[9] Kshitiz Aryal, Maanak Gupta, and Mahmoud Abdelsalam. A survey on adversarial attacks for malware analysis. *arXiv preprint arXiv:2111.08223*, 2021.

[10] Ping Wang and Yu-Shih Wang. Malware behavioural detection and vaccine development by using a support vector model classifier. *Journal of Computer and System Sciences*, 81

(6):1012–1026, 2015.

[11] James B Fraley and Marco Figueroa. Polymorphic malware detection using topological feature extraction with data mining. In *SoutheastCon 2016*, pages 1–7. IEEE, 2016.

[12] James Scott. Signature based malware detection is dead. *Institute for Critical Infrastructure Technology*, 2017.

[13] Andreas Moser, Christopher Kruegel, and Engin Kirda. Limits of static analysis for malware detection. In *Twenty-third annual computer security applications conference (ACSAC 2007)*, pages 421–430. IEEE, 2007.

[14] Manuel Egele, Theodoor Scholte, Engin Kirda, and Christopher Kruegel. A survey on automated dynamic malware-analysis techniques and tools. *ACM computing surveys (CSUR)*, 44(2):1–42, 2008.

[15] Ilsun You and Kangbin Yim. Malware obfuscation techniques: A brief survey. In *2010 International conference on broadband, wireless computing, communication and applications*, pages 297–300. IEEE, 2010.

[16] Jyoti Landage and MP Wankhade. Malware and malware detection techniques: A survey. *International Journal of Engineering Research*, 2(12):61–68, 2013.

[17] Cheng Wang, Jianmin Pang, Rongcai Zhao, Wen Fu, and Xiaoxian Liu. Malware detection based on suspicious behavior identification. In *2009 First International Workshop on Education Technology and Computer Science*, volume 2, pages 198–202. IEEE, 2009.

[18] PV Shijo and AJPCS Salim. Integrated static and dynamic analysis for malware detection. *Procedia Computer Science*, 46:804–811, 2015.

[19] Ronghua Tian, Rafiqul Islam, Lynn Batten, and Steve Versteeg. Differentiating malware from cleanware using behavioural analysis. In *2010 5th international conference on malicious and unwanted software*, pages 23–30. Ieee, 2010.

[20] Rafiqul Islam, Ronghua Tian, Lynn M Batten, and Steve Versteeg. Classification of malware based on integrated static and dynamic features. *Journal of Network and Computer Applications*, 36(2):646–656, 2013.

[21] Tobias Wüchner, Martín Ochoa, and Alexander Pretschner. Robust and effective malware detection through quantitative data flow graph metrics. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 12th International Conference, DIMVA 2015, Milan, Italy, July 9-10, 2015, Proceedings 12*, pages 98–118. Springer, 2015.

[22] Peyman Khodamoradi, Mahmood Fazlali, Farhad Mardukhi, and Masoud Nosrati. Heuristic metamorphic malware detection based on statistics of assembly instructions using classification algorithms. In *2015 18th CSI International Symposium on Computer Architecture and Digital Systems (CADS)*, pages 1–6. IEEE, 2015.

[23] Edward Raff and Charles Nicholas. An alternative to ncd for large sequences, lempel-ziv jaccard distance. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1007–1015, 2017.

[24] Tuan Van Dao, Hiroshi Sato, Masao Kubo, and Yasuhiro Nakamura. Malware classification using low-level characteristics. *International Journal of Computer Theory and Engineering*, 15(3):111–116, 2023.

[25] Weijie Han, Jingfeng Xue, Yong Wang, Lu Huang, Zixiao Kong, and Limin Mao. Maldae: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics. *computers & security*, 83:208–233, 2019.

[26] William Fleshman, Edward Raff, Richard Zak, Mark McLean, and Charles Nicholas. Static malware detection & subterfuge: Quantifying the robustness of machine learning and current anti-virus. In *2018 13th International Conference on Malicious and Unwanted Software (MALWARE)*, pages 1–10. IEEE, 2018.

[27] Trung Kien Tran and Hiroshi Sato. Nlp-based approaches for malware classification from api sequences. In *2017 21st Asia Pacific Symposium on Intelligent and Evolutionary Systems (IES)*, pages 101–105. IEEE, 2017.

