Biological Applications of Deep Learning
Lecture 10

Alexander Schönhuth

UNIVERSITÄT BIELEFELD
Faculty of Technology

Bielefeld University
December 14, 2022

CONTENTS TODAY

- ALSNet
  - Predicting ALS disease status from genotype profiles
  - Employ convolution on sequences directly
- Capsule Networks: Tutorial
  - Motivation
  - Methodology
  - Effects

Reminder: ALS-Net

Reference

- B. Yin, M. Balvert, R. van de Spek, B. Dutilh, S. Bohte, J. Veldink, A. Schönhuth
  Using the structure of genome data in the design of deep neural networks for predicting amyotrophic lateral sclerosis from genotype
  Bioinformatics, 2019

Genetic Architecture:
A Formal Description

UNIVERSITÄT
BIELEFELD

Individual Genotypes

polymorphic sites

Reference	ACG	C	GTTAA	T	ACGGT	A	CTGGA	G	CGT
Alternative		T		C		G		A	
Mother Copy	ACG	C	GTTAA	C	ACGGT	A	CTGGA	A	CGT
Father Copy	ACG	C	GTTAA	T	ACGGT	G	CTGGA	A	CGT
Genotype		0		1		1		2	

Represent individuals by their genotypes

- Vectors whose length is number of polymorphic sites, with entries
  - 0 = homozygous for reference
  - 1 = heterozygous
  - 2 = homozygous for alternative

The Genetic Architecture of ALS

Definition

Let $\mathcal{X}$ be all people, represented by their genotypes.

The *genetic architecture* f_{ALS} of ALS is a function

$$f_{\mathrm{ALS}} : \mathcal{X} \longrightarrow \{0,1\}$$

where for $X \in \mathcal{X}$

$$f(X) = \begin{cases} 1 & X \text{ affected by ALS} \\ 0 & \text{otherwise} \end{cases}$$

Machine Learning the Genetic Architecture

Learning the Genetic Architecture

Let $\mathcal{M}$ is a class of ML compatible functions:

Approximate f_{ALS} by $f^*_{\text{ALS}} \in \mathcal{M}$

- using known examples
  - cases: $(x, f_{\text{ALS}}(x) = 1)$
  - controls: $(x, f_{\text{ALS}}(x) = 0)$

  as *training/validation data*
- Goodness of $f^*_{\text{ALS}}(x)$ is evaluated on previously unseen *test data*
- Linear approximations f^*_{ALS} work poorly
  ☞ This explains the large amount of missing heritability

Challenges

- *Input size*: Length of genotype string N
  - N several millions
  - Dimensionality of problem too large, too many parameters to be learnt
- *Convolving genotype vectors?* Use of convolution tailored towards image analysis

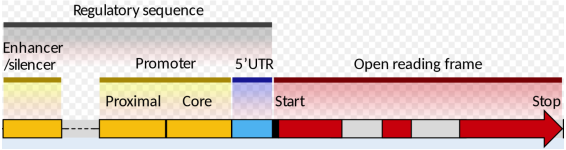
Feature Selection

—

Picking relevant variant sites from 5 million sites overall

Feature Selection I: Regulatory Regions

Genetics Principle Guided Feature Selection



Regulatory Region **Gene**

- Regulatory regions responsible for controlling gene activation status
- Majority of disease-associated variants sit in regulatory regions [Maurano et al, Science, 2012]
- Consider only 64 variants from each of these ($\approx 20\,000$) regions
- *But:* still more than 1 million sites remaining

Promoter-CNN

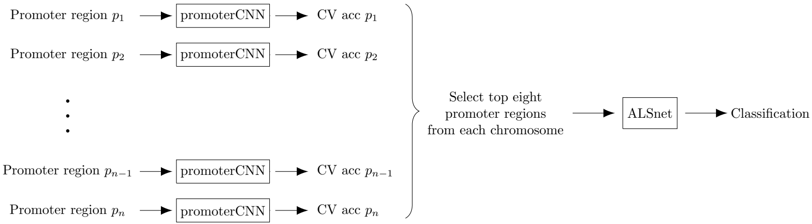
Architecture

Layer type	Description	Output shape
Input		$(64, 1)$
Convolution, BN and Act	1×1 filter, 4 output channels	$(64, 4)$
Convolution, BN and Act	4×4 filter, 32 output channels	$(61, 32)$
Reshape	Flatten	$(1952, 1)$
Dense, BN and Act		$(148, 1)$
Dense, BN and Act		$(16, 1)$
Output	Softmax	$(2, 1)$

