Feature Extraction 中心による機械学習モデルの再構成

Y. Matsuda

29 March 2026

1　問題設定

機械学習モデルはしばしば、入力、特徴量、学習、出力という語で説明される。しかし、この説明は実用上は便利である一方、モデル内部で何が本質的に行われているかを十分に捉えているとは言い難い。

特に、以下のような代表的モデルを比較すると、そのことが明らかになる。

- CNN は画像上の局所パターンを抽出する
- GNN はグラフ構造上で情報を伝播する
- Transformer は要素間の関係を動的に生成する

これらは一見するとまったく異なる仕組みに見える。CNN は格子状データを扱い、GNN はノードとエッジから成る構造を扱い、Transformer は系列中の要素間関係を attention によって計算する。したがって、通常の教科書的な整理では、それぞれ別種のモデルとして個別に説明されることが多い。

しかし他方で、これら三者には共通する側面が存在する。いずれも入力をそのまま出力へ写像しているのではなく、入力から何らかの中間表現を生成し、その中間表現を通して予測を行っている。すなわち、三者はいずれも何らかの意味で「特徴」を生成している。

ここで問題となるのは、従来の用語法における曖昧さである。たとえば “Encoding” は入力の数値化を意味することもあれば、モデルに適した初期表現の設計を意味することもある。また “Feature Extraction” も、手設計特徴量の抽出を指す場合と、ニューラルネットワーク内部での表現生成を指す場合とが混在している。さらに “学習” という語も、重みの更新を指すのか、表現の獲得を指すのか、あるいは目的関数への適応を指すのかが必ずしも明確ではない。

この曖昧さは、単なる用語上の問題ではない。というのも、モデル比較を行う際に、

何が初期条件であり、何が中核計算であり、何が最適化の対象であるのか

が整理されないまま議論されると、CNN、GNN、Transformer の違いが表層的なアーキテクチャの違いとしてしか見えなくなるからである。

本稿の問題意識は、この点にある。すなわち、機械学習モデルを

- Encoding
- Feature Extraction
- Weight Learning
- Objective

という構成要素にいったん分解した上で、それらの関係を再整理する。そして、特に CNN、GNN、Transformer という異なるモデル群を比較しつつ、それらに共通する本質がどこにあるのかを明らかにすることを目的とする。

ただし、本稿は単なる要素分解にとどまらない。むしろ重要なのは、上記の各要素が対等に並ぶのではなく、ある中心概念の周囲に再配置されるべきではないか、という問いである。従来の説明では、Encoding、Feature、Learning、Objective はしばしば並列に置かれる。しかし実際には、それらのうち何か一つが中核であり、他はそれを支える補助的構成要素として理解した方が、モデル間の比較はより明瞭になる可能性がある。

この観点から、本稿では次の問いを立てる。

機械学習モデルの本質は、何を中心に記述されるべきか。

そして本稿の答えは、後に詳述するように、Feature Extraction を中心に据える立場である。すなわち、Encoding は Feature 生成の初期条件、Weight Learning は Feature 生成写像の変形、Objective は Feature の良さの評価として理解されるべきであり、モデル間の差異は主として Feature 生成の様式の差として捉えられる、というのが本稿の基本的立場である。

2 基本仮説

本稿の基本仮説は、以下の一文に要約される。

すべての機械学習モデルは、*Feature Extraction* 装置として理解できる。

この仮説は、一見すると自明に見えるかもしれない。実際、機械学習において特徴量の重要性は古くから知られており、ニューラルネットワークもまた中間表現を学習する仕組みとして説明されることが多い。しかし本稿で主張したいのは、そのような一般論ではない。ここで言う “Feature Extraction” は、単なる補助的工程ではなく、モデル全体の中核をなす概念である。