[28] Nur Adibah Rosli, Warusia Yassin, MA Faizal, and Siti Rahayu Selamat. Clustering analysis for malware behavior detection using registry data. *International Journal of Advanced Computer Science and Applications*, 10(12), 2019.

[29] Mcafee_wp_appcontrol-good-bad-unknown.pdf. `http://techdata.ca/(S(2dqagm55kzndh4553gbxdl45))/mcafee/files/MCAFEE_wp_appcontrol-good-bad-unknown.pdf`.

[30] Nannan Xie, Xing Wang, Wei Wang, and Jiqiang Liu. Fingerprinting android malware families. *Frontiers of Computer Science*, 13:637–646, 2019.

[31] Gregory Conti, Erik Dean, Matthew Sinda, and Benjamin Sangster. Visual reverse engineering of binary and data files. In *International Workshop on Visualization for Computer Security*, pages 1–17. Springer, 2008.

[32] Tao Ban, Ryoichi Isawa, Shanqing Guo, Daisuke Inoue, and Koji Nakao. Efficient malware packer identification using support vector machines with spectrum kernel. In *2013 Eighth Asia Joint Conference on Information Security*, pages 69–76. IEEE, 2013.

[33] Hamad Naeem, Bing Guo, Farhan Ullah, and Muhammad Rashid Naeem. A cross-platform malware variant classification based on image representation. *KSII Transactions on Internet & Information Systems*, 13(7), 2019.

[34] S Abijah Roseline, G Hari, S Geetha, and R Krishnamurthy. Vision-based malware detection and classification using lightweight deep learning paradigm. In *Computer Vision*

and Image Processing: 4th International Conference, CVIP 2019, Jaipur, India, September 27–29, 2019, Revised Selected Papers, Part II 4, pages 62–73. Springer, 2020.

[35] Erik Bochinski, Tobias Senst, and Thomas Sikora. Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. In *2017 IEEE international conference on image processing (ICIP)*, pages 3924–3928. IEEE, 2017.

[36] Jeyaprakash Hemalatha, S Abijah Roseline, Subbiah Geetha, Seifedine Kadry, and Robertas Damaševičius. An efficient densenet-based deep learning model for malware detection. *Entropy*, 23(3):344, 2021.

[37] Sunoh Choi, Jangseong Bae, Changki Lee, Youngsoo Kim, and Jonghyun Kim. Attention-based automated feature extraction for malware analysis. *Sensors*, 20(10):2893, 2020.

[38] Hiromu Yakura, Shinnosuke Shinozaki, Reon Nishimura, Yoshihiro Oyama, and Jun Sakuma. Neural malware analysis with attention mechanism. *Computers & Security*, 87: 101592, 2019.

[39] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[40] Jongkwan Lee and Jongdeog Lee. A classification system for visualized malware based on multiple autoencoder models. *IEEE Access*, 9:144786–144795, 2021.

[41] Olorunjube James Falana, Adesina Simon Sodiya, Saidat Adebukola Onashoga, and Biodun Surajudeen Badmus. Mal-detect: An intelligent visualization approach for malware detection. *Journal of King Saud University-Computer and Information Sciences*, 34(5): 1968–1983, 2022.

[42] Dashun Zheng, Rongsheng Wang, Yaofei Duan, Patrick Cheong-Iao Pang, and Tao Tan. Focus-rcnet: a lightweight recyclable waste classification algorithm based on focus and knowledge distillation. *Visual Computing for Industry, Biomedicine, and Art*, 6(1):19, 2023.

[43] 2023 sonicwall cyber threat report — sonicwall. `https://www.sonicwall.com/2023-cyber-threat-report/`.

[44] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *science*, 313(5786):504–507, 2006.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Lakshmanan Nataraj, Sreejith Karthikeyan, Gregoire Jacob, and Bangalore S Manjunath.

Malware images: visualization and automatic classification. In *Proceedings of the 8th international symposium on visualization for cyber security*, pages 1–7, 2011.

[47] Hamad Naeem, Bing Guo, Muhammad Rashid Naeem, Farhan Ullah, Hamza Aldabbas, and Muhammad Sufyan Javed. Identification of malicious code variants based on image visualization. *Computers & Electrical Engineering*, 76:225–237, 2019.

