11-785 01/06/2020

L8 - CNNS

(*) Next 4 lectures (2 weeks) → scanning for patterns (CNNS)

(*) Scott Fahlman - guest lectures

(*) New problem - spectrogram

(*) From this data, can we check if the signal contains 'welcome'

BR:- Yes, as on I/O problem

(*) Issue:- Two signals have welcome stated at diff. locations

(Q): will MLP trained on signal with 'welcome' give the same classification if the signal with 'welcome' is shifted in time?

- NO. Covariates have different values; no guarantee.
- consider through vector representations

BR: We want network to fire, regardless of location → shift/translation invariance

---

(*) Visual analogone

- A flower 'shifted' within a picture
- Patterns lie on different subspaces

(Ans): we require shift invariance for both speech and visual recog.

(*) Solution:- use scanning i.e. a sliding window over inputs and classify each window.

How to combine classification over each window frame to give a classification

- use max or or or logistic

(*) BR: Switched issue of whether recording contains word welcome to one of scanning and then combining outputs.

Scanning with an MLP

- (9) (11) - see pseudocode ✓
- can view as one large neural network
- NN composed of many subnets (same one)
- subnets are identical
- This is a shared parameter network

(*) scanning to find a flower

- we have a classification for each window of the picture
- pass all of these intern class through a max/softmax.

(*) scanning image pseudocode.

(10) (A2) - Go through subnet 1

- For MLP, not just scan L→K but T→B.

(*) Network training

- constraint of shared parameters → each lower-level subnet that classifies window frame are identical

(*) any update of param of one copy → update all copies.

(*) Learning in shared param networks

- shared params → arbitrarily deciding that some params. have same value

(11) e.g. $W^R_{ij} = W^1_{mn} = W^S$

(11) for training instance X, small perturb. of W^S perturbs both W^R_{ij} and W^1_{mn} identically

- each perturbation will marginally influence $Div(y, \hat{y})$

- Attempting to estimate common value ws

(*) computing divegence of shared params.

(13) - Review influence diagram ✓
of Divergs

(*) computing divegence of shared params ✓

(44) - Review - formula

(*) Training a network with shared params.

(AK) - pseudocode + detail - review

- slight modification on backprop (76 slides in 18 mins)

story so far

(*) position-invariant pattern class → scanning

(*) Scanning ⊖ composing a lage network with repeating subnets.

(*) Param estimation → backprop rules must be modified to combine gradients from param that share same value. (applies to all networks with shared param.)

(*) Scanning - close look.

- visual illustration → very clear

- order of computation/operations. (scanning)

- for W windows of spectrogram.

- W_1 - window 1 - pass through 4 unit, 2 unit, 1 unit layes
- W_2 —11— 2 —11—
- ⋮
- W_W - window W

↳ pass all through softmax

(611) window by window

(*) Different order

- pass each window through individual unit (by unit), layer by layer.
- does not change output
- (16) - ~~soti~~ review. ✓

(*) Save computation

- columnwise vs row wise - both scanning MLPs

- original pseudocode
  - scan L→R, pull out segment, MLP it; compute softmax
- different order pseudocode. → (16) pseudocode ✓
  - Run through neurons

(*) can switch for loop nesting order; does not change comp.

(*) scanning with MLP - vector notation → (17) (011) ✓

(*) scanning flower image

1) window by window (entire MLP)
2) neuron by neuron (scan entire image with one neuron)
   - output for that neuron for every window of image. (scan image) → ~~map of~~
   - do same for every neuron
   - we then have a map of the image at every neuron
     (i.e. a collection of outputs over windows at every neuron)

(*) to find whether there is a flower / classify a patch

(011) pick those locations from 4 maps; pass it through MLP.

(011): note two equivalent ways of conducting scan operations

(*) process all neurons in one location

(*) process all locations with each neuron

(*) logic can be reversed at next layer.

(*) Scanning - 2nd layer neurons / 3rd / output?

