

# Dynamic Routing Between Capsules

---

- Geoffrey E. Hinton (Google Brain)
- Sara Sabour (Google Brain)
- Nicholas Frosst (Google Brain)

## Abstract

---

カプセルとはニューロンのグループのことで、その活性ベクトルはオブジェクトまたはオブジェクトの一部などのエンティティの特定のタイプのインスタンス化パラメータを表す。活性ベクトルの長さはエンティティが存在する確率を、活性ベクトルの方向はインスタンス化パラメータを表す。あるレベルの活性カプセルは、より高いレベルのカプセルのインスタンス化パラメータに対して、変換行列を介して予測を行う。複数の予測が一致すると、より高いレベルのカプセルが活性化される。我々は、識別可能に訓練された多層カプセルシステムが、MNISTにおいて最先端の性能を達成し、重なり合った数字を認識する際に畳み込みネットワークよりもかなり優れていることを示す。これらの結果を達成するために、我々は合意メカニズムによる反復ルーティングを使用する。（低いレベルのカプセルは、より低いレベルのカプセルから来る予測を伴う大きなスカラー積を持つ、より高いレベルのカプセルにその出力を送ることを好む。）

## 1 Introduction

---

人間の視覚は、注意深く決定された固定点のシーケンスを使用して、光学アレイのごく一部だけが最高解像度で処理されることを確実にすることによって、無関係な細部を無視します。イントロスペクションは、シーンの知識がどれくらい固定のシーケンスから来ているのか、単一の固定からどれくらい集まってきたのかを理解するうえでの貧弱なガイドですが、このペーパーでは、単一の固定によって、識別されたオブジェクトとそのプロパティ私たちのマルチレイヤービジュアルシステムは、各固定上に構文木のような構造を作成すると仮定し、これらの単一固定構文解析木が複数の固定に対してどのように調整されるかの問題は無視します。

パースツリーは一般に、動的にメモリを割り当てることでオンザフライで構築されます。Hinton et al. しかし、我々は、単一の固定のために、岩石から刻まれた彫刻のように、固定された多層ニューラルネットワークからパースツリーを切り取ったと仮定する。各層は、「カプセル」（Hinton et al. [2011]）と呼ばれる多くの小グループのニューロンに分割され、パースツリーの各ノードはアクティブなカプセルに対応します。反復ルーティングプロセスを使用して、各アクティブカプセルは、ツリーの親であるために、上記の層のカプセルを選択します。ビジュアルシステムのレベルが高い場合、この反復プロセスは、パーツを全体に割り当てるという問題を解決することになります。

アクティブなカプセル内のニューロンの活動は、画像中に存在する特定のエンティティの様々な特性を表す。これらのプロパティには、姿勢（位置、サイズ、方向）、変形、速度、アルベド、色相、テクスチャなど、さまざまな種類のインスタンス化パラメータが含まれます。非常に特殊なプロパティの1つは、イメージ内にインスタンス化されたエンティティの存在です。存在を表現する明

白な方法は、エンティティが存在する確率を出力とする別個のロジスティクスユニットを使用することである。この論文では、エンティティの存在を表現するためにインスタンス化パラメータのベクトルの全体的な長さを使用し、エンティティのプロパティを表すベクトルの向きを強制的に使用するという興味深い代替案を検討します。我々は、ベクトルの向きを変えずにその大きさを縮小する非線形性を適用することによって、カプセルのベクトル出力の長さが1を超えないことを保証する。

カプセルの出力がベクトルであるという事実は、カプセルの出力が上記の層の適切な親に送られることを確実にするために、強力な動的ルーティング機構を使用することを可能にする。最初に、出力はすべての可能な親にルーティングされるが、合計1の結合係数によって縮小される。各可能な親について、カプセルは、自身の出力に重み行列を乗算することによって「予測ベクトル」を計算する。この予測ベクトルが、可能な親の出力を伴う大きなスカラー積を有する場合、その親の結合係数を増加させ、他の親のために減少させるトップダウンフィードバックが存在する。これは、カプセルがその親に与える寄与を増加させ、したがって親の出力とのカプセルの予測のスカラー積をさらに増加させる。このタイプの「合意によるルーティング」は、あるレイヤのニューロンがレイヤ内のローカルプール内の最もアクティブなフィーチャディテクタを無視することを可能にする、max-poolingによって実装されるルーティングの非常に基本的な形式よりはるかに効果的である以下。ダイナミックルーティングメカニズムは、高度にオーバーラップするオブジェクトをセグメント化するために必要な「解説」を実装する効果的な方法であることを示しています。

畳み込みニューラルネットワーク（CNN）は、学習された特徴検出器の翻訳された複製を使用する。これにより、画像のある位置で取得された良好な重み値に関する知識を他の位置に翻訳することができます。これは、画像解釈において非常に有用であることが証明されている。CNNのスカラー出力特徴検出器をベクトル出力カプセルに置き換え、max-poolingを経路指定で置き換えるとしても、学習した知識を空間全体に複製したいと考えています。これを達成するために、カプセルの最後のレイヤーを除くすべてのレイヤーを畳み込みにします。CNNと同様に、より高レベルのカプセルは画像のより広い領域をカバーします。しかし、max-poolingとは異なり、領域内のエンティティの正確な位置に関する情報を捨てません。低レベルカプセルの場合、位置情報は、カプセルがアクティブであることによって「スペース・コード」される。階層を上るにつれて、カプセルの出力ベクトルの実数値コンポーネントでは、ますます多くの位置情報が「レートコード」されています。スペースコーディングからレートコーディングへの移行は、より高いレベルのカプセルがより自由度の高い複雑なエンティティを表すという事実と相まって、階層の昇順にカプセルの次元性が向上することを示唆しています。