この立場に立つと、従来は並列的に語られてきた諸概念の役割が再編成される。

まず、Encoding は独立した本質的操作ではなく、Feature 生成のための初期条件の設定として理解される。入力をどの形式に写像するか、どの構造を事前に与えるか、どの表現空間から出発するかは、すべて後続の Feature 生成を可能にするための前提である。CNN におけるピクセル表現、GNN におけるグラフ構造、Transformer における埋め込みと位置情報は、それぞれ異なるが、いずれも Feature 生成の出発点として位置づけられる。

次に、Weight Learning は Feature 生成写像そのものの変形である。重みは単なる数値パラメータではない。CNN ではどの局所パターンを検出するか、GNN ではどの構造情報をどのように伝播するか、Transformer ではどの要素間関係を重要視するかを決定する。したがって学習とは、出力だけを合わせる操作ではなく、

どのような *Feature* を生成するかという規則の最適化

として理解される。

さらに、Objective は Feature 生成の外部に置かれた単なる損失ではない。むしろ Objective は、どのような Feature が良い Feature であるかを定義する評価基準である。分類問題であればクラス分離に有効な Feature、構造予測であれば局所・大域構造を反映する Feature、言語モデルであれば系列予測に資する Feature が、それぞれ Objective によって要請される。したがって Objective は、重み学習の方向を決めるだけでなく、Feature 空間の意味づけそのものに関与する。

以上より、Encoding、Weight Learning、Objective は互いに独立な並列要素ではなく、すべて Feature

Extraction を中心として再解釈される。すなわち、

- Encoding：Feature 生成の初期条件
- Weight Learning：Feature 生成写像の変形
- Objective：Feature 生成結果の評価基準

である。

この仮説の意義は、異なるモデルを同一の言語で比較できる点にある。CNN、GNN、Transformer は見かけ上まったく異なるが、本稿の立場では、それぞれ次のように理解される。

- CNN：固定された格子構造上で局所パターンを階層的に抽出する Feature 生成
- GNN：与えられたグラフ構造に沿って情報を伝播し統合する Feature 生成
- Transformer：要素間関係を動的に生成し、その関係に基づいて特徴を構成する Feature 生成

このように見ると、モデル間の差異は「Feature をどのように生成するか」の差異に帰着する。逆に言えば、モデル比較の本質はアーキテクチャ名の違いではなく、Feature 生成様式の違いとして整理されるべきである。

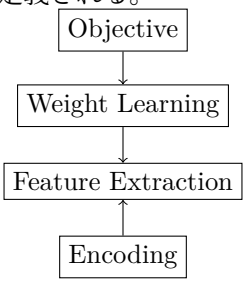
もちろん、この仮説はすべてを単純化しすぎる危険も持つ。たとえば Encoding が非常に強い意味を持つ GNN では、初期構造の与え方自体が結果を大きく左右する。また Transformer では、Feature 生成と構造生成がほぼ同一過程に融け合っているため、単純な段階分解はかえって誤解を招く可能性もある。しかし、それにもかかわらず Feature Extraction を中心に据える立場は、少なくとも以下の利点を持つ。

第一に、異種モデル間の比較軸を統一できる。第二に、学習、構造、表現、目的の役割分担を明確にできる。第三に、モデルの違いを単なる実装差ではなく、意味生成の差として捉え直すことができる。

