6.3. L-Model Backward

Backward func. for whole net.: iter[ate] all hidden layers backward, starting L; on each step, use cached values for prog. through l.
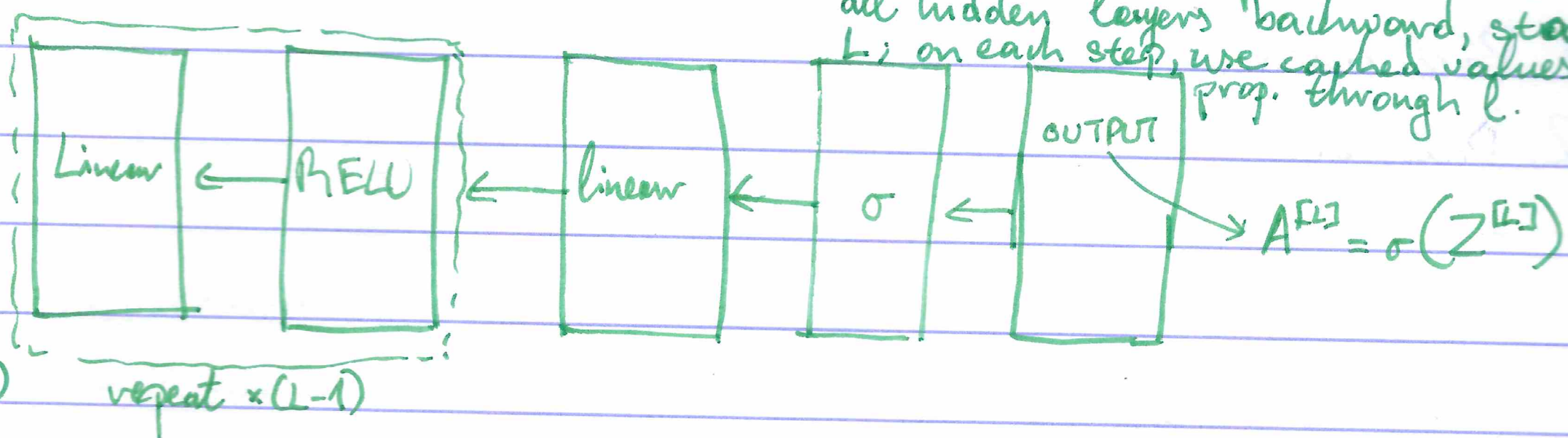

caches: list of caches containing:
- every cache of linear_activation_forward() w/ relu (caches[l] for l in range(L-1))
- cache of linear_act..._forw() w/ sigmoid (caches[L-1])

repeat × (L-1)

$A^{[L]} = \sigma(Z^{[L]})$

· Need to compute: $dAL = \dfrac{\partial L}{\partial A^{[L]}}$

$dAL = -(np.divide(Y, AL) - np.divide(1-Y, 1-AL))$

· Use dAL to keep going backward
  - Feed dAL into LINEAR→SIGMOID backw. (which uses c...
  - Iter. through all other layers using LINEAR→R[ELU]
  - Store each dA, dW, db into grads (grads ["dW" + str(...

  grads ["dA" + str(l+1)]

6.4. Update parameters

$W^{[l]} := W^{[l]} - \alpha\, dW^{[l]}$
$b^{[l]} := b^{[l]} - \alpha\, db^{[l]}$
for l = 1, 2, ..., L

Compute updated params store them in the param[s]

Week 4 Programming Assignment #2: Deep NN - Application

- Deep NN classifying cats vs. non-cats
- Objectives
  · Learn how to use all the previously written helper functi[ons] to build a model of any structure
  · Experiment w/ different model architectures
  · Recognise that it's always easier to build my helper functions before attempting to build a NN from scratch