## 2 How the vector inputs and outputs of a capsule are computed

---

カプセルの一般的な考え方を実行する方法はたくさんあります。このホワイトペーパーの目的は、この全体の空間を探索することではなく、単純な実装がうまく機能し、動的ルーティングが役立つことを示すことです。

カプセルの出力ベクトルの長さが、カプセルによって表されるエンティティが現在の入力に存在する確率を表すことを望む。したがって、非線形の「squashing」関数を使用して、短いベクトルがほぼゼロの長さに収縮し、長いベクトルが1よりわずかに下に収縮するようにします。この非線形性を有効に活用するために差別的な学習を行います。

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1 + \|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|} \quad (1)$$

ここで、 $\mathbf{v}_j$ は、カプセル $j$ のベクトル出力であり、 $\mathbf{s}_j$ は、その総入力である。

カプセルの第1層を除くすべての場合、カプセル $\mathbf{s}_j$ への総入力は、下の層のカプセルからのすべての「予測ベクトル」 $\mathbf{u}_j \mid i$ にわたる重み付けされた和であり、重み行列 $\mathbf{W}_{ij}$ により、下の層のカプセルの出力 $\mathbf{u}_i$ を計算する。

$$\mathbf{s}_j = \sum_i c_{ij} \hat{\mathbf{u}}_{j|i}, \quad \hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i \quad (2)$$

ここで、 $c_{ij}$ は、反復動的ルーティングプロセスによって決定される結合係数である。

カプセル $i$ と上の層のすべてのカプセルとの間の結合係数は1に足し合わされ、その初期ロジット $b_{ij}$ は、カプセル $i$ がカプセル $j$ に結合されるべき事前対数確率である「ルーティングソフトマックス」によって決定される。

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})} \quad (3)$$

事前対数確率は他のすべての重みと同時に区別して学習することができます。それらは2つのカプセルの位置およびタイプに依存するが、現在の入力画像には依存しない。次に、上の層の各カプセル $j$ の現在の出力 $\mathbf{v}_j$ と、カプセル $i$ によって作られた予測 $\mathbf{u}_j \mid i$ との間の一致を測定することによって、最初の結合係数が反復的に精緻化される。

この合意は単にスカラー積 $a_{ij} = \mathbf{v}_j \cdot \mathbf{u}_j \mid i$ である。この合意は、それが対数尤度であるかのように扱われ、カプセル $i$ をより高レベルのカプセルに結び付けるすべての結合係数の新しい値を計算する前に初期ロジット $b_{ij}$ に加えられる。

畳み込みカプセル層では、各カプセルは、グリッドの各メンバーおよびカプセルの各タイプごとに異なる変換マトリックスを使用して、上のレイヤーの各タイプのカプセルにベクトルのローカルグリッドを出力します。

---

#### Procedure 1 Routing algorithm.

---

```

1: procedure ROUTING( $\hat{\mathbf{u}}_{j|i}, r, l$ )
2:   for all capsule  $i$  in layer  $l$  and capsule  $j$  in layer  $(l + 1)$ :  $b_{ij} \leftarrow 0$ .
3:   for  $r$  iterations do
4:     for all capsule  $i$  in layer  $l$ :  $\mathbf{c}_i \leftarrow \text{softmax}(\mathbf{b}_i)$  ▷ softmax computes Eq. 3
5:     for all capsule  $j$  in layer  $(l + 1)$ :  $\mathbf{s}_j \leftarrow \sum_i c_{ij} \hat{\mathbf{u}}_{j|i}$ 
6:     for all capsule  $j$  in layer  $(l + 1)$ :  $\mathbf{v}_j \leftarrow \text{squash}(\mathbf{s}_j)$  ▷ squash computes Eq. 1
7:     for all capsule  $i$  in layer  $l$  and capsule  $j$  in layer  $(l + 1)$ :  $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i} \cdot \mathbf{v}_j$ 
   return  $\mathbf{v}_j$ 

```

---

## 3 Margin loss for digit existence

---

インスタンス化ベクトルの長さを使用して、カプセルのエンティティが存在する確率を表します。その数字が画像内に存在する場合に限り、数字クラスkのトップレベルカプセルに長いインスタンス化ベクトルを持たせたい。複数桁を許容するために、各桁のカプセルkごとに別々のマージンロス $L_k$ を使用します。

$$L_k = T_k \max(0, m^+ - \|\mathbf{v}_k\|)^2 + \lambda(1 - T_k) \max(0, \|\mathbf{v}_k\| - m^-)^2 \quad (4)$$

ここで、クラスkの桁が存在する場合には $T_k = 1$ であり、 $m^+ = 0.9$ および $m^- = 0.1$ である。不在桁クラスの損失の $\lambda$ ダウン重み付けは、初期学習がすべての桁カプセルの活動ベクトルの長さを縮小するのを止める。 $\lambda = 0.5$ を使用します。総損失は、単純にすべての桁のカプセルの損失の合計です。