- *Input*: Genotypes from promoter region $\leftrightarrow$ element of $\{0, 1, 2\}^{64}$
- *Output*: 0 for 'No ALS', 1 for 'ALS'

Feature Selection II: Promoter CNN

Workflow



- *Second Step*: Evaluate promoter variant blocks using (5-layered) 'Promoter-CNN' for being predictive of ALS
    - Keep only 8 highest-scoring regions per chromosome; discard all others
    - *Remaining sites*: 512 per chromosome ☞ Perfect!

UNIVERSITÄT BIELEFELD

Feature Selection Protocol: Advantages

- *Applying convolution sensible*: By laws of recombination, variants within promoter region inherited as haplotype block
- *Interpretability*: Reveal potentially relevant ALS genes

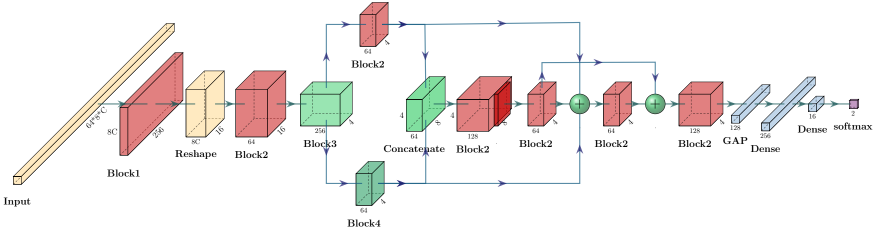
Feature selection addresses three key technical challenges

Left To Do

Construct deep CNN

Classification: ALS-Net

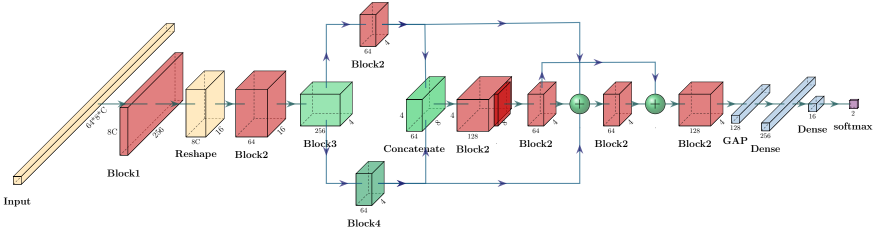
ALS-NET: ARCHITECTURE



Depth: 24 hidden layers overall

- *Block 1*: 2 x conv. + batch norm. (BN), *Block 2*: 3 x conv.
- *Reshape*: [Howard et al., 2017]: yields 16 × 16 × 32 ☞ convolution
- *Block 3*: 1 x separable convolution [Gao et al., 2018]
  ☞ saves on parameters, 1 x conv., 2 x pooling
- *Block 4*: 2 x conv. + 1 x separable conv.
- *Blue Arrows*: Bypass layers if needed, see [ResNet, 2015]
- *GAP*: Global average pooling
- *Dense*: Fully connected layer