したがって本稿では、この基本仮説を出発点として、各構成要素の再定義、各モデルの再解釈、そしてモデル横断的な比較へと進む。

3 再定義

本稿では、機械学習モデルを Feature Extraction を中心に再構成する立場をとる。この立場において、従来の各構成要素は以下のように再定義される。



Feature Extraction

Feature Extraction とは、入力データから意味的構造を生成する写像である。

一般に、入力 X に対して、

$$h = f_\theta(X)$$

と表されるとき、この写像 f_θ が Feature 生成機構に対応する。

ここで重要なのは、Feature が単なる「特徴量」ではなく、

「入力に内在する関係・構造・パターンを再構成した表現」

である点である。

CNN においては局所パターン、GNN においてはグラフ構造、Transformer においては要素間関係がそれぞれ Feature として生成される。

したがって Feature Extraction は、

「どのような構造を意味として取り出すかを決定する操作」

として位置づけられる。

Encoding

Encoding は、入力データを Feature 生成が可能な形式へ写像する操作である。

$$h^{(0)} = \phi(X)$$

ここで ϕ は Encoding 写像であり、初期表現を与える。

この役割はモデルにより大きく異なる：

- CNN：単なる数値表現（弱い Encoding）
- GNN：グラフ構造と初期特徴（強い Encoding）
- Transformer：埋め込みと位置情報（最小限の構造）

したがって Encoding は、

「*Feature* 生成の初期条件および表現空間の選択」

として理解される。

重要なのは、Encoding 自体がすでに一部の構造を固定する場合があることである（特に GNN）。

Weight Learning

Weight Learning は、Feature 生成写像 f_θ のパラメータ θ を最適化する過程である。

$$\theta^* = \arg\min_\theta \mathcal{L}(f_\theta(X), Y)$$

ここで重み θ は、Feature 生成の以下を決定する：

- どのパターンを強調するか（CNN）
- どの構造情報を伝播するか（GNN）
- どの関係を生成するか（Transformer）

したがって Weight Learning は、

「*Feature* 生成過程の変形および選択」

として解釈される。

すなわち、重みは単なる係数ではなく、Feature 生成ルールそのものを規定する。

Objective

Objective は、生成された Feature の良し悪しを評価する基準である。

$$\mathcal{L} = \mathcal{L}(f_\theta(X), Y)$$

このとき Objective は、単に出力誤差を測るだけでなく、

「どのような *Feature* が意味的に有効であるかを定義する」

役割を持つ。

例えば：

- 分類問題：クラスを分離可能な Feature
- 構造予測：局所・大域構造を反映する Feature
- 言語モデル：系列の予測可能性を高める Feature

したがって Objective は、

「*Feature* 空間における評価関数」

として理解される。

統合的関係

以上を統合すると、機械学習は以下の構造を持つ：

- Encoding：初期表現 $h^{(0)}$ を与える
- Feature Extraction：写像 f_θ により特徴を生成する
- Weight Learning：θ を最適化する
- Objective：Feature の良さを定義する

この関係は以下のようにまとめられる：

$$\theta^* = \arg\min_\theta \mathcal{L}(f_\theta(\phi(X)), Y)$$

ここで本質的なのは、

すべての要素が *Feature* 生成を中心に結びついている

ことである。

したがって本稿の立場では、機械学習は

「*Feature* 生成写像の設計と最適化」

として統一的に理解される。

4　モデルの再解釈

MNIST（CNN）

CNN は、格子状データ（画像）上において局所的なパターンを抽出し、それを階層的に統合する Feature 生成機構である。

入力画像を

$$X \in \mathbb{R}^{H \times W}$$

とする（MNIST では $H = W = 28$）。

Encoding　Encoding は、画像をピクセル値として表現することに対応する：

$$h^{(0)} = X$$

これは単なる数値化であり、構造的な制約はほとんど含まれない。すなわち、Encoding は

「*Feature* 生成のための最小限の初期条件」

として機能する。

Feature Extraction　CNN の各層における特徴生成は、畳み込み演算として定式化される。
カーネル $K^{(k)}$ を用いた第 k 層の出力は：

$$h_{i,j}^{(k+1)} = \sigma\left(\sum_{u,v} K_{u,v}^{(k)} \cdot h_{i+u,j+v}^{(k)}\right)$$