## 4 CapsNet architecture

シンプルなCapsNetアーキテクチャが図1に示されています。アーキテクチャは浅く、畳み込みレイヤーが2つと完全に接続されたレイヤーが1つしかありません。Conv1は、1のストライドとReLUの活性化を持つ256, 9×9畳み込みカーネルを持っています。この層は、画素強度を局所特徴検出器の活動に変換し、次いでこれを主カプセルへの入力として使用する。

PrimaryCapsulesは、多次元エンティティの最低レベルであり、逆グラフィックスの観点から、PrimaryCapsulesをアクティブ化することは、レンダリングプロセスを反転することに対応する。これは、身近なものを作るためにインスタンス化されたパーツをつなぎ合わせることはまったく異なるタイプの計算です。これは、カプセルがうまく設計されているものです。

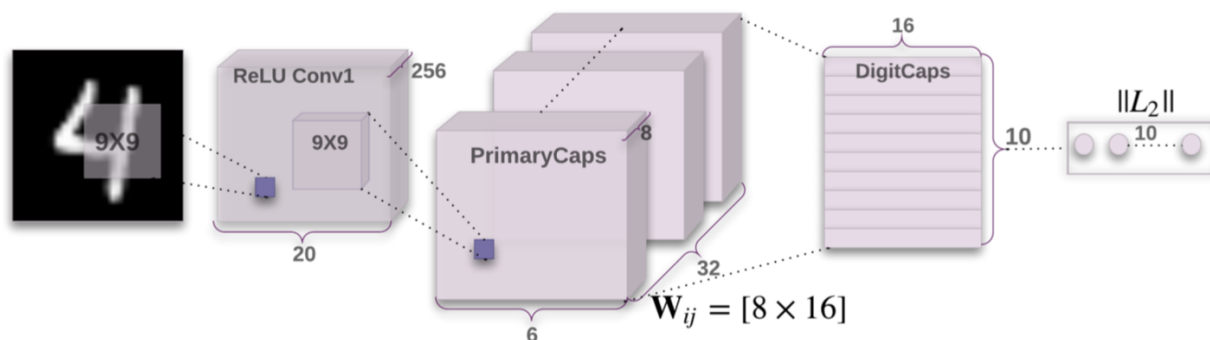


図1：3層のシンプルなCapsNetこのモデルは、深い畳み込みネットワーク（Chang and Chen [2015]など）に匹敵する結果をもたらします。DigitCaps層の各カプセルの活動ベクトルの長さは、各クラスのインスタンスの存在を示し、分類損失を計算するために使用されます。W<sub>ij</sub>はPrimaryCapsulesの各u<sub>i</sub>、i ∈ (1, 32×6×6)とv<sub>j</sub>、j ∈ (1, 10)の間の重み行列である。

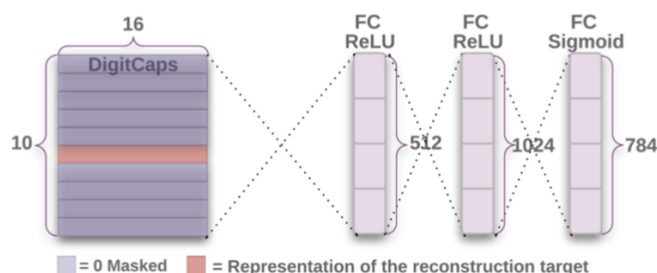


図2：DigitCaps層表現から桁を再構築するためのデコーダ構造。画像とシグモイド層の出力との間のユークリッド距離は、訓練中に最小化される。我々は、トレーニング中に真のラベルを再構成ターゲットとして使用する。

第2の層（PrimaryCapsules）は、32チャンネルの畳み込み8Dカプセルを有する畳み込みカプセル層である（すなわち、各主カプセルは、 $9 \times 9$ カーネルと2のストライドを有する8つの畳み込み単位を含む）。各1次カプセル出力は、受容野がカプセルの中心の位置と重なるすべての $256 \times 81$ 個のConv1ユニットの出力を見る。合計で、PrimaryCapsulesには $[32 \times 6 \times 6]$ カプセル出力（各出力は8Dベクトル）があり、 $[6 \times 6]$ グリッドの各カプセルは互いに重みを共有しています。PrimaryCapsulesは、ブロック非線形性として式1を持つ畳み込みレイヤーとして見ることができます。1をそのブロックの非線形性として表す。最終的なレイヤー（DigitCaps）には、1桁あたり16Dカプセルが1クラスあり、これらのカプセルのそれぞれは、下のレイヤーのすべてのカプセルからの入力を受け取ります。

2つの連続したカプセル層（例えば、PrimaryCapsulesとDigitCaps）の間をルーティングするだけです。Conv1の出力は1Dなので、その空間には同意する方向はありません。したがって、Conv1とPrimaryCapsule間のルーティングは使用されません。すべてのルーティングログ（ $b_{ij}$ ）はゼロに初期化されます。したがって、最初に、カプセル出力（ $u_i$ ）がすべての親カプセル（ $v_0 \dots v_9$ ）に等しい確率（ $c_{ij}$ ）で送信されます。我々の実装はTensorFlow（Abadi et al. [2016]）にあり、方程式のマージン損失の合計を最小限に抑えるために、Adamオプティマイザ（KingmaとBa [2014]）をTensorFlowのデフォルトパラメータ（指数関数的に低下する学習率を含む）で使います。