ALS-NET: ARCHITECTURE

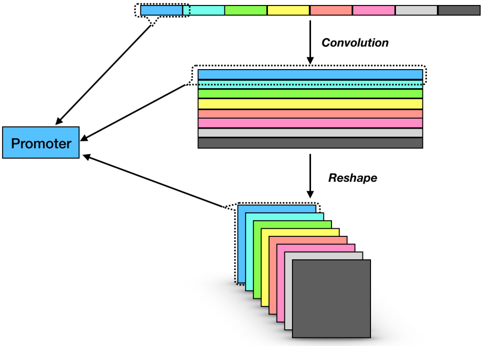


Depth: 24 hidden layers overall

- *Block 1*: 2 x conv. + batch norm. (BN), *Block 2*: 3 x conv.
- *Reshape*: [Howard et al., 2017]: yields 16 × 16 × 32 ☞ convolution
- *Block 3*: 1 x sep. conv. [Gao et al., 2018]
  ☞ saves on parameters, 1 x conv., 2 x pooling
- *Block 4*: 2 x conv. + 1 x separable conv.
- *Blue Arrows*: Bypass layers if needed, see [ResNet, 2015]
- *GAP*: Global average pooling
- *Dense*: Fully connected layer

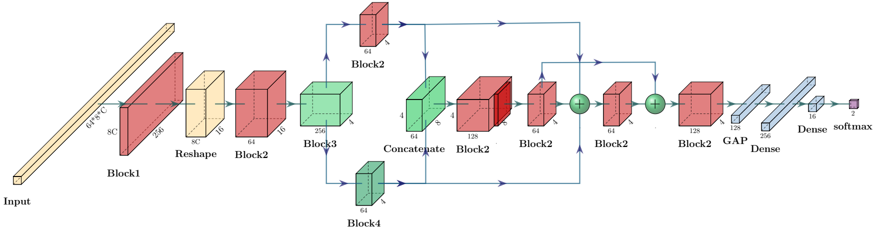
ALS-NET

RESHAPE FOR ONE CHROMOSOME



Interpretation: Each promoter makes a channel: “8x8-image” for each participating chromosome

ALS-NET: ARCHITECTURE



Depth: 24 hidden layers overall

- *Block 1*: 2 x conv. + batch norm. (BN), *Block 2*: 3 x conv.
- *Reshape*: [Howard et al., 2017]: yields 16 × 16 × 32 ☞ convolution
- *Block 3*: 1 x sep. conv. [Gao et al., 2018]
  ☞ saves on parameters, 1 x conv., 2 x pooling
- *Block 4*: 2 x conv. + 1 x separable conv.
- *Blue Arrows*: Bypass layers if needed, see [ResNet, 2015]
- *GAP*: Global average pooling
- *Dense*: Fully connected layer

Chromosomes 7, 9, 17 and 22

	Accuracy	Precision	Recall
Logistic Regression	73.9	75.9	69.9
Support Vector Machines	72.5	78.3	62.4
Random Forest	59.6	**81.3**	24.9
Ada-Boost	66.1	70.0	56.5
ALS-Net	**76.9**	71.1	**90.8**

- Recall: ALS-Net recovers substantially more cases
- Choice of chromosomes favors additive approaches [Van Rheenen et al., Nat.Gen., 2016]
- All methods required (here: CNN-based) feature selection: without feature selection no method runs on all 4 chromosomes
- *Important finding:* ALS-Net picks up less confounding variables (batch effects) than Logistic Regression

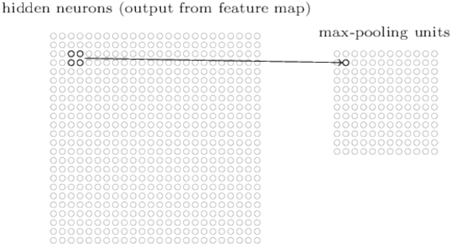
Capsule Networks

Motivation: Pooling in CNN's

- In addition to convolutional layers, CNN’s make use of *pooling layers*.
- Pooling layers generate *condensed feature maps*: it takes a rectangle of neurons, and summarizes their values into one value
- This generates a considerably smaller layer

Convolutional Neural Networks

Pooling Layers

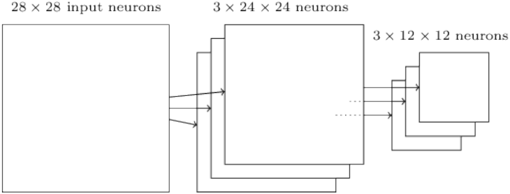


2×2 pooling

- *Max pooling*: Each $L \times L$ rectangle is mapped onto the maximum of its values
- *L2 pooling*: Each $L \times L$ rectangle is mapped to the rooted average of the squares of the values
- This overall yields a layer that is $L \times L$ times smaller
- Usually $L = 2$ is used

