ECE 1786 Lecture #3

Work-in-Flight: Assignment 2 - Classification of Language, due Monday Oct 9
Next Week: will discuss the structure and timing of the course project

Last Day: How Word Embeddings are Trained

Today: Classification of Language using word embeddings

Assignment 1 due last night, latest to hand in is tonight at 9pm, with penalty

Two Interesting Media to keep your eyes (& ears) on:

1. "The Batch" a weekly newsletter from Andrew Ng, and AI luminary:
    - https://www.deeplearning.ai/the-batch/
    - very interesting, timely, tightly written about AI, much about LLMs

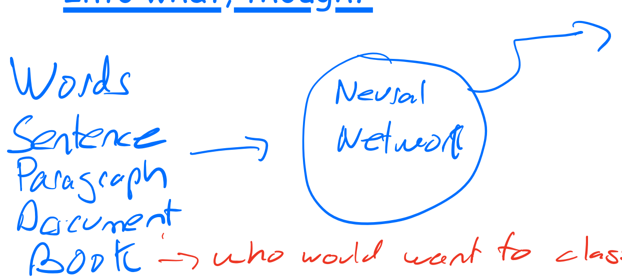
2. The Practical AI Podcast - https://changelog.com/practicalai
    - Two practitioners talking about/interviewing on many interesting topics
    - many recent episodes on Large Language Models
    - one of the principals has a new startup in the area.

After Assignment 1, we now know/see how embeddings respect meaning

- Two words with the same meaning would have similar embeddings, so you don't have to have a program deal with specific words
- But: One word with multiple meanings: still a problem -> later

A key general application in machine learning is classification

- You have seen picture classification in your prior ML class(es)
- Now, we'd like to classify words, sentences, paragraphs and documents
    - Into what, though?



1. Sentiment - positive or negative
   - range from -1 to +1 (regression)

2. Named Entity Recognition identify something that can described in many ways
   - E.g. a specific reason for quitting smoking: "makes me calm" = "relaxes me"

3. Style of talking - change talk vs. sustain talk in Motivational Interviewing

4. Politics - left vs. right; 5. Depression/Anxiety in Speech. 6. Suicidality
7. Lawyers looking for specific facts/privilege

- Now that we have text represented as embeddings, we can use a Neural Network to work with it/classify it

In Assignment 2, you'll be training two types of networks to detect if a sentence is either 1. objective (a statement of fact) or 2. Subjective (an opinion).

- you'll also look inside the network to see what it is learning

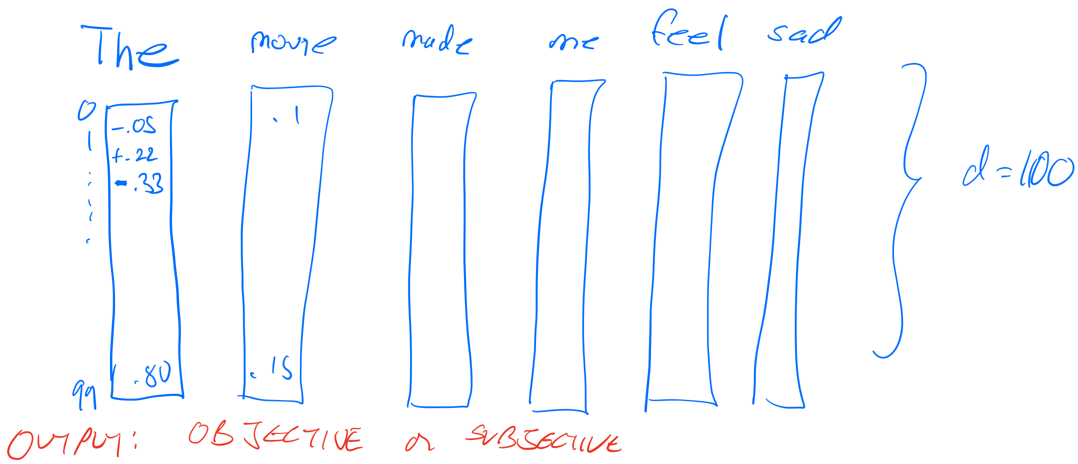
- Dataset used for training: Pang & Lee @Cornell created the subjective sentences from movie reviews, assumed to be subjective (but may not be)

- The objective sentences are taken from plot summaries from IMDB (the Internet Movie Database) - assumed to be objective, but also may not be

- Each of these sentences begin as regular ASCII text; the first processing step is called 'tokenization' which often (but not always) breaks words into smaller units. (The lemmatization in A1 was a little like this).
- Each of those units will need an embedding; in the case that there is a word that doesn't break down into known units, it will be assigned the 'unknown' token and embedding.