## 4.1 Reconstruction as a regularization method

ディジットカプセルが入力桁のインスタンス化パラメータをエンコードするように、再構成損失を追加します。トレーニング中に、正しい桁のカプセルの活動ベクトル以外のすべてをマスクする。次に、このアクティビティベクトルを使用して入力画像を再構成します。ディジット・カプセルの出力は、図2で説明したようにピクセル強度をモデル化する3つの完全に接続された層からなるデコーダに供給される。ロジスティック・ユニットの出力とピクセル強度の差の二乗和が最小になる。この再構成損失を0.0005だけ縮小して、トレーニング中の証拠金損失を支配しないようにします。図3に示すように、CapsNetの16D出力からの再構築は、重要な詳細のみを維持しながら堅牢である。




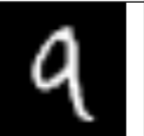








$(l, p, r)$	(2, 2, 2)	(5, 5, 5)	(8, 8, 8)	(9, 9, 9)	(5, 3, 5)	(5, 3, 3)
Input						
Output						

図3：3回の経路指定反復を伴うCapsNetのMNISTテスト再構成の例。(l、p、r)はそれぞれラベル、予測および再構成ターゲットを表す。2つの右端の列は、故障例の2つの再構成を示し、モデルがこの画像内で5と3をどのように混同しているかを説明します。他の列は正しい分類からのものであり、モデルはノイズを平滑化しながら多くの細部を保持していることを示しています。

Method	Routing	Reconstruction	MNIST (%)	MultiMNIST (%)
Baseline	-	-	0.39	8.1
CapsNet	1	no	$0.34 \pm 0.032$	-
CapsNet	1	yes	$0.29 \pm 0.011$	7.5
CapsNet	3	no	$0.35 \pm 0.036$	-
CapsNet	3	yes	<b><math>0.25 \pm 0.005</math></b>	<b>5.2</b>

表1：CapsNet分類テスト精度。MNIST平均および標準偏差の結果は、3つの試験から報告される。

## 5 Capsules on MNIST

訓練は、ゼロパディングを用いて各方向に2ピクセルまでシフトされた28×28MNIST (LeCunら[1998])の画像に対して行われる。他のデータ拡大/変形は使用されない。データセットには、それぞれトレーニングとテストのための60Kと10Kのイメージがあります。

モデルの平均化を行わずに単一のモデルを使用してテストします。Wanら[2013]は、回転とスケールリングを使用してデータのアンサンブルと補強を行って0.21%のテストエラーを達成しました。彼らは彼らなしで0.39%を達成する。以前より深いネットワークでのみ達成された3層ネットワークでは、低いテストエラー(0.25%)が得られます。表1は、異なるCapsNet設定に対するMNISTのテストエラー率を報告し、ルーティングと再構成の正式化の重要性を示しています。再構成正則化器を追加することにより、ポーズ符号化をカプセルベクトルに強制することによってルーティング性能を向上させる。

ベースラインは、256チャンネル、256チャンネル、128チャンネルの3つの畳み込みレイヤを備えた標準的なCNNです。各々は5×5のカーネルと1のストライドを有する。最後の畳み込み層の後に、サイズ328,192の2つの完全に接続された層が続く。最後の完全に接続された層は、クロスエントロピー損失を伴う10クラスのソフトマックス層へのドロップアウトと接続される。ベースラインは、Adamオプティマイザを搭載した2ピクセルシフトMNISTでも訓練されています。ベースラインは、CapsNetに近い計算コストを維持しながら、MNISTで最高のパフォーマンスを達成するように設計されています。パラメータ数に関しては、ベースラインは35.4M、CapsNetは8.2Mパラメータ、6.8Mパラメータは再構成サブネットワークなしです。

### 5.1 What the individual dimensions of a capsule represent

1桁のみのエンコーディングを渡し、他の桁をゼロにするので、ディジットカプセルのディメンションは、そのクラスのディジットがインスタンス化される方法のバリエーションの空間をまたぐことを学ぶべきです。これらのバリエーションには、ストロークの厚さ、スキューおよび幅が含まれます。それらには、2の尾の長さなどの桁固有の変化も含まれます。デコードネットワークを使用することによって、個々の次元が表すものがわかります。正しいディジットカプセルのアクティビティベ

クトルを計算した後、このアクティビティベクトルの乱れたバージョンをデコードネットワークに供給し、その摂動がどのように再構成に影響を与えるかを見ることができます。これらの摂動の例を図4に示します。カプセルの1つのディメンション（16のうちの1つ）がほぼ常に数字の幅を表すことがわかりました。いくつかのディメンションはグローバルバリエーションの組み合わせを表しますが、ディジットのローカライズされた部分のバリエーションを表す他のディメンションがあります。例えば、6の昇順の長さとのループのサイズには、異なる次元が使用されます。