Convolutional Neural Networks

Combining Convolutional and Pooling Layers

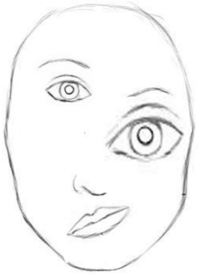


Convolutional layer followed by pooling layer

- Convolutional and pooling layers are used in combination
- Pooling layers usually follow convolutional layers
- *Intuition*:
  - The exact location of the occurrence of a feature is not important
  - Pooling helps to handle distortions and rotations

Capsule Networks
Vector-Valued Neural Networks

MOTIVATION

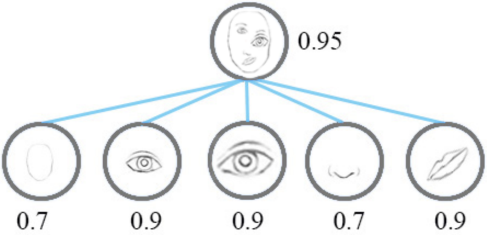


Recognized as face by CNN

"The pooling operation used in convolutional neural networks is a big mistake and the fact that it works so well is a disaster."

Geoffrey Hinton

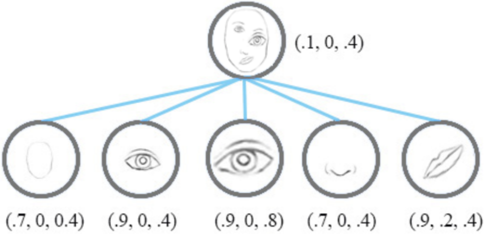
Motivation



CNN face recognition: interaction of neurons across layers

- Substructures form larger structures regardless of relative orientation and size
- *Issue:* Each neuron transmits information about existence only
- *Inverse image rendering:* Each neuron transmits information about existence, orientation and size

MOTIVATION



Capsules instead of neurons: existence of face has probability 0.1

- Capsules transmitting (probability, orientation, size)

CAPSULES VS CNNS: VECTORS VS SCALARS

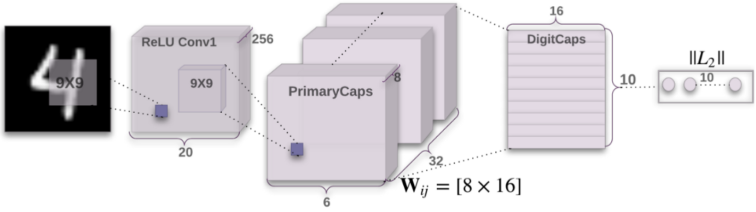
Capsule vs. Traditional Neuron			
Input from low-level capsule/neuron		vector($\mathbf{u}_i$)	scalar(x_i)
Operation	Affine Transform	$\widehat{\mathbf{u}}_{j\|i} = \mathbf{W}_{ij}\mathbf{u}_i$	–
	Weighting	$\mathbf{s}_j = \sum_i c_{ij}\widehat{\mathbf{u}}_{j\|i}$	$a_j = \sum_i w_i x_i + b$
	Sum		
	Nonlinear Activation	$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1+\|\mathbf{s}_j\|^2}\frac{\mathbf{s}_j}{\|\mathbf{s}_j\|}$	$h_j = f(a_j)$
Output		vector($\mathbf{v}_j$)	scalar(h_j)

Capsules: vectors; CNNs: scalars

From: Talk on CapsNets by naturomics

- **Pose matrices** W_{ij} learnt during backpropagation
- **Routing coefficients** c_{ij} supposed to replace pooling, determined during forward pass

General Advantages



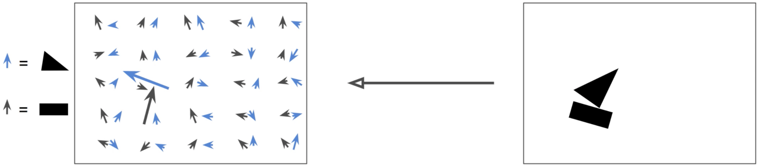
CapsNet Architecture for Digit Recognition