- Assignment 2 uses the GloVe embeddings, but with dimension d=100 this time

- So the input to the models in A2 is a sentence of words that is converted into a sequence of embeddings:

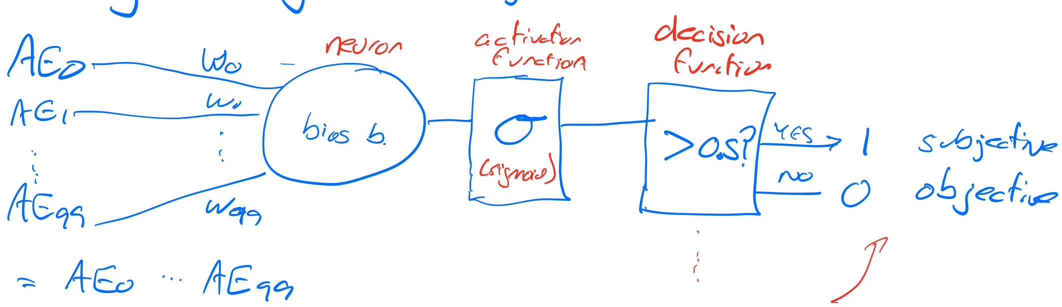


- Notice that input sentences like the one above would be of different lengths; but some neural nets are set up for fixed-sized inputs
  - In that case, must "pad" inputs with zeroes (i.e. embeddings that are zero) to make a batch of all equal size length inputs. Why zeroes?
  - This applies to the CNN, (Method 2 below) but not Method 1

<u>METHOD 1: Single Neuron</u>

- In A2, the first model you'll create and train will be a single neuron
  - We'll use this as a baseline to compare to
- The computation of the neural network will be:
  - Compute the average embedding across all words - what does this accomplish?
  - Gives a fixed-length vector of size 100
  - Feed the result into a single neuron

- Let the average average embedding (across sentences) be $AE_0 \ldots AE_{99}$



i.e. Input = $AE_0 \cdots AE_{99}$

Compute $\sigma\left(\sum_{i=0}^{99} w_i AE_i + b\right)$ --- if ≥ .5

- You'll train this neural network - the single neuron - with the given dataset 3-4

- It works surprisingly well!
- In other applications that process sentences, the average was commonly used

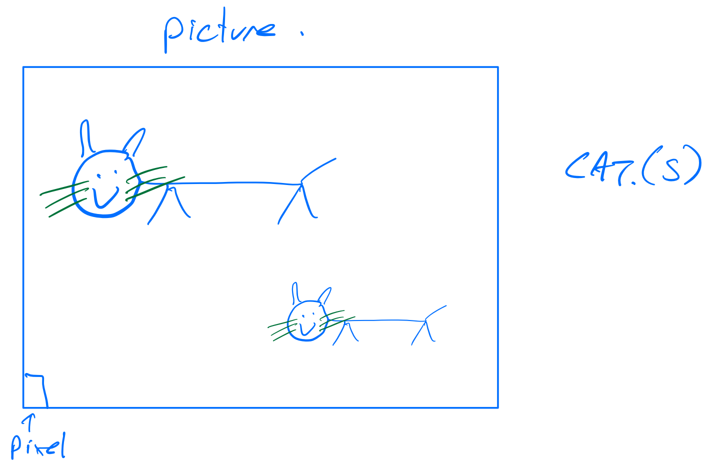
- Very interesting: notice that the trained weights - W0->W99 is a vector of the same size as the embedding. Does it have meaning in the same 'space' as the embedding space? You're asked to explore this in Assignment 2. How?