ここで、

- $K^{(k)}$：学習されるフィルタ（カーネル）
- σ：非線形関数

この演算は、局所領域におけるパターン検出として解釈される。
さらに、プーリング操作（例：最大値）により、

$$h_{i,j}^{(k+1)} = \max_{(u,v)\in\Omega} h_{i+u,j+v}^{(k)}$$

空間的な不変性が導入される。
この過程を重ねることで、

$$h^{(K)} = f_\theta(X)$$

が得られる。
ここで生成される Feature は、

「局所パターンの階層的統合」

として特徴づけられる。

Weight Learning　重み $\theta = \{K^{(k)}\}$ は、どの局所パターンを検出するかを決定する。
すなわち、

- エッジ
- 角
- ストローク構造

といった特徴を抽出するフィルタが学習される。
したがって Weight Learning は、

「局所 *Feature* 検出器の選択」

として解釈される。

Objective　最終的な出力は、全結合層を通じてクラス確率として与えられる：

$$\hat{y} = \mathrm{softmax}(W_{\mathrm{out}} \cdot h^{(K)})$$

目的関数は分類誤差であり：

$$\mathcal{L} = -\sum_{c} y_c \log \hat{y}_c$$

と定義される。

この Objective は、

「数字クラスを識別可能な *Feature* を生成する」

ことを要求する。

総合解釈　以上より CNN は、

「固定された空間上で、局所パターンを階層的に統合する *Feature* 生成機構」

として理解される。

ここで重要なのは、

- 構造（格子）は固定されている
- Feature は局所性に強く依存する
- 重みは検出すべきパターンを選択する

ことである。

したがって CNN は、

「構造固定・局所抽出型 *Feature Extraction*」

として位置づけられる。

- Encoding：弱い（単なるピクセル）
- Feature：CNN による局所構造抽出
- 特徴：Feature 依存型

GNN

GNN は、グラフ構造上に定義された関係に従って特徴を伝播・統合する Feature 生成機構である。

グラフを

$$G = (V, E)$$

とし、各ノード $v \in V$ に初期特徴ベクトル $h_v^{(0)}$ が与えられるとする。

GNN の基本操作は、層 k における以下の更新として定式化される：

$$m_v^{(k)} = \mathrm{AGG}\left(\{h_u^{(k)} \mid u \in \mathcal{N}(v)\}\right)$$

$$h_v^{(k+1)} = \sigma\left(W^{(k)} \cdot \mathrm{COMB}\left(h_v^{(k)}, m_v^{(k)}\right)\right)$$

ここで、

- $\mathcal{N}(v)$：ノード v の近傍集合
- AGG：集約関数（平均、和、最大など）
- COMB：自己特徴と集約特徴の結合
- $W^{(k)}$：学習される重み

- σ：非線形関数

—

この定式化において、GNN の Feature 生成は以下の構造を持つ：

- Encoding：グラフ構造 G および初期特徴 $h_v^{(0)}$
- Feature 生成：近傍に基づく反復的な情報統合
- Weight の役割：統合の強さおよび変換の形を決定

—

この過程を反復することで、ノード特徴は次第に広域情報を取り込み、

$$h_v^{(K)} = f_\theta(G, \{h_u^{(0)}\})$$

として表される。

—

したがって GNN は、

「構造に拘束された反復的 *Feature* 生成過程」

として理解される。

すなわち、Feature は自由に生成されるのではなく、グラフ構造という制約の下で局所的に拡張される。

—

さらに重要なのは、情報の到達範囲が層数 K に依存することである。

$h_v^{(K)}$ は K-hop 近傍に依存する

このことは、GNN における Feature が

「局所構造の反復的拡張」

として生成されることを意味する。

—

以上より、GNN は

「構造によって制約された情報伝播を通じて *Feature* を生成する機構」

として位置づけられる。

Transformer

Transformer は、入力系列に対して要素間の関係を動的に生成し、その関係に基づいて特徴を構成する Feature 生成機構である。

入力系列を

$$X = (x_1, x_2, \ldots, x_n)$$

とする。

Encoding　各トークンは埋め込みによりベクトルへ写像される：

$$h_i^{(0)} = E(x_i) + p_i$$

ここで、

- $E(x_i)$：トークン埋め込み
- p_i：位置エンコーディング

Encoding は系列構造を明示的に与えず、最小限の順序情報のみを付加する。
したがって Encoding は、

「関係生成のための初期状態の付与」

として理解される。

Feature Extraction（Self-Attention）　Transformer の中核は Self-Attention であり、各トークンは他のすべてのトークンとの関係を通じて更新される。
まず、Query・Key・Value を定義する：