From: "Dynamic Routing Between Capsules", Sabour et al. (original paper; Figure 4)

- Primary capsule layer followed by "digit capsule" layer
- Primary capsules $\mathbf{u}_i$ code for image elements
- Digit capsules $\mathbf{s}_j$ code for digits
- *Recall:* $\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$ where $\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij}\mathbf{u}_i$
- Coding = probability, orientation, size, ... (whatever else makes sense)

Pose Matrices W_{ij}

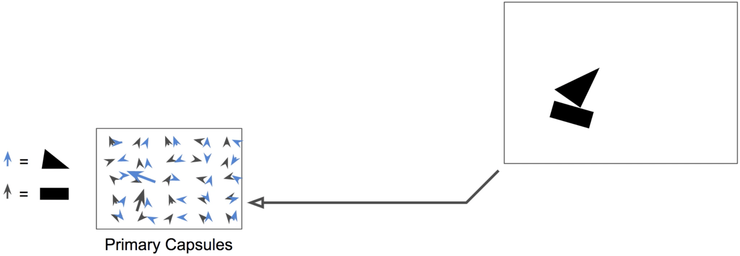
Vectors and Pose Matrices



From: Capsule Networks, Tutorial by Aurelién Géron

- *Length:* probability that object is present
- *Angle:* orientation of object at location
- *Line thickness, skewedness, gray scale etc:* could be captured by multi-dimensional vector

Vectors and Pose Matrices



From: Capsule Networks, Tutorial by Aurelién Géron

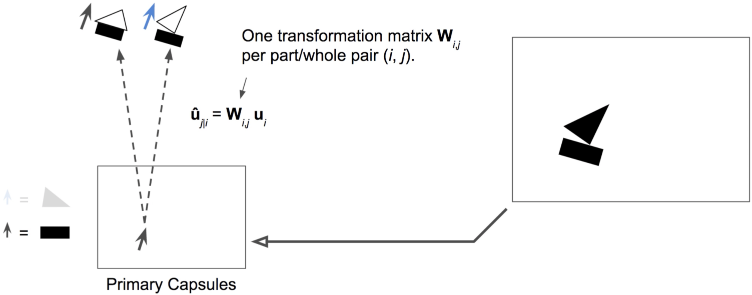
- Boat or house? How to answer question using vectors?

Notation

Attention

- *Lower layer:*
  - black = rectangle
  - blue = triangle
- *Higher layer:*
  - black = house
  - blue = boat

VECTORS AND POSE MATRICES

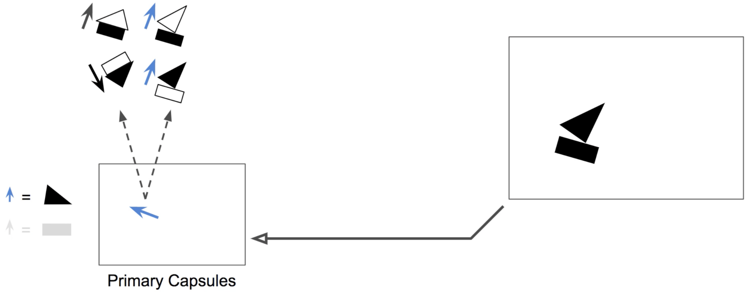


Probability and orientation of boat and house based on identified rectangle

From: Capsule Networks, Tutorial by Aurelién Géron

- Application of pose matrix $\mathbf{W}_{ij}$ to $\mathbf{u}_i$ determines orientation of higher layer object j, boat or house; here i represents *rectangle*
- So, $\hat{\mathbf{u}}_{j|i}$ reflects probability and orientation of boat and house based only on *rectangle*