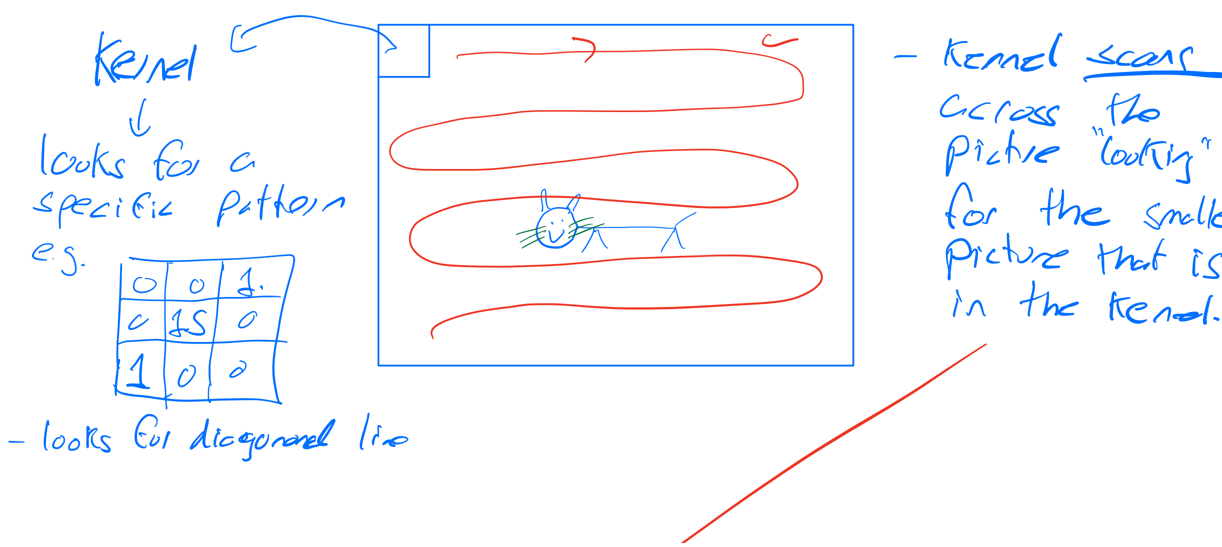
<u>METHOD 2: Convolutional Neural Net</u> (<u>CNN</u>)

Let's first spend a few minutes reviewing CNNs

Recall that pictures are made up of pixels, each pixel is possibly three numbers, the amount of Red, Green and Blue in the pixel

  - Say a picture is 1000x1000 pixels:

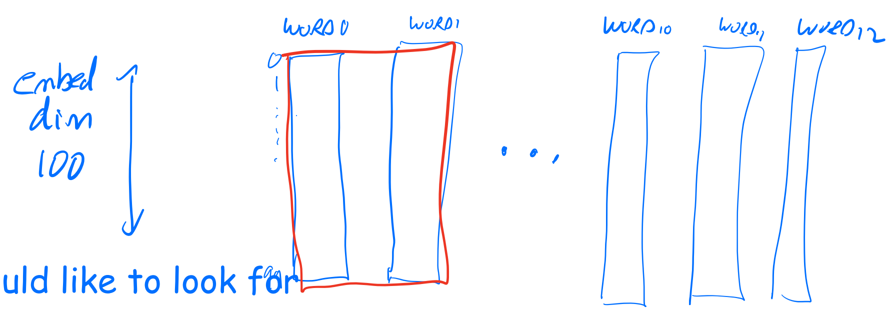




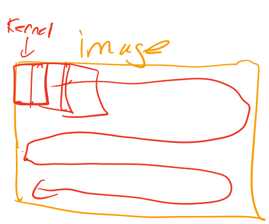
- the output from a kernel is a new array/picture/feature map
- it is the dot product (convolution) of the kernel with each 3x3 sub-image of the image.
- dot product ≡ convolution is a way to detect if the kernel "matches" the sub-image.
- higher dot product (a single value) means greater match.

- during training the kernels are learned
- there are lots of kernels.