$$q_i = W_Q h_i^{(k)}, \quad k_j = W_K h_j^{(k)}, \quad v_j = W_V h_j^{(k)}$$

Attention 重みは以下で与えられる：

$$\alpha_{ij} = \frac{\exp(q_i^\top k_j / \sqrt{d})}{\sum_{j'} \exp(q_i^\top k_{j'} / \sqrt{d})}$$

特徴更新は：

$$h_i^{(k+1)} = \sum_{j=1}^{n} \alpha_{ij} v_j$$

この操作は、

「要素間の関係を重みとして構成し、その関係に基づいて特徴を再構成する」

ことを意味する。
さらに Feed Forward Network により非線形変換が施される：

$$h_i^{(k+1)} = \mathrm{FFN}(h_i^{(k+1)})$$

この過程を繰り返すことで、

$$h_i^{(K)} = f_\theta(X)$$

が得られる。
生成される Feature は、

「全体関係に基づく動的構造」

である。

Weight Learning　重み $\theta = \{W_Q, W_K, W_V, W_{\mathrm{FFN}}\}$ は、

- どのトークン同士を関連付けるか
- どの関係を強調するか

を決定する。
したがって Weight Learning は、

「関係生成ルールの学習」

として解釈される。

Objective　言語モデルの場合、次トークン予測が目的となる：

$$\hat{y}_t = \mathrm{softmax}(W_{\mathrm{out}} \cdot h_t^{(K)})$$

$$\mathcal{L} = -\sum_t \log P(x_t \mid x_{<t})$$

この Objective は、

「系列構造を予測可能にする *Feature* を生成する」

ことを要求する。

総合解釈　以上より Transformer は、

「関係を動的に生成し、その関係に基づいて *Feature* を構成する機構」

として理解される。

ここで重要なのは：

- 構造は事前に与えられない
- Feature 生成と構造生成が一体化している
- 重みが関係そのものを定義する

したがって Transformer は、

「関係生成型 *Feature Extraction*」

として位置づけられる。

- Encoding：最小限
- Feature：attention による関係生成
- 特徴：Feature が構造を生成

5　総合結論

本稿では、機械学習モデルを Feature Extraction を中心として再構成した。

まず、CNN、GNN、Transformer といった異なるモデルを比較する際、従来の用語（Encoding、Feature、学習、Objective）が必ずしも一貫した意味で使われていないことを指摘した。この曖昧さを解消するため、本稿では各要素を再定義し、それらの関係を整理した。

その結果、以下の理解に到達する。

- Encoding は Feature 生成の初期条件である
- Weight Learning は Feature 生成写像の変形である
- Objective は Feature の良さを定義する評価基準である

したがって、これらの要素は互いに独立ではなく、すべて Feature Extraction を中心として統合される。

この立場に立つと、異なるモデルは次のように統一的に理解される：

- CNN：局所パターンを階層的に統合する Feature 生成
- GNN：構造に沿って情報を伝播する Feature 生成
- Transformer：関係を動的に生成する Feature 生成