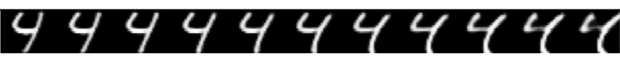

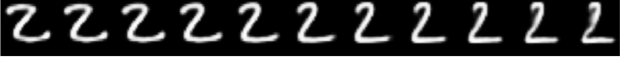
Scale and thickness	
Localized part	
Stroke thickness	
Localized skew	
Width and translation	
Localized part	

図4：次元の摂動。各行は、DigitCaps表示の16個のディメンションの1つが $[-0.25, 0.25]$ の範囲で0.05の間隔で調整されたときの再構成を示しています。

## 5.2 Robustness to Affine Transformations

実験では、各DigitCapsカプセルが伝統的な畳み込みネットワークよりも各クラスのより堅牢な表現を学習することが示されています。手書き数字のスキュー、回転、スタイルなどには自然な差異があるため、訓練されたCapsNetはトレーニングデータの小さなアフィン変換に対して適度に堅牢です。

アフィン変換に対するCapsNetの頑健性をテストするために、私たちは、パッド付きで変換されたMNISTトレーニングセットにCapsNetと従来の畳み込みネットワーク（MaxPoolingとDropOut）を訓練しました。各例は40の黒い背景にランダムに配置されたMNIST数字×40ピクセル。次に、affNISTデータセットでこのネットワークをテストしました。各サンプルはランダムなアフィン変換を持つMNISTディジットです。私たちのモデルは、標準的なMNISTに見られる翻訳や自然な変換以外のアフィン変換では決して訓練されませんでした。早期に停止して、拡張されたMNISTテストセットで99.23%の精度を達成した、トレーニングを受けていないCapsNetが、アニーニストテストセットで79%の精度を達成しました。拡張された反復テストセットで同様の精度（99.22%）を達成した同様の数のパラメータを持つ従来の畳み込みモデルは、アニーニストテストセットで66%しか達成できませんでした。

## 6 Segmenting highly overlapping digits



ダイナミックルーティングは、1つのレベルの各カプセルが、下のレベルでいくつかのアクティブなカプセルに出席し、他のレベルを無視することを可能にする並列注意メカニズムとして見る事ができる。これにより、オブジェクトが重なっていてもモデル内の複数のオブジェクトを認識することができます。HintonらHintonら[2000]らは同様の領域でネットワークをテストしている

(Goodfellowら[2013]、Baら[2014]、Greffら[ルーティング・アグリーメントによって、セグメント化を助けるためにオブジェクトの形状について事前に使用することが可能になり、ピクセルの領域においてより高いレベルのセグメンテーション決定を行う必要性がなくなるはずである。

## 6.1 MultiMNIST dataset

同じセット（トレーニングまたはテスト）の別の桁の上に数字を重ねることによって、MultiMNIST トレーニングおよびテストデータセットを生成しますが、異なるクラスです。各桁は、各方向に4ピクセルまでシフトされ、その結果、36×36の画像が得られる。28×28の画像内の数字が20×20のボックスに囲まれていることを考慮すると、平均して2桁の境界ボックスは80%の重なりを有する。MNISTデータセットの各桁に対して、1K MultiMNISTの例を生成します。したがって、トレーニングセットのサイズは60Mで、テストセットのサイズは10Mです。

R:(2,7) L:(2,7)	R:(6,0) L:(6,0)	R:(6,8) L:(6,8)	R:(7,1) L:(7,1)	*R:(5,7) L:(5,0)	*R:(2,3) L:(4,3)	R:(2,8) L:(2,8)	R:P:(2,7) L:(2,8)
R:(8,7) L:(8,7)	R:(9,4) L:(9,4)	R:(9,5) L:(9,5)	R:(8,4) L:(8,4)	*R:(0,8) L:(1,8)	*R:(1,6) L:(7,6)	R:(4,9) L:(4,9)	R:P:(4,0) L:(4,9)

図5：MultiMNISTテストデータセットでの3回のルーティング反復によるCapsNetの再構成例2つの再構成された数字は、下側の画像として緑色と赤色で重ねられます。上の画像は入力画像を示しています。L：(l1、l2) は画像内の2桁のラベルを表し、R：(r1、r2) は再構成に使用される2桁の数字を表します。2つの右端の列は、ラベルおよび予測 (P) から再構成された間違っ分類を有する2つの例を示す。(2,8) の例では、モデルは8を7と混同し、(4,9) では9を0と混同します。他の列は正しい分類を持ち、モデルがすべてのピクセルを占め、一方非常に難しいシナリオ (1〜4桁目) では1ピクセルから2桁になります。データセット生成では、ピクセル値は1でクリップされます。(\*) マークが付いた2つの列は、ラベルでも予測でもない桁からの再構成を示します。これらの列は、モデルが、存在しないものも含めて、画像内のすべての桁に最適なものを見つけるだけではないことを示唆しています。したがって、(5,0) の場合、7に再構成することはできません。な



ぜなら、最も適合する5と0があり、すべてのピクセルを考慮していることがわかっているからです。また、(8,1) の場合、8のループはすでに8で説明されているため、0をトリガしません。したがって、1つのピクセルが他のサポートを持たない場合は、1つのピクセルを2桁に割り当てません。