- At next level
- scan outputs of 1st layer

(*) to find if there is a flower in a particular location, pick the location from the maps ...

(A8) - review logic - cancel (A11)

(*) Redrawing final layer

- still scanning MLP, except change in comput. order; but still scanning MLP with shared params.

- (A9) - changes in pseudocode, comparison of window by window, neuron by neuron

BR: Forward idea of MLPs/NNs capturing progressively more abstract/complex patterns hierarchically

---

(*) Distributing the scan/behavior of the layers.

- 1st input → pixel by pixel (:)

(A11) First layer neurons resp. for evaluating entire block of pixels.
- subsequent layers look at single pixel on input maps

(A11) - distribute scanning over 2 layers
(to achieve some block analysis at 2nd layer)
- First layer → smaller block of pixels
- Next layer → blocks of output from first layer.
- effectively evaluates large block of original image

(*)
- still scanning with a shared param network
- with minor modification. → (A11) - clarity on parameter sharing

(~) (A10) (S11) - logic revise

- (A11) - smearing information over layers
  - recursively decomposing patterns over layers

(*) Hierarchical build up of features.
- scanning and combing outputs as giant shared MLP
- individual nets have shared param sets → does not change.

- (A12) pseudocode.
- can re-order computation again. → but size of maps change. (A13) pseudocode

- Affine combination → review details
  - inner product between weights and inputs with transl.
(*) (S11) (A14) - This is a convolution → convolutional neural network

- NN with scanning
- typically, scan input at finer and finer resolutions as we go through the network.

(*) CNN - vector notation (A15)

(*) Why distribute?

(S1) distribution forces localised patterns in lower layers

(S2) (why not one simple scanning MLP that scans entire image?)

(*) scanning without distri
- need to print new slides S.2020 significantly different

(*) why distribute?

- distrib. - localised patterns in lower layers
- no. parameters - large reduction (OOM) reduction in param.
  - significant shared computation gains.
- Regardless of how scanning is distr. computationally; still a scanning MLP.

(*) story so far

(*) operations in scanning input with full network can be reordered

- scanning the input with individual neurons in the first layer to produce scanned 'maps' of input
- Jointly scanning the 'map' of outputs by all neurons in previous layers by neurons in subseq. layers

(*) Scanning block can be distr. over multiple layers of network

- significant reduction in total no. param.

(*) Hierarchical composition - different persp.

1) first layer → sub regions of main image (eg. petals)
2) 2nd layer → regions of output of first layer.
   - Put petals → flower
   - Large region of origin. input image

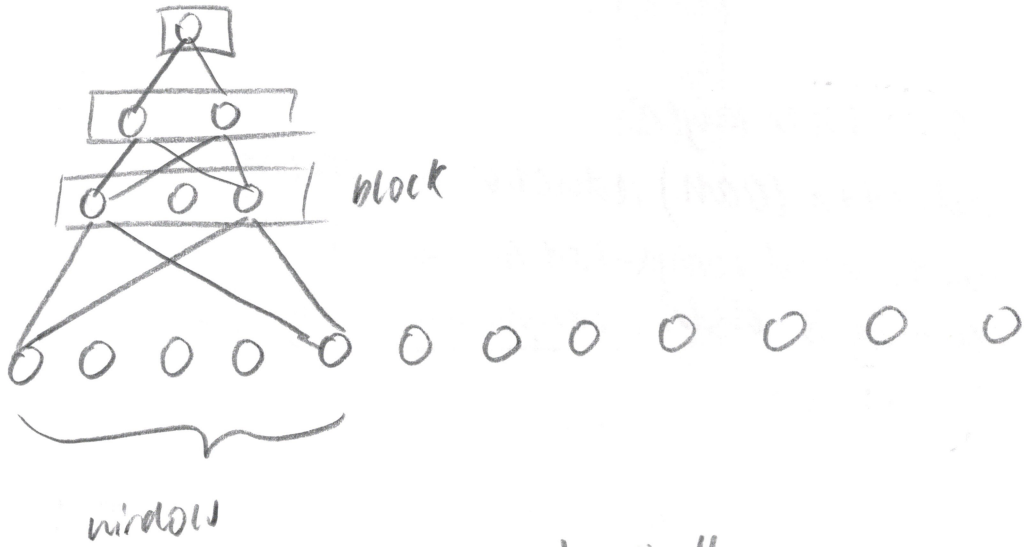
(*) Each of the scanning neurons as a filter