すなわち、モデル間の本質的な差異は、アーキテクチャの違いではなく、

「*Feature* をどのように生成するか」

の違いに帰着する。

以上より、本稿の結論は以下のように要約される：

機械学習とは、*Feature* 生成写像の設計と最適化である。

以上より、機械学習は以下のように再定義される：

機械学習とは、*Feature Extraction* を中心に、
その初期条件・変形・評価を設計する過程である

すなわち、

- Encoding：開始点
- Weight Learning：変形
- Objective：評価

はすべて Feature Extraction に従属する。

モデル間の違いは、Feature Extraction の設計の違いとして統一的に理解される。

A　EpiGraph における Feature Extraction の具体例

本付録では、タンパク質構造に基づくエピトープ予測モデルである EpiGraph を例に取り、Feature Extraction 中心の枠組みで再解釈する。

A.1　問題設定

入力は抗原タンパク質の 3 次元構造であり、各アミノ酸残基に対して「抗体が結合する可能性（エピトープ性）」を予測する。

重要な点は、学習時に抗体の構造情報が明示的に与えられないことである。すなわち、問題は

「抗体との相互作用を直接観測せずに、結合可能性を推定する」

という性質を持つ。

A.2　Encoding

タンパク質構造はグラフとして表現される：

$$G = (V, E)$$

- ノード $v \in V$：アミノ酸残基
- エッジ $(u, v) \in E$：空間的近接関係（距離閾値など）

各ノードには初期特徴が与えられる：

$$h_v^{(0)} = \phi(v)$$

ここで $\phi(v)$ は以下を含む：

- アミノ酸タイプ
- 物理化学特性（電荷、疎水性など）
- 構造情報（溶媒アクセス性、二次構造）

したがって Encoding は、

「幾何構造と局所物性を組み込んだ初期条件の定義」

として理解される。

A.3 Feature Extraction

GNN により、各ノードの特徴は近傍構造に基づいて更新される：

$$m_v^{(k)} = \mathrm{AGG}\left(\{h_u^{(k)} \mid u \in \mathcal{N}(v)\}\right)$$

$$h_v^{(k+1)} = \sigma\left(W^{(k)} \cdot \mathrm{COMB}(h_v^{(k)}, m_v^{(k)})\right)$$

この反復により、

$$h_v^{(K)} = f_\theta(G, \{h_u^{(0)}\})$$

が得られる。

このとき生成される Feature は、

「局所構造・物性・空間配置の複合パターン」

であり、単一の属性ではなく構造的コンテキストに依存する。

A.4 Weight Learning

最終的な予測は各ノードに対して与えられる：

$$\hat{y}_v = \sigma(W_{\mathrm{out}} \cdot h_v^{(K)})$$

重み $\theta = \{W^{(k)}, W_{\mathrm{out}}\}$ は以下を学習する：

- どの構造パターンが重要か
- どの物性の組み合わせが寄与するか
- 情報伝播の範囲と強度

したがって Weight Learning は、

「*Feature* 生成過程における重要度の選択」

として解釈される。

A.5 Objective

教師信号は各残基のラベル $y_v \in \{0, 1\}$ であり、エピトープか否かを示す。

目的関数は通常、以下で与えられる：

$$\mathcal{L} = -\sum_{v \in V} [y_v \log \hat{y}_v + (1 - y_v) \log(1 - \hat{y}_v)]$$

この Objective は、

「エピトープらしい *Feature* を持つノードを識別する」

ことを要求する。

A.6 教師データの特殊性

EpiGraph において重要なのは、教師信号が以下の性質を持つことである：

- 抗体構造は入力に含まれない
- ラベルは抗体との相互作用の結果として与えられる

すなわち、

「原因（抗体）が観測されず、結果のみが与えられる」

という構造を持つ。

このためモデルは、

「抗体に依存しない普遍的な結合特徴」

を学習せざるを得ない。

これは以下を意味する：

- Feature は特定の相互作用ではなく「潜在的相互作用性」を表す
- 学習は直接的な対応付けではなく統計的パターン抽出となる

A.7 総合解釈

以上より、EpiGraph は以下として理解される：

「観測されない相互作用を、構造特徴から間接的に再構成する *Feature Extraction*」

すなわち、

- Encoding：構造と物性の初期化
- Feature：構造的コンテキストの生成
- Weight：重要パターンの選択
- Objective：結合可能性の識別

が統合された枠組みである。