Now, back to the language classification problem; consider the input sentence(s) embedding sequence:



- We would like to look for
1) Single words that indicate subjective/objective
2) Pairs of words, triplets, 4 words ….?
- In general to look for K words
- In CNN-terminology, we'd say that we want to train Kernels of size Kx100
- In a similar way that CNN CV kernels "scan" (sweep) across the field of a picture, these a kernel would 'scan' across a sentence



→ kernel is multiplied + accumulated (convolved) with image — dot product
→ remember that during training, the kernel is learned

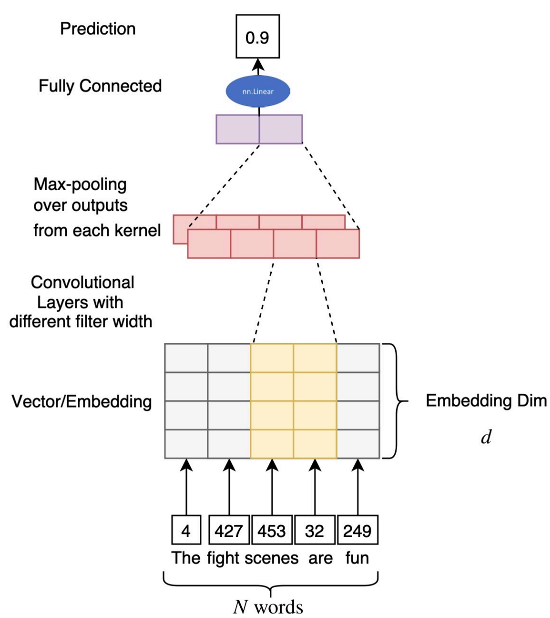
- In the context of classifying the language sentence, the kernel, which would be of size say 2x100, would just scan across the sentence word embedding once.
- It would be trained to look for a 2-word pattern of meaning that would contribute to learning the labels, subjective/objective

- Is one kernel enough? Maybe not. What size(s) should K be?

- Assignment 2 suggests having n1 kernels of size k1 x100 and n2 of size k2x100
    - Each one randomly initialized, as all weights/kernels are before training

What is the output size that you get from an input sentence of N words a kernel size of k1=2x100? (Where 100 is the embedding dimension?); assume the stride=1. (what is stride?)

- Get N-1 values out; if you have n1 such kernels, then you get n1 x (N-1)
- In general for kernel of size k, you get N-k+1 values



Yoon & Kim (paper referenced in A2) suggest choosing the maximum across all N-k+1 values

- i.e. maxpool
- Then feed all of those maximum values, from all the kernels into a multi-layer perceptron (MLP) - also called linear, fully connected layer(s).

Does this make sense?

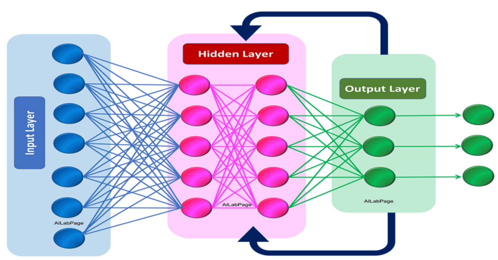
- Should look for patterns of 1-6 words that give sense of whether sentence is objective or subjective

- How might you know what the kernels are trained to looked for? I.e. what do they learn? (Discuss)
  - You're asked to explore this in assignment 2
  - These should be textual features just as a vision CNN has visual features in its kernels

- Important note in A2: for Method 2, I also asked you to enable the training of the embeddings themselves. That is, the gradient descent is set to propagate back into the embeddings, so that they learn to do this task as well
  - This is the case for the next network - Transformers - that we'll use

---

- Aside: Recurrent Neural Networks (RNNs) had traditionally been used for Natural Language Processing
  - An RNN typically takes 1 embedding in at a time, and has a cycle in which the output feeds back into the network, along with the next input:



  - RNNs (LSTMs, GRUs) were always rather problematic in that convergence of training was difficult to achieve and unreliable.
  - Also, to me, the fact that all information was "pinched" through the size of the embedding meant that lots of information was lost
  - The next (and central) topic of this course is the Transformer neural network, which keeps much more information 'alive' in parallel
  - Transformers are closer to CNN's in that sense, and they have essentially replaced RNNs for NLP