[48] Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Antonio Theophilo, Fabio Ramos, and Paulo de Geus. Malicious software classification using vgg16 deep neural network' s bottleneck features. In *Information Technology-New Generations: 15th International Conference on Information Technology*, pages 51–59. Springer, 2018.

[49] Mazhar Javed Awan, Osama Ahmed Masood, Mazin Abed Mohammed, Awais Yasin, Azlan Mohd Zain, Robertas Damaševičius, and Karrar Hameed Abdulkareem. Image-based malware classification using vgg19 network and spatial convolutional attention. *Electronics*, 10(19):2444, 2021.

[50] WK Wong, Filbert H Juwono, and Catur Apriono. Vision-based malware detection: A transfer learning approach using optimal ecoc-svm configuration. *IEEE Access*, 9:159262–159270, 2021.

[51] Aykut Çayır, Uğur Ünal, and Hasan Dağ. Random capsnet forest model for imbalanced malware type classification task. *Computers & Security*, 102:102133, 2021.

[52] Edmar Rezende, Guilherme Ruppert, Tiago Carvalho, Fabio Ramos, and Paulo De Geus. Malicious software classification using transfer learning of resnet-50 deep neural network. In *2017 16th IEEE international conference on machine learning and applications (ICMLA)*, pages 1011–1014. IEEE, 2017.

[53] Danish Vasan, Mamoun Alazab, Sobia Wassan, Babak Safaei, and Qin Zheng. Image-based malware classification using ensemble of cnn architectures (imcec). *Computers & Security*, 92:101748, 2020.

[54] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[55] V Anandhi, P Vinod, and Varun G Menon. Malware visualization and detection using densenets. *Personal and Ubiquitous Computing*, pages 1–17, 2021.

[56] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[57] Maryam Nisa, Jamal Hussain Shah, Shansa Kanwal, Mudassar Raza, Muhammad Attique Khan, Robertas Damaševičius, and Tomas Blažauskas. Hybrid malware classifica-

tion method using segmentation-based fractal texture analysis and deep convolution neural network features. *Applied Sciences*, 10(14):4966, 2020.

[58] Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.

[59] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[60] Roland Burks, Kazi Aminul Islam, Yan Lu, and Jiang Li. Data augmentation with generative models for improved malware detection: A comparative study. In *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0660–0665. IEEE, 2019.

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

[62] Xin Ma, Shize Guo, Haiying Li, Zhisong Pan, Junyang Qiu, Yu Ding, and Feiqiong Chen. How to make attention mechanisms more practical in malware classification. *IEEE Access*, 7:155270–155280, 2019.

[63] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[64] Quan Le, Oisín Boydell, Brian Mac Namee, and Mark Scanlon. Deep learning at the shallow end: Malware classification for non-domain experts. *Digital Investigation*, 26: S118–S126, 2018.

[65] Sang Ni, Quan Qian, and Rui Zhang. Malware identification using visualization images and deep learning. *Computers & Security*, 77:871–885, 2018.

[66] Daniel Gibert, Carles Mateu, Jordi Planes, and Ramon Vicens. Using convolutional neural networks for classification of malware represented as images. *Journal of Computer Virology and Hacking Techniques*, 15:15–28, 2019.

[67] Miguel Nicolau, James McDermott, et al. Learning neural representations for network anomaly detection. *IEEE transactions on cybernetics*, 49(8):3074–3087, 2018.

[68] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[69] Solomon Kullback. Kullback-leibler divergence, 1951.

[70] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[71] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention

for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998.

[72] Maurizio Corbetta and Gordon L Shulman. Control of goal-directed and stimulus-driven attention in the brain. *Nature reviews neuroscience*, 3(3):201–215, 2002.

[73] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. In *International conference on machine learning*, pages 7354–7363. PMLR, 2019.

[74] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Sca-cnn: Spatial and channel-wise attention in convolutional networks for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5659–5667, 2017.

[75] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[76] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[77] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[78] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. Ieee, 2013.

[79] Royi Ronen, Marian Radu, Corina Feuerstein, Elad Yom-Tov, and Mansour Ahmadi. Microsoft malware classification challenge. *arXiv preprint arXiv:1802.10135*, 2018.