Vectors and Pose Matrices

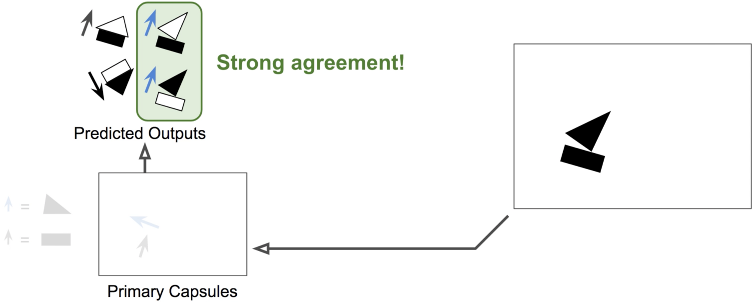


Probability and orientation of boat and house based on identified triangle

From: Capsule Networks, Tutorial by Aurelién Géron

- Consider lower layer object *triangle*; here i represents triangle
- Here, $\hat{\mathbf{u}}_{j|i}$ reflects probability and orientation of boat and house based only on *triangle*

Vectors and Pose Matrices



Prediction of rectangle and triangle agree for boat, but not house

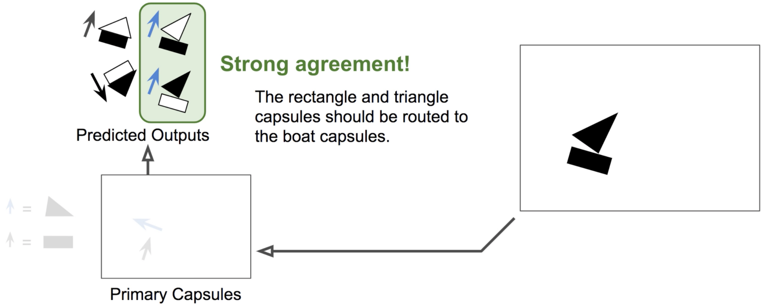
From: Capsule Networks, Tutorial by Aurelién Géron

How to Decide Further?

- Rectangle and triangle agree on boat, but disagree on house
  - should increase probability for boat to exist
  - should decrease probability for house to exist
- How to make this explicit?

Dynamic Routing

MOTIVATION



Prediction of rectangle and triangle agree for boat, but not house

From: Capsule Networks, Tutorial by Aurelién Géron

- *Solution:* Make rectangle and triangle capsule fire to boat, but not house

Capsules vs CNNs: Vectors vs Scalars

Capsule vs. Traditional Neuron			
Input from low-level capsule/neuron		vector($\mathbf{u}_i$)	scalar(x_i)
Operation	Affine Transform	$\widehat{\mathbf{u}}_{j\|i} = \mathbf{W}_{ij}\mathbf{u}_i$	–
	Weighting	$\mathbf{s}_j = \sum_i c_{ij}\widehat{\mathbf{u}}_{j\|i}$	$a_j = \sum_i w_i x_i + b$
	Sum		
	Nonlinear Activation	$\mathbf{v}_j = \frac{\|\|\mathbf{s}_j\|\|^2}{1+\|\|\mathbf{s}_j\|\|^2}\frac{\mathbf{s}_j}{\|\|\mathbf{s}_j\|\|}$	$h_j = f(a_j)$
Output		vector($\mathbf{v}_j$)	scalar(h_j)

Capsules: vectors; CNNs: scalars

From: Talk on CapsNets by naturomics

- *Explicit solution:* In $\mathbf{s}_j = \sum_i c_{ij}\hat{\mathbf{u}}_{j|i}$
  - c_{ij} should be small for j referring to house
  - c_{ij} should be large for j referring to boat

Routing Coefficients c_{ij}

- i corresponds to lower level, j to higher level capsule
- Each $c_{ij} \in [0, 1]$
- $\sum_j c_{ij} = 1$ for all i
- *Task:* For each i, determine predominant j:
  - c_{ij} really greater than zero only for little j
  - For each i, majority of c_{ij} close to zero

☞ Capsule i fires exclusively to selected, little capsule(s) j

- How to arrange for determining c_{ij} accordingly?