## 6.2 MultiMNIST results

MultiMNISTトレーニングデータのゼロから訓練された3層CapsNetモデルは、ベースライン畳み込みモデルよりも高いテスト分類精度を実現します。Ba et al. の逐次注意モデルのように、高度に重複する桁の対について5.0%の同じ分類誤り率を達成している。[2014]ははるかに少ないオーバーラップを実現するはるかに簡単なタスクを実現します（Ba et al. [2014]の場合、我々のケースでは2桁の囲みの80%が4%未満です）。テストセットからの画像対からなるテスト画像では、カプセルネットワークによって生成された分類として、2つの最もアクティブなディジットカプセルを扱う。再構成中に、一度に1桁を選択し、選択した桁の画像を再構成するために、選択した桁カプセルの活動ベクトルを使用します（我々は合成画像を生成するためにこの画像を使用しています）。我々のMNISTモデルとの唯一の違いは、訓練データセットがより大きいため、学習レートの減衰ステップの期間を10倍大きくしたことです。

図5に示す再構成は、CapsNetが画像を2つの元の数字に分割できることを示しています。このセグメンテーションはピクセルレベルではないので、モデルはすべてのピクセルを考慮しながらオーバーラップ（ピクセルが両方の桁でオン）を正しく処理できることがわかります。各桁の位置とスタイルはDigitCapsでエンコードされます。デコードは、符号化された数字を再構成することを学んだ。オーバーラップに関係なく数字を再構成できるという事実は、各桁のカプセルがPrimaryCapsulesレイヤから受け取った票からスタイルと位置を拾うことができることを示しています。

表1は、この作業でルーティングするカプセルの重要性を強調しています。CapsNetの精度の分類のベースラインとして、2つの畳み込みレイヤーとその上に完全に接続された2つのレイヤーを持つ畳み込みネットワークを訓練しました。第1層は、サイズ9×9およびストライド1の512個のコンボリューションカーネルを有する。第2層は、サイズ5×5およびストライド1の256個のカーネルを有する。各コンボリューションレイヤの後、モデルはサイズ2×2およびストライド2のプール層を有する。第3の層は1024Dの完全に接続された層である。すべての3つのレイヤーにReLU非線形性があります。10ユニットの最終層は完全に接続されています。最終層の出力にS字状のクロスエントロピー損失を訓練するために、TensorFlowのデフォルトAdamオプティマイザ（KingmaとBa [2014]）を使用します。このモデルは24.56Mのパラメータを持ち、11.36Mのパラメータを持つCapsNetの2倍のパラメータです。我々は、より小さなCNN（5×5の32と64の畳み込みカーネルと1のストライドと完全に接続された512Dレイヤー）から始め、MultiMNISTデータの10Kサブセットで最良のテスト精度に達するまでネットワークの幅を段階的に増加させました。また、10K検証セットで正しい減衰ステップを検索しました。

最もアクティブな2つのDigitCapsカプセルを1つずつデコードし、2つの画像を取得します。次に、各桁に0でない強度のピクセルを割り当てることによって、各桁のセグメンテーション結果が得られます。

## 7 Other datasets

---

CIFAR10でカプセルモデルをテストし、 $24 \times 24$ パッチの画像に対して3回のルーティング反復で訓練された7つのモデルのアンサンブルで10.6%の誤差を達成しました。各モデルは、3つのカラーチャンネルがあり、我々は64種類の主カプセルを使用したことを除いて、MNISTのために使用した単純なモデルと同じ構造をしています。我々は、10個のカプセルの最終層がイメージのすべてを説明することを期待していないので、ルーティングソフトマクロのための「あまりにも高い」カテゴリを導入するのに役立つことも発見した。10.6%のテストエラーは、CIFAR10に最初に適用されたときにどのような標準畳み込みネットが達成されたかに関するものです（Zeiler and Fergus [2013]）。

生成モデルと共有するカプセルの1つの欠点は、画像内のすべてのものを考慮に入れて、ダイナミックルーティングで追加の「孤立した」カテゴリを使用するよりも、クラッタをモデル化できるときに優れているということです。CIFAR-10では、バックグラウンドがあまりにも変化しすぎて、妥当なサイズのネットではモデル化できないため、パフォーマンスが低下することがあります。

我々はまた、smallINORB（2004年LeCun [2004]）上でMNISTに使用したものと全く同じアーキテクチャをテストし、最先端技術と同等の2.7%のテストエラーレートを達成した（CiresHanet al. [2011]）。smallINORBデータセットは、 $96 \times 96$ ステレオグレースケール画像で構成されています。画像を $48 \times 48$ にリサイズし、トレーニング中に $32 \times 32$ 作物をランダムに処理しました。テスト中に中央の $32 \times 32$ パッチを通過しました。

私たちはまた、SVHN（Netzer et al. [2011]）の小さな訓練セットで、わずか73257の画像だけで、より小さなネットワークを訓練しました。第1の畳み込み層チャンネルの数を64に減らし、一次カプセル層を最終的に8D最終カプセル層を有する16個の6Dカプセルにし、テストセットで4.3%を達成した。

