Deep Learning and LLMs

Raymond J. Mooney

University of Texas at Austin

Deep Learning Revolution (2010-)

- Recent machine learning methods for training "deep" neural networks (NNs) have demonstrated remarkable progress on many challenging AI problems (e.g. speech recognition, visual object recognition, machine translation, game playing, chat bots)

Very Brief History of Machine Learning

- Single-layer neural networks (1957-1969)
- Symbolic AI & knowledge engineering (1970-1985)
- Multi-layer NNs and symbolic learning (1985-1995)
- Statistical (Bayesian) learning and kernel methods (1995-2010)
- Deep learning (CNNs,RNNs,Transformers) (2010-?)

Deep Learning Revolution (2010…)

- Improved methods developed for training deep neural works.
- Particular successes with:
  - Convolutional neural nets (CNNs) for vision.
  - Recurrent neural nets (RNNs) for machine translation (MT) and speech recognition (ASR).
  - Deep reinforcement learning for game playing.
  - Transformers for MT, LLMs, ChatBots, etc…

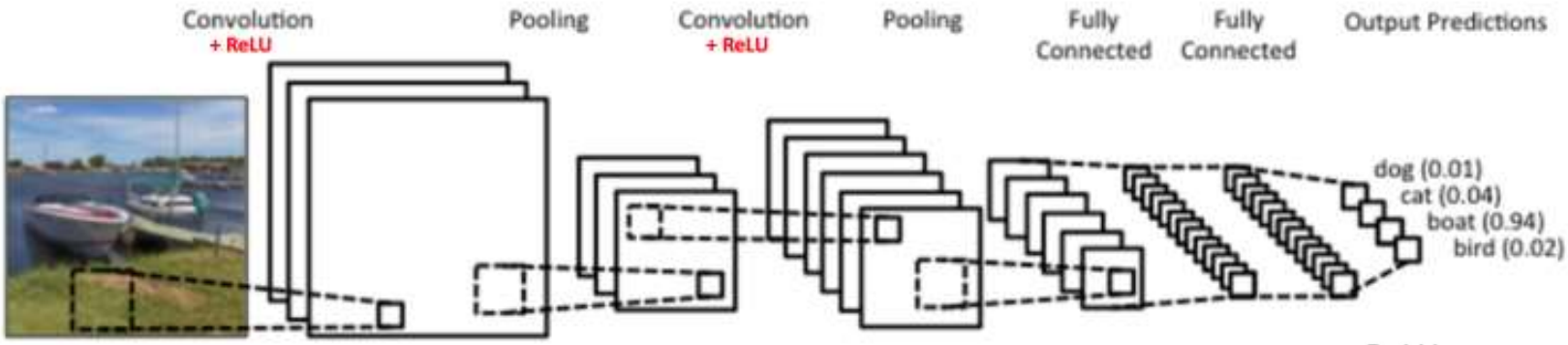
Massive Data and Specialized Hardware

- Large collections of supervised (crowdsourced) training data has been critical.
- Self-supervision (e.g. LMs) on large internet data.
- Efficient processing of this big data using specialized hardware (Graphics Processing Units, GPUs) has been critical.