Dynamic Routing

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: $\mathbf{c}_i \leftarrow \texttt{softmax}(\mathbf{b}_i)$ ▷ 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 $(l+1)$: $\mathbf{v}_j \leftarrow \texttt{squash}(\mathbf{s}_j)$ ▷ squash computes Eq. 1
7: for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow b_{ij} + \hat{\mathbf{u}}_{j|i}.\mathbf{v}_j$
return $\mathbf{v}_j$

Routing Algorithm: Pseudo Code

From: “Dynamic Routing Between Capsules”, Sabour et al. (original paper)

- *Softmax:* $c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$; turns b_{ij} into “probabilities” c_{ij}
- For large c_{ij}, agreement (by scalar product) of $\hat{\mathbf{u}}_{j|i}$ with $\mathbf{v}_j$ required
  - $\hat{\mathbf{u}}_{j|i}$ prediction of j to exist by i alone
  - $\hat{v}_j$ prediction of j to exist overall
- *Line 7:* Iterative update in that respect

Dynamic Routing

Procedure 1 Routing algorithm.

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

Routing Algorithm: Pseudo Code

From: “Dynamic Routing Between Capsules”, Sabour et al. (original paper)

Large c_{ij}: large scalar product of $\hat{\mathbf{u}}_{j|i}$ with $\mathbf{s}_j$ required

Dynamic Routing

Procedure 1 Routing algorithm.

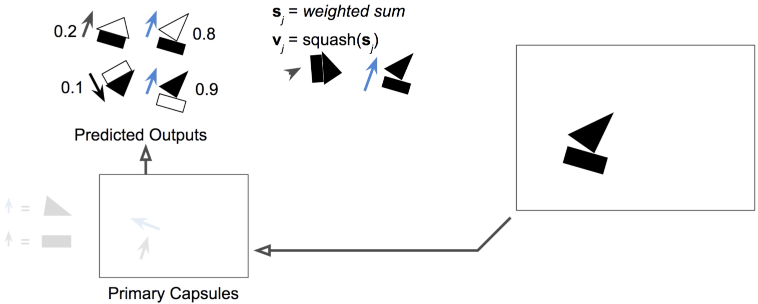
procedure ROUTING($\hat{\mathbf{u}}_{j|i}, r, l$)
    for all capsule i in layer l and capsule j in layer $(l+1)$: $b_{ij} \leftarrow 0$.
    for r iterations **do**
        for all capsule i in layer l: $\mathbf{c}_i \leftarrow \mathtt{softmax}(\mathbf{b}_i)$ ▷ softmax computes Eq. 3
        for all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

Routing Algorithm: Pseudo Code

From: "Dynamic Routing Between Capsules", Sabour et al. (original paper)

- Important: Routing algorithm run during forward pass
- Routing coefficients determined for each data point individually
  - Both for labeled (training) data and unlabeled data (prediction)
  - *Note:* This is considerably slower than a forward pass in CNN's

Dynamic Routing

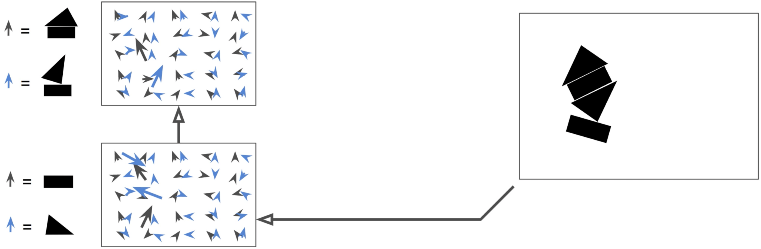


Situation after Routing: Clear Vote for Boat

From: Capsule Networks, Tutorial by Aurelién Géron

Advantages of Capsule Networks

ADVANTAGES: CROWDED SCENES

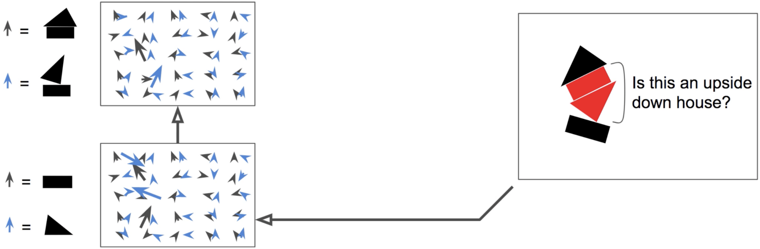


Capsules Resolve Crowded Scenes

From: Capsule Networks, Tutorial by Aurelién Géron

- *Either:* Boat and house?
- *Or:* House upside down?