[80] Ahmet Selman Bozkir, Ahmet Ogulcan Cankaya, and Murat Aydos. Utilization and comparision of convolutional neural networks in malware recognition. In *2019 27th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4. IEEE, 2019.

[81] S Abijah Roseline, S Geetha, Seifedine Kadry, and Yunyoung Nam. Intelligent vision-based malware detection and classification using deep random forest paradigm. *IEEE Access*, 8: 206303–206324, 2020.

[82] Felan Carlo C Garcia and Felix P Muga II. Random forest for malware classification. *arXiv preprint arXiv:1609.07770*, 2016.

[83] Tuan Van Dao, Hiroshi Sato, and Masao Kubo. An attention mechanism for combination of cnn and vae for image-based malware classification. *IEEE Access*, 10:85127–85136, 2022.

[84] Sushil Kumar et al. Mcft-cnn: Malware classification with fine-tune convolution neural networks using traditional and transfer learning in internet of things. *Future Generation*

Computer Systems, 125:334–351, 2021.

[85] Pedro H Barros, Eduarda TC Chagas, Leonardo B Oliveira, Fabiane Queiroz, and Heitor S Ramos. Malware-smell: A zero-shot learning strategy for detecting zero-day vulnerabilities. *Computers & Security*, 120:102785, 2022.

[86] Sravani Yajamanam, Vikash Raja Samuel Selvin, Fabio Di Troia, and Mark Stamp. Deep learning versus gist descriptors for image-based malware classification. In *Icissp*, pages 553–561, 2018.

[87] Vinita Verma, Sunil K Muttoo, and VB Singh. Multiclass malware classification via first- and second-order texture statistics. *Computers & Security*, 97:101895, 2020.

[88] Stefanos Tsimenidis, Thomas Lagkas, and Konstantinos Rantos. Deep learning in iot intrusion detection. *Journal of network and systems management*, 30:1–40, 2022.

[89] Quoc-Dung Ngo, Huy-Trung Nguyen, Van-Hoang Le, and Doan-Hieu Nguyen. A survey of iot malware and detection methods based on static features. *ICT Express*, 6(4):280–286, 2020.

[90] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.

[91] Ho-myung Kim and Kyung-ho Lee. Iiot malware detection using edge computing and deep learning for cybersecurity in smart factories. *Applied Sciences*, 12(15):7679, 2022.

[92] Yuxin Ding, Xiao Zhang, Jieke Hu, and Wenting Xu. Android malware detection method based on bytecode image. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–10, 2020.

[93] Itzhak Fogel and Dov Sagi. Gabor filters as texture discriminator. *Biological cybernetics*, 61(2):103–113, 1989.

[94] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.

[95] Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42:145–175, 2001.

[96] Pauline C Ng and Steven Henikoff. Sift: Predicting amino acid changes that affect protein function. *Nucleic acids research*, 31(13):3812–3814, 2003.

[97] Tran The Son, Chando Lee, Hoa Le-Minh, Nauman Aslam, and Vuong Cong Dat. An enhancement for image-based malware classification using machine learning with low dimension normalized input images. *Journal of Information Security and Applications*, 69:

103308, 2022.

[98] Duc-Ly Vu, Trong-Kha Nguyen, Tam V Nguyen, Tu N Nguyen, Fabio Massacci, and Phu H Phung. Hit4mal: Hybrid image transformation for malware classification. *Transactions on Emerging Telecommunications Technologies*, 31(11):e3789, 2020.

[99] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[100] Barath Narayanan Narayanan, Ouboti Djaneye-Boundjou, and Temesguen M Kebede. Performance analysis of machine learning and pattern recognition algorithms for malware classification. In *2016 IEEE national aerospace and electronics conference (NAECON) and ohio innovation summit (OIS)*, pages 338–342. IEEE, 2016.

[101] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

[102] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[103] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[104] Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. Pay attention to mlps. *Advances in Neural Information Processing Systems*, 34:9204–9215, 2021.

[105] Konrad Rieck, Philipp Trinius, Carsten Willems, and Thorsten Holz. Automatic analysis of malware behavior using machine learning. *Journal of computer security*, 19(4):639–668, 2011.

[106] Baraa Tareq Hammad, Norziana Jamil, Ismail Taha Ahmed, Zuhaira Muhammad Zain, and Shakila Basheer. Robust malware family classification using effective features and classifiers. *Applied Sciences*, 12(15):7877, 2022.