## 8 Discussion and previous work

30年間、音声認識の最先端技術は、出力分布としてガウス混合を用いた隠れマルコフモデルを使用していました。これらのモデルは小型コンピュータでは簡単に学習できましたが、最終的には致命的な表現上の制約がありました。たとえば、分散表現を使用するリカレントニューラルネットワークと比較して1対n表現が指数関数的に非効率的です。HMMがこれまでに生成した文字列について覚えることができる情報量を倍にするには、隠れノードの数を2乗する必要があります。反復的なネットでは、隠れニューロンの数を2倍にするだけです。

今度は畳み込みニューラルネットワークが物体認識に対する支配的なアプローチとなったので、指数的な非効率性が存在するかどうかを尋ねるのは理にかなっています。良い候補は、畳み込みネットが新しい視点に一般化するのが難しいことです。翻訳を処理する能力は組み込まれていますが、アフィン変換の他の次元では、ディメンション数に応じて指数関数的に増加するグリッド上にフィーチャディテクタを複製するか、同様に指数関数的な方法。カプセル（Hinton et al. [2011]）は、ピクセル強度を認識されたフラグメントのインスタンス化パラメータのベクトルに変換し、変換行列をフラグメントに適用して、より大きなフラグメントのインスタンス化パラメータを予測することによってこれらの指数的非効率性を回避する。部分と全体の間の本質的な空間的關係を符号化することを学ぶ変換行列は、自動的に新しい視点に一般化する視点不変の知識を構成する。Hintonら [2011]は、PrimaryCapsuleレイヤのインスタンス化パラメータを生成するためのトランスコーディングオートエンコーダを提案し、そのシステムは変換マトリックスを外部から提供する必要があります。 「アクティブで低レベルのカプセルによって予測されるポーズの合意を使用することによって、どれほど大きくて複雑な視覚的エンティティが認識されるか」という答えを示す完全なシス

テムを提案します。

カプセルは非常に強力な表現を前提としています。画像の各位置には、カプセルが表すエンティティの種類のインスタンスが最大で1つあります。このような仮定は、「crowding」 (Pelli et al. [2004]) と呼ばれる知覚現象によって動機づけられたものであり、結合問題 (Hinton [1981a]) を排除し、カプセルに分散表現 (その活動ベクトル) 指定された場所のその型のエンティティのインスタンス化パラメータこの分散表現は、高次元グリッド上の点をアクティブ化することによってインスタンス化パラメータを符号化するより指数関数的に効率的であり、適切な分散表現でカプセル化すると、行列の乗算によって空間関係をモデル化できるという事実を十分に活用できます。

カプセルは、視点が変化するにつれて変化する神経活動を使用するが、視点の変化を活動から排除しようとするのではない。これにより、空間トランスフォーマーネットワーク (Jaderberg et al. [2015]) のような「正規化」メソッドよりも優れています。異なるオブジェクトやオブジェクトパーツの複数の異なるアフィン変換を同時に扱うことができます。

カプセル化はまた、視覚化において最も困難な問題の1つであるセグメンテーションを処理するのに非常に適しています。このホワイトペーパーで説明したように、インスタンス化パラメータのベクトルによってルーティングごとに使用できるためです。ダイナミックルーティング手順の重要性は、視覚野における不慣れなパターン認識の生物学的にもっともらしいモデルによって裏付けられる。Hinton [1981b]は、オブジェクト認識のために使用できる形状記述を生成するために、ダイナミック接続と標準オブジェクトベースの参照フレームを提案する。OlshausenらHinton [1981b]の動的接続を改善し、オブジェクト表現の生物学的にもっともらしい、位置およびスケール不変モデルを提示する[1993]。

カプセルに関する研究は、今世紀初めの音声認識のためのリカレントニューラルネットワークの研究と同様の段階にあります。より良いアプローチであると信じる根本的な理由がありますが、高度に開発された技術を凌駕するには、もっと小さな洞察が必要になるでしょう。単純なカプセルシステムがすでに重複する数字をセグメント化する際に比類のない性能を発揮しているという事実は、カプセルが探索の価値のある方向であることの早期の示唆である。

**Acknowledgement.** 建設的なコメントを私たちに提供した多くの人の中で、私たちはロバート・ゲンス、エリック・ラングロワ、ヴィンセント・ヴァノウケ、クリス・ウィリアムズ、そして批評家に彼らの有益なコメントと修正に特に感謝しています。