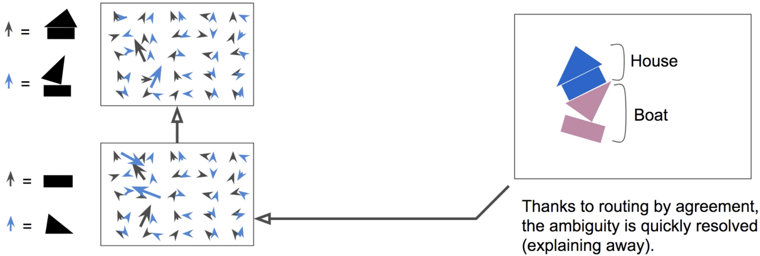
Advantages: Crowded Scenes



Capsules Resolve Crowded Scenes

From: Capsule Networks, Tutorial by Aurelién Géron

Advantages: Crowded Scenes

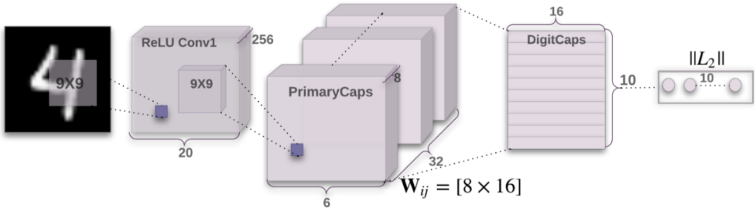


Capsules Resolve Crowded Scenes

From: Capsule Networks, Tutorial by Aurelién Géron

- Dynamic routing iteratively emphasizes boat and house, instead of house upside down

General Advantages

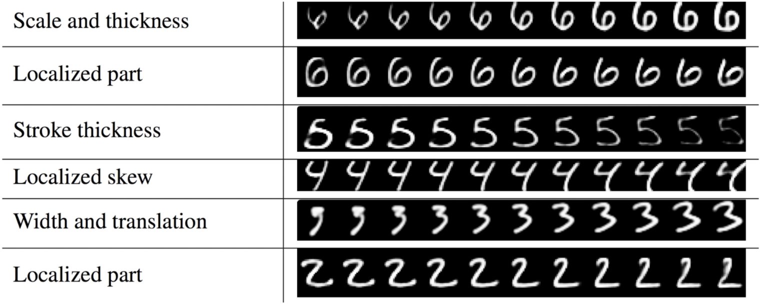


CapsNet Architecture for Digit Recognition

From: “Dynamic Routing Between Capsules”, Sabour et al. (original paper; Figure 4)

- Low depth sufficient
- Require little training data
  ☞ In CNNs, training image for each angle required

Advantages: Interpretation of Low Level Capsules



Low Level Capsules (PrimaryCaps) enjoy Human Mind-Friendly Interpretation

From: “Dynamic Routing Between Capsules”, Sabour et al. (original paper; Figure 4)

UNIVERSITÄT
BIELEFELD

Disadvantages

- Tend to see "too much" in larger images
  - losses in prediction accuracy
  - differently initialized training runs unstable
- Training is slow
- Implementation more complex

References

- ALS-Net:
  - `https://academic.oup.com/bioinformatics/article/35/14/i538/5529261`
- Capsule Networks:
  - `https://proceedings.neurips.cc/paper/2017/file/2cad8fa47bbef282badbb8de5374b894-Paper.pdf`

OUTLOOK

- Disease Capsule
  - Motivation
  - Methods
  - Results
- Attention Mechanisms I
  - Basic Idea
- Sequence-2-Sequence Models
  - *Motivation:* Translating Languages
- Attention Mechanisms II
  - Transformer Architecture

Thanks for your attention