[107] Changguang Wang, Ziqiu Zhao, Fangwei Wang, and Qingru Li. Msaam: A multiscale adaptive attention module for iot malware detection and family classification. *Security and Communication Networks*, 2022, 2022.

[108] Médéric Hurier, Guillermo Suarez-Tangil, Santanu Kumar Dash, Tegawendé F Bissyandé, Yves Le Traon, Jacques Klein, and Lorenzo Cavallaro. Euphony: Harmonious unification of cacophonous anti-virus vendor labels for android malware. In *2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)*, pages 425–435. IEEE, 2017.

[109] Silvia Sebastián and Juan Caballero. Avclass2: Massive malware tag extraction from av labels. In *Annual Computer Security Applications Conference*, pages 42–53, 2020.

[110] Sangwon Kim, Wookhyun Jung, KyungMin Lee, HyungGeun Oh, and Eui Tak Kim. Sumav: Fully automated malware labeling. *ICT Express*, 8(4):530–538, 2022.

[111] Wei-Chung Huang, Fabio Di Troia, and Mark Stamp. Robust hashing for image-based malware classification. In *ICETE (1)*, pages 617–625, 2018.

[112] Lahouari Ghouti and Muhammad Imam. Malware classification using compact image features and multiclass support vector machines. *IET Information Security*, 14(4):419–429, 2020.

[113] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. *Advances in neural information processing systems*, 26, 2013.

[114] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[115] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.

[116] Soravit Changpinyo, Wei-Lun Chao, and Fei Sha. Predicting visual exemplars of unseen classes for zero-shot learning. In *Proceedings of the IEEE international conference on computer vision*, pages 3476–3485, 2017.

[117] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. *Advances in neural information processing systems*, 26, 2013.

[118] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the IEEE international conference on computer vision*, pages 4247–4255, 2015.

[119] Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh Nguyen, Matthias Hein, and Bernt Schiele. Latent embeddings for zero-shot classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 69–77, 2016.

[120] Meng Ye and Yuhong Guo. Zero-shot classification with discriminative semantic representation learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7140–7148, 2017.

[121] Top trends in cyber security — cyber attacks trends — m-trends. `https://www.mandiant.com/m-trends`. (Accessed on 11/07/2023).

[122] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word

representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[123] Fengli Shen and Zhe-Ming Lu. A semantic similarity supervised autoencoder for zero-shot learning. *IEICE TRANSACTIONS on Information and Systems*, 103(6):1419–1422, 2020.

[124] Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. Zero-shot learning with semantic output codes. *Advances in neural information processing systems*, 22, 2009.

[125] Elyor Kodirov, Tao Xiang, and Shaogang Gong. Semantic autoencoder for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3174–3183, 2017.

[126] Jeongwoo Kim, Joon-Young Paik, and Eun-Sun Cho. Attention-based cross-modal cnn using non-disassembled files for malware classification. *IEEE Access*, 11:22889–22903, 2023.

[127] Pratyush Panda, Om Kumar CU, Suguna Marappan, Suresh Ma, and Deeksha Veesani Nandi. Transfer learning for image-based malware detection for iot. *Sensors*, 23(6):3253, 2023.

[128] Mohamad Mulham Belal and Divya Meena Sundaram. Global-local attention-based butterfly vision transformer for visualization-based malware classification. *IEEE Access*, 2023.

[129] Santosh K Smmarwar, Govind P Gupta, and Sanjay Kumar. Ai-empowered malware detection system for industrial internet of things. *Computers and Electrical Engineering*, 108: 108731, 2023.

[130] Tuan Van Dao, Hiroshi Sato, and Masao Kubo. Mlp-mixer-autoencoder: A lightweight ensemble architecture for malware classification. *Information*, 14(3):167, 2023.

[131] Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2021–2030, 2017.

[132] Yuhan Chai, Lei Du, Jing Qiu, Lihua Yin, and Zhihong Tian. Dynamic prototype network based on sample adaptation for few-shot malware detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(5):4754–4766, 2022.

[133] Mauro Conti, Shubham Khandhar, and P Vinod. A few-shot malware classification approach for unknown family recognition using malware feature visualization. *Computers & Security*, 122:102887, 2022.

