EM

thedanielschwartz

June 2019

1 CNN

- CNNs need to use the same knowledge at all locations in the image
  - Ties weights of feature detectors so that features learned at one location are available at other locations
  - Convolutional capsules share knowledge across locations to include part-whole relationships to characterize a familiar shape
    - Aim of capsules is to make good use of this underlying linearity for dealing with viewpoint variations and improving segmentation decisions
  - Capsule high-dimensional coincidence filtering
    - Familiar object can be detected by looking for agreement between votes for its pose matrix
    - Votes come from parts that have been detected
      - Part produces a vote by multiplying its own pose matrix by a learned transformation matrix representing that viewpoint between the part and the whole
      - As viewpoint changes, pose matrices of parts and whole change in a coordinated way

2 Assigning parts to whole

  - Use fast iterative process "routing-by-agreement"
    - Updates probability with which a part is assigned to a whole
    - Based on the proximity of the vote coming from that part of the votes coming from other
    - parts assigned to that whole
    - Powerful segmentation principle that allows knowledge of familiar shapes to derive segmentation

3 Difference between capsule and CNN

- Activation of a capsule is based on a comparison between multiple incoming pose predictions
- Standard net is comparison between single incoming activity vector and learned weight vector

4 How Capsules Work

- NNs use simple non-linearities where a non-linear function is applied to the scalar output of a linear filter
- Capsules use complicated non-linearities that convert a whole set of actiavation probabilities and poses of capsules in one layer to activation probabilities and poses of capsules in next layer

5 Architecture

- 5x5 Convolutional layer with 32 channels and stride of 2 with ReLU non-linearity
- Several layers of capsules
  - Set of capsules in layer L, Ω_L
  - Each capsule has:
    - 4x4 pose matrix, M
    - Activation probability, a
  - Between layers:
    - 4x4 trainable matrix, W
    - These matrices and two learned biases per capsule are stored and learned discriminatively
  - Activations of primary capsules are produced by applying sigmoid function to weighted sums of same set of lower-layer ReLUs

6 EM Process

- Pose matrix of capsule i in layer L is transformed by W_{ij} to cast a vote in $V_{ij} = M_i W_{ij}$ for the pose matrix of capsule j
- Poses and activations of all capsules in layer L+1 are calculated by using non-linear routing procedure
  - Input: V_{ij} and a_i for all i $\in \Omega_L, j \in \Omega_{L+1}$
  - Steps (Expectation Maximization):

        - Iteratively adjusts the means, variances, and activation probabilities of capsules in layer L+1
        - Assigns probabilities between all capsules $i \in \Omega_L$ and j $\in \Omega_{L+1}$
    * Each capsule in the higher-layer corresponds to a Gaussian
    * Each pose of the active capsule in the lower-layer (vector) corresponds to a data-point
    * Use Minimum Description Length principle to choose whether to activate a higher-level capsule
        - Choice 0: Do not activate it
        - Pay a fixed cost of $-\beta_\mu$ per datapoint for describing poses of all lower-level capsules assigned to higher-level capsules
        - Cost is the negative log probability density of the data-point under an improper uniform prior
    * Choice 1: Activate higher-layer capsule
        - Pay a fixed cost of $-\beta_a$ for coding its mean and variance
        - Use negative log probability density of that datapoint's vote under the Gaussian distribution fitted by whatever higher-level capsule it gets assigned to
- Dynamic routing is performed by two adjacent layers of capsules (higher-level and lower-level)
    * Complete routing between one pair of layers before starting routing between next pair of layers
    * Routing process has strong resemblance to fitting mixture of Gaussians using EM
        - Higher-level capsules play role of Gaussians
        - Means of activated lower-level capsules for a single input image play role of datapoints
    * Procedure 1: Routing algorithm
        - Returns activation and pose of capsules in layer L+1 given activation and votes of capsules in layer L

6.1 EM Algorithm

- Alternates between E-step and M-step
    * E-step: used to determine for each datapoint, probability which it is assigned to each of the Gaussians
        - Assignment probabilities act as weights
        - Adjusts assignment probabilities for each datapoint to minimize "free energy"
        - Free Energy: expected energy minus the entropy

- Can minimize expected energy by assigning each datapoint with probability 1 to whichever Gaussian gives the lowest energy (highest probability density)
- Maximize entropy by assigning each datapoint with equal probability to every Gaussian ignoring the energy
- Boltzmann Distribution: Assigning probabilities proportional to $exp(_E)$ (best trade-off)
- M-step: finds the mean of these weighted datapoints and variance about the mean
- Adjusts each Gaussian to maximize the sum of the weighted log probabilities that the
- Gaussian would generate the datapoints assigned to it
- Negative log probability density of a datapoint under a Gaussian can be treated like the energy of a physical system that M-step is minimizing the expected energy where expectations are take using assignment probabilities
- Mixing proportions are set to fraction of data assigned to Gaussian

6.2 Softmax function

* Computes distribution that minimizes free energy when logits are viewed as negative energies
    - Minimizing free energy: Using softmax in routing procedure to recompute assignment probabilities
    - Refitting Gaussian model of each capsule provided the logits of softmax are based on same energies as are optimized when refitting Gaussians
    - Energies used are negative log probabilities of votes from lower capsule to higher

6.3 Spread Loss

* Make training less sensitive to initialization and hyper-parameters
* Equivalent to hinge loss with m = 1
* Maximizes gap between activation of target class and activation of other classes

6.4 General Capsule viewpoint

* Unit is not activated based on matching score with a filter (fixed or dynamically changed during inference)

* A capsule is activated only if the transformed poses coming from the layer below match each other
    - Fewer parameters
    - Generalizes better

6.5 Mixtures of Transforming Gaussians

* Normally Gaussians only have a subset of datapoints assigned to it but all see the same data
* View capsules in higher layer as Gaussians and means of active lower-layer capsules as the dataset, each Gaussian sees a dataset in which different datapoints have been transformed by transformation matrices
    - Matrices are different for different Gaussians
    - Transformation matrices learn discriminatively in an outer loop
    - They are restricted to dynamic routing to modifying the means and variances of Gaussians
    - Prevents solution where transformation matrices collapse to zero and transformed data points are identical

6.6 Mixtures of Switchable Transforming Gaussians

- Each transforming Gaussian has an activation parameter which is its probability of being switched on for a given dataset
- Probability is determined by "description length" (energy)
- Difference in two description lengths is then put through a logistic function