- A correlation filter
- Each filter scans for a pattern in the map it operates on

(*) Pattern in the input image each filter sees → receptive field

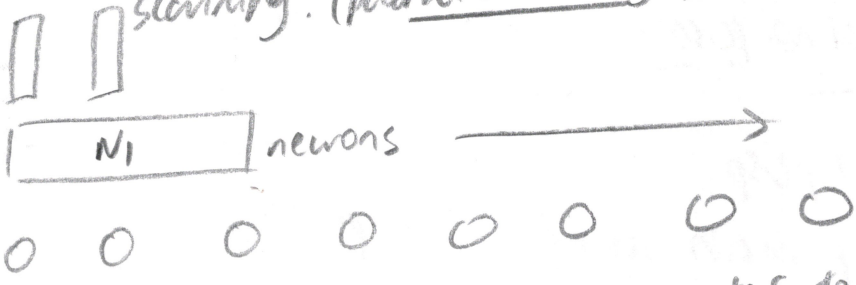
- First layer filter: actual receptive field is simply arrangement of weights
- Higher level filters: - receptive field not obvious, must be calculated.
  - may not have natural correspondences due to complex combos.

(*)



- Redraw neurons in a block (layer) vertically
- $N_1 \rightarrow N_2$ - 6 connections ($N_1 N_2$)
- $N_2 \rightarrow N_3$ - 2 connections ($N_2 N_3$)

(A16) → check/review formulae on no. param for generic scanning vs distributed scanning. (parameter savings) calculation → calc.



yellow columns
- snapshot
- columns of N_1 neurons scanning 2 IKo patterns at a time.

(o11): distributed scanning ⇒ fewer parameters to estimate

(o11): large additional benefit from the fact that scans at neighbouring positions share the computation of low-level blocks

(o11): note unique parameters

-(o11): re-use of computation → more efficient

(*) Go through 2D case in your review

(A17) - 1034 parameters vs 160 parameters (distrib. scanning)
(non-distri scanning)

(*) Modifications

- first layer may feed into MLP rather than single neuron.
- shared param net.

(*) mod 1 - conv 'stride'

- stride is key concept (for CNNs)
- Pseudocode (A18)

(*) Accounting for jitter

- minor noise

(*) similar principle

Is there a flower in image?

- scan entire input, take a max of all outputs → tells us whether there is a flower or not

↓

(*) Small jitter acceptable

- replace each value by maximum of values within a small region around it
- max filtering /pooling
- max is just an activation

Or equivalent to adding a layer with weights all one; activation is max. that follows previous layer

(*) does not change network as shared param.

(*) max pooling 'strides'

(*) max operations may stride by more than one pixel

- shrinks map
- operation is called 'pooling'
- consists of pooling a no. of outputs to get a single output.
- stride > 1 → downsampling

(*) next layer works on max-pooled maps

(*): overall structure

- many layers of convolution (scanning) followed by max pooling (and reduction) before final MLP.

(*) individual perceptrons at any scanning/convolutional layers are called filters.

(*) filter image (input) to produce output image (map)

(*) Indiv max operations → max pooling filters

(*) Entire structure → CNN

- scanning for patterns, distributing manner in which they are represented across network.

(*) shift invariance

- LeCun
- MNIST digit scanning (not on individual frames)

(*) 1D convolution

- speech
- still convolution (but across time)
- Representation:- spectographic time frequency components
- NO jitter (and max pooling) allowance in speech

(*) story so far - summary