[134] Kien Tran, Hiroshi Sato, and Masao Kubo. Mannware: A malware classification approach with a few samples using a memory augmented neural network. *Information*, 11(1):51, 2020.

[135] Faiza Babar Khan, Muhammad Hanif Durad, Asifullah Khan, Farrukh Aslam Khan, Saj-

jad Hussain Chauhdary, and Mohammed Alqarni. Detection of data scarce malware using one-shot learning with relation network. *IEEE Access*, 2023.

[136] Fnv hash. `http://www.isthe.com/chongo/tech/comp/fnv/index.html`. (Accessed on 11/22/2023).

[137] Anh Pham Tuan, An Tran Hung Phuong, Nguyen Vu Thanh, and Toan Nguyen Van. Malware detection pe-based analysis using deep learning algorithm dataset. figshare. dataset, 2018.

[138] Antonio Nappa, M Zubair Rafique, and Juan Caballero. The malicia dataset: identification and analysis of drive-by download operations. *International Journal of Information Security*, 14:15–33, 2015.

[139] Duc Thang Nguyen and Soojin Lee. Lightgbm-based ransomware detection using api call sequences. *International Journal of Advanced Computer Science and Applications*, 12(10), 2021.

[140] Bao Ngoc Vi, Huu Noi Nguyen, Ngoc Tran Nguyen, and Cao Truong Tran. Adversarial examples against image-based malware classification systems. In *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–5. IEEE, 2019.

発表実績

学術論文

1. Van Dao Tuan, Hiroshi Sato, and Masao Kubo. “An Attention Mechanism for Combination of CNN and VAE for Image-Based Malware Classification.” IEEE Access 10 (2022): 85127-85136.
2. Dao Tuan Van, Hiroshi Sato, and Masao Kubo. “MLP-Mixer-Autoencoder: A Lightweight Ensemble Architecture for Malware Classification.”Information 14.3 (2023): 167.
3. Tuan Van Dao, Hiroshi Sato, Masao Kubo, and Yasuhiro Nakamura, “Malware Classification Using Low-Level Characteristics.” International Journal of Computer Theory and Engineering 15.3 (2023): 111-116.

国際学会

1. Dao Van Tuan, Hiroshi Sato, Masao Kubo, “A Malware Classification Method by Combining Computer Vision and Deep Learning,” Vietnamese Academic Network in Japan (VANJ), December 4-5, 2021.
2. Tuan Van Dao, Hiroshi Sato, Masao Kubo, “ZSL-SLCNN: Zero-Shot Learning with Semantic Label CNN for Malware Classification,” 12th International Conference on Control, Automation and Information Sciences (ICCAIS), November 27-29, 2023.

国内学会

1. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “軽量化 DenseNet を用いたマルウェア分類,” 情報処理学会, 2022.
2. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “MLP-mixer-AE を 用いたマルウェア検知,” 第５０回画像電子学会年次大会, 2022.
3. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “MLP-mixer を用いたマルウェア分類,” 第 21 回情報科学技術フォーラム, 2022.
4. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, 中村　康弘, “静的特性アンサンブルを用いたマルウェアの分類,” コンピュータセキュリティシンポジウム, 2022.
5. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “静的と画像ベースの特徴量のアンサンブルを用いたランサムウェア検知,” 計測自動制御学会　システム・情報部門学術講演会, 2022.
6. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “VQ-VAE を用いたマルウェア検知,” 情報処理学会, 2023.
7. ダオ・ヴァン・トゥアン, 佐藤　浩, 久保　正男, “ゼロショット学習を用いたマルウェアの分類,” 計測自動制御学会　システム・情報部門学術講演会, 2023.

表彰

- VANJ2021 最優秀口頭発表賞
  ダオ・ヴァン・トゥアン，佐藤　浩, 久保　正男.
  Vietnamese Academic Network in Japan 2021 (VANJ 2021)
  2021 年 12 月 4 日 ~ 5 日
- ICCAIS2023 優秀論文賞
  ダオ・ヴァン・トゥアン，佐藤　浩, 久保　正男.
  12th International Conference on Control, Automation and Information Sciences 2023 (ICCAIS 2023)
  2023 年 11 月 27 日 ~ 29 日