Lec 2 Word Representations, Text Classification

Recall

- Bag of word : one-hot every word & sum all one hot vecs in sentence

Subword Modelling

Handles conjugations!

companies expanding   compan ies expand ing

Good: - share params
- save space

{ Zipf's law, handle out of vocab tokens

Problems: - lack expressiveness if two few words
- if pieces too short, hard to compute
- need to balance subword length

Byte Pair Encoding ← LAMMA, GPT

{ seems like greedy Huffman info theory?

1. Start char level
2. Merge most frequent pairs
3. Iterate

Unigram Models

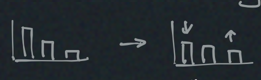
{ Import SentencePiece
BPE / Unigram similar result
60k-80k good for English

Use unigram LM that can generate sentence
Pick a vocab size that maximises likelihood of corpus

$$P(X) = \prod_{i=1}^{|X|} P(x_i)$$

Segment sentence by maximising $P(X)$

Limitations - Multilinguality is hard — since things depend on frequency, but data dist wrt long not always balanced
→ Can workaround by reweighting



- Arbitrary — "e st" or "es t"
→ workaround: subword regularisation

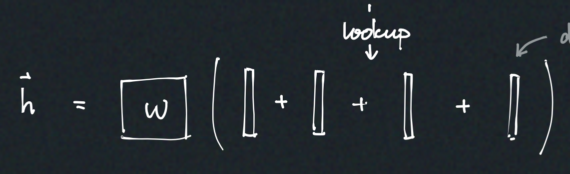
{ If adding new vocab, can do probability stuff and interpolate
$\lambda P_{old}(x) + (1-\lambda) P_{new}(x)$

Embedding

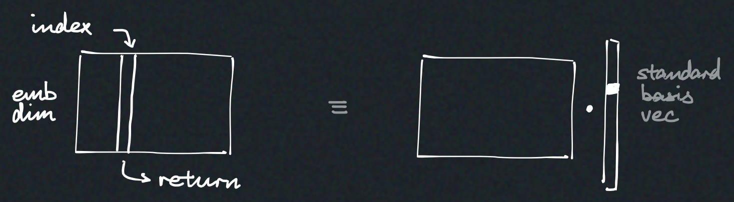
Handles word similarity!

Continuous BOW

I hate this movie



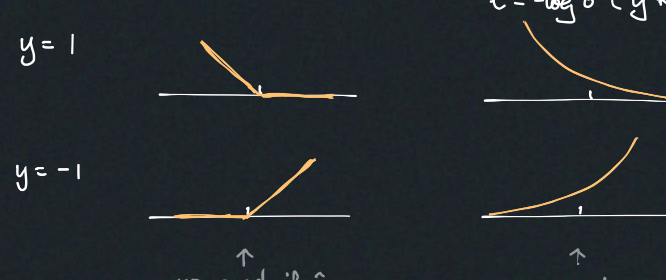
Emb matrix



Training Embeddings — gradient descent

Hinge Loss   Sigmoid + NLL

$l = \max(y * s, 0)$ $\sigma(y * s) = \frac{1}{1 + e^{-(y*s)}}$ ← probability
$l = -\log \sigma(y * s)$

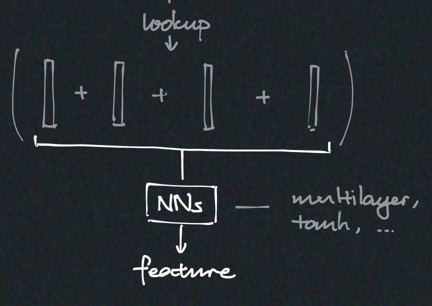


→ then take derivative

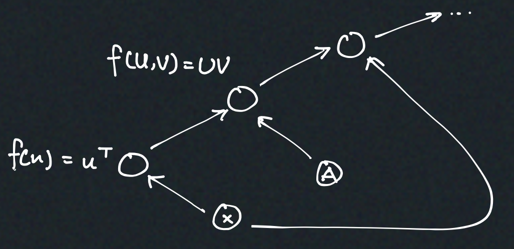
Neural Networds

Handles combined feature
→ Run NN over tkn embs to extract some feature

Deep Continuous BOW (CBOW)   I hate this movie



Computation Graph — DAG



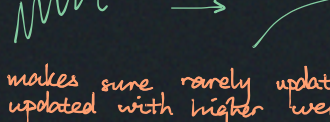
Note: more node — slower

$h = Wx + b$
vs
lin = Linear()
h = lin(x)   ← more optimised

Implementation: - Forward: like work scheduling
- Backward: take derivative in reverse topological order

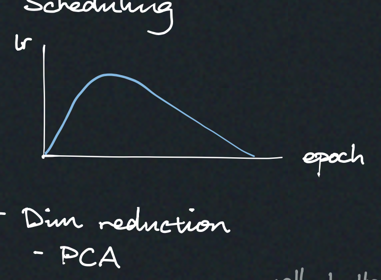
Framework: - PyTorch ← dynamic execution
- Tensorflow] ← define graph + compile
- JAX]

Optimiser: - Adam ← default
Considers momentum & rolling average of gradient varience



makes sure rarely updated params are updated with higher weight

- Scheduling



Visualisation: - Dim reduction
- PCA
- t-SNE ← usually better than PCA but perplexity hparam sensitive

▸ Syntax