## References

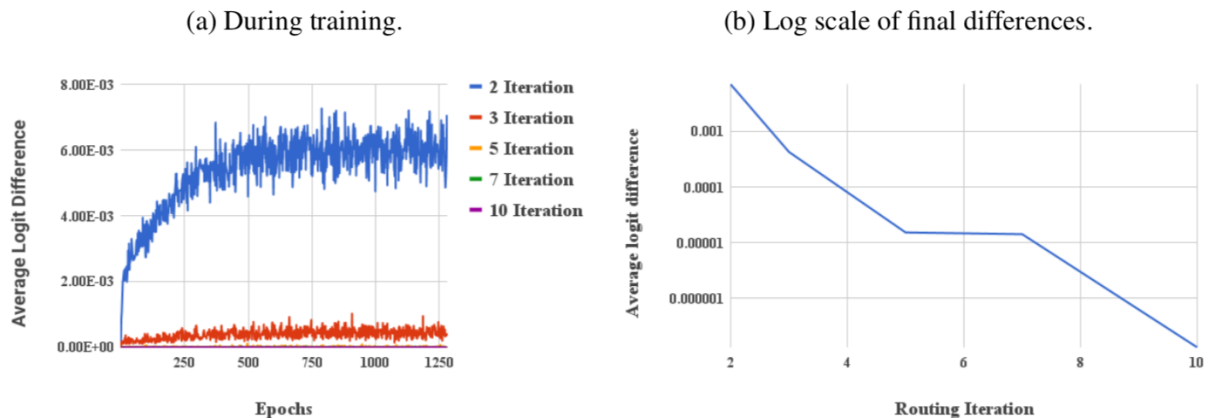
---

- Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg SCorrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. Tensorflow: Large-scale machinelearning on heterogeneous distributed systems. arXiv preprint arXiv:1603.04467, 2016.
- Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visualattention. arXiv preprint arXiv:1412.7755, 2014.
- Jia-Ren Chang and Yong-Sheng Chen. Batch-normalized maxout network in network. arXiv preprintarXiv:1511.02583, 2015.
- Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca M Gambardella, and Jürgen Schmidhuber. High-performance neural networks for visual object classification. arXiv preprint arXiv:1102.0183,2011.

- Ian J Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. arXiv preprint arXiv:1312.6082, 2013.
- Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hao, Harri Valpola, and Jürgen Schmidhuber. Tagger: Deep unsupervised perceptual grouping. In *Advances in Neural Information Processing Systems*, pages 4484–4492, 2016.
- Geoffrey E Hinton. Shape representation in parallel systems. In *International Joint Conference on Artificial Intelligence Vol 2*, 1981a.
- Geoffrey E Hinton. A parallel computation that assigns canonical object-based frames of reference. In *Proceedings of the 7th international joint conference on Artificial intelligence- Volume 2*, pages 683–685. Morgan Kaufmann Publishers Inc., 1981b.
- Geoffrey E Hinton, Zoubin Ghahramani, and Yee Whye Teh. Learning to parse images. In *Advances in neural information processing systems*, pages 463–469, 2000.
- Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In *International Conference on Artificial Neural Networks*, pages 44–51. Springer, 2011.
- Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transform networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015.
- Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
- Yann LeCun, Corinna Cortes, and Christopher JC Burges. The mnist database of handwritten digits, 1998.
- Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–104. IEEE, 2004.
- Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.
- Bruno A Olshausen, Charles H Anderson, and David C Van Essen. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Journal of Neuroscience*, 13(11):4700–4719, 1993.
- Denis G Pelli, Melanie Palomares, and Najib J Majaj. Crowding is unlike ordinary masking: Distinguishing feature integration from detection. *Journal of vision*, 4(12):12–12, 2004.
- Li Wan, Matthew D Zeiler, Sixin Zhang, Yann LeCun, and Rob Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013.
- Matthew D Zeiler and Rob Fergus. Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint arXiv:1301.3557, 2013.

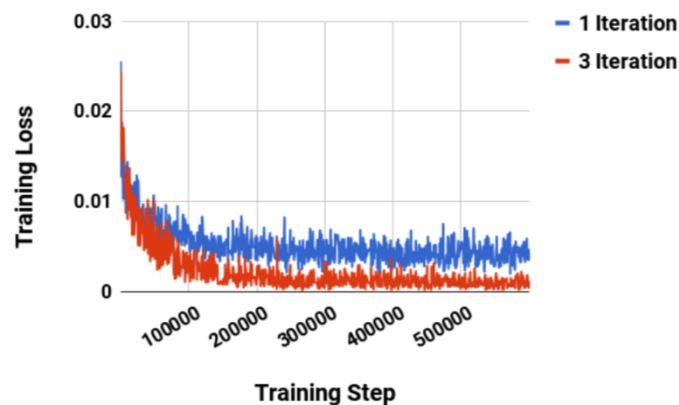
# A How many routing iterations to use?

ルーティングアルゴリズムのコンバージェンスを実験的に検証するために、各ルーティング反復におけるルーティングログの平均変化をプロットします。図A.1は、各ルーティング反復後の平均bij変化を示す。実験的に我々は、訓練の開始から5回の反復によるルーティングの変化がごくわずかであることを観察する。ルーティングの第2パスの平均変化は、トレーニングの500エポック後に0.007に落ち着きますが、ルーティングの繰り返し5では、ロギングは平均で1〜5だけ変化します。



図A.1：各ルーティング反復による各ルーティングロジット（bij）の平均変化。MNISTのトレーニングが500エポック後に、平均変化が安定し、右図に示すように、それはより多くのルーティング反復で対数スケールでほぼ直線的に減少する。

一般に、より多くのルーティング反復がネットワーク容量を増加させ、トレーニングデータセットにあまりにもフィットする傾向があることがわかった。図A.2は、ルーティングの1回の反復と3回のルーティングの反復でトレーニングした場合のCifar10でのCapsuleトレーニング損失の比較を示しています。図A.2と図A.1に動機付けされているので、すべての実験で3回のルーティングの繰り返しを提案します。



図A.2：cifar10データセットでのCapsuleNetの損失損失。各トレーニングステップでのバッチサイズは128です。ルーティングの3回の反復を伴うCapsuleNetは、損失をより迅速に最適化し、最後にはより低い損失に収束します。