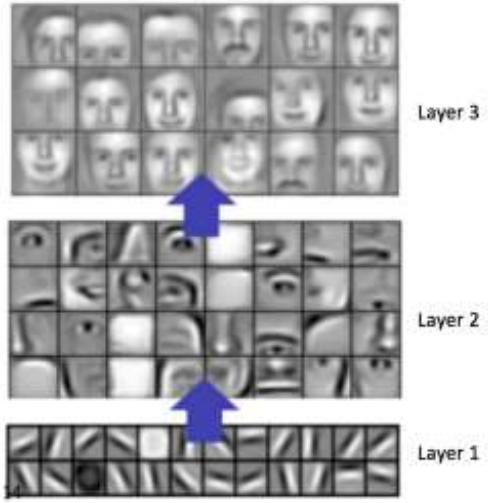
CNNs

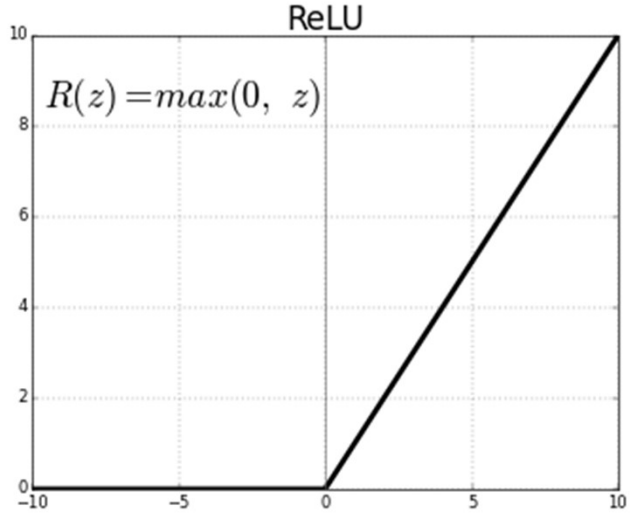
- Convolutional layers learn to extract local features from image regions (receptive fields) analogous to human vision (LeCun, et al., 1998).
- Deeper layers extract higher-level features.
- Pool activity of multiple neurons into one at the next layer using max or mean.
- Nonlinear processing with Rectified Linear Units (ReLUs)
- Decision made using final fully connected layers.

CNNs



Increasingly broader local features extracted from image regions





ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

- Recognize 1,000 categories of objects in 150K test images (given 1.2M training images).

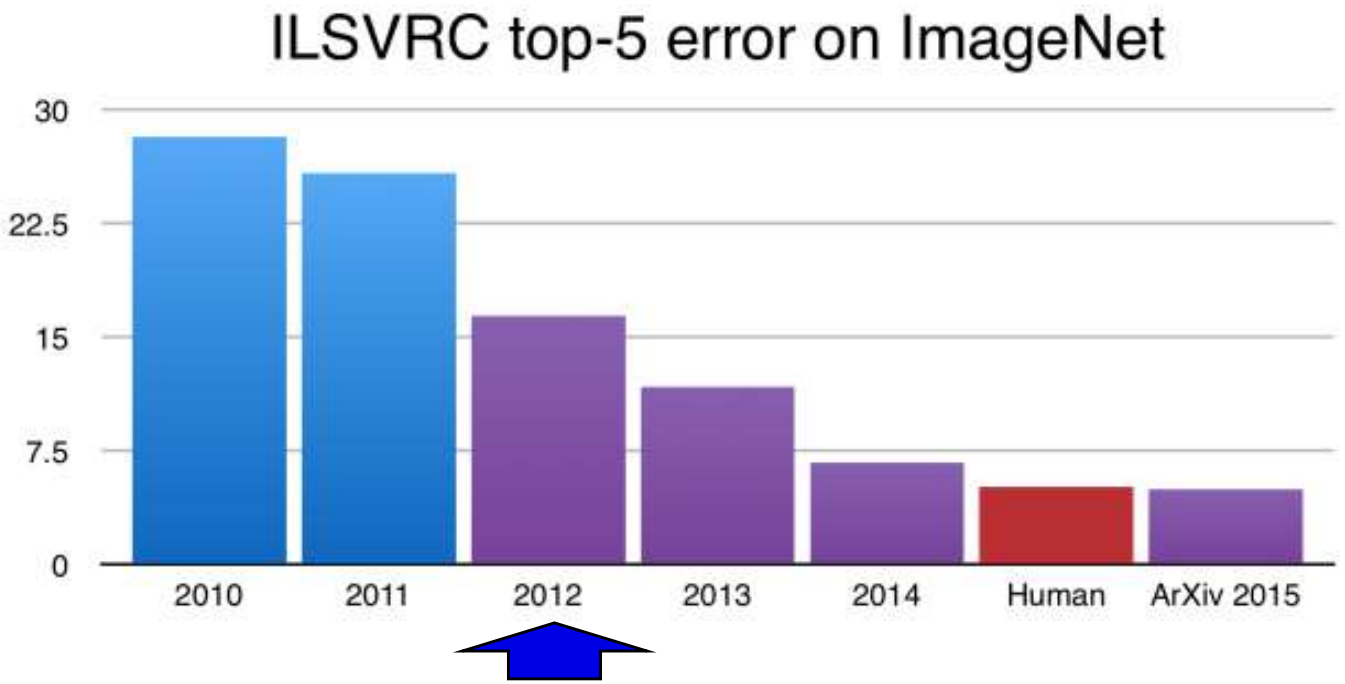
Mongoose

Canoe

Missile

Trombone

ImageNet Performance Over Time



CNNs introduced

Word Embeddings

- Represent words as dense vectors by building representations that capture the context in which they occur, i.e. by the Firth principle:
  - You shall know a word by the company it keeps.
- Semantically similar words will have similar embeddings (close in Euclidian space).

Word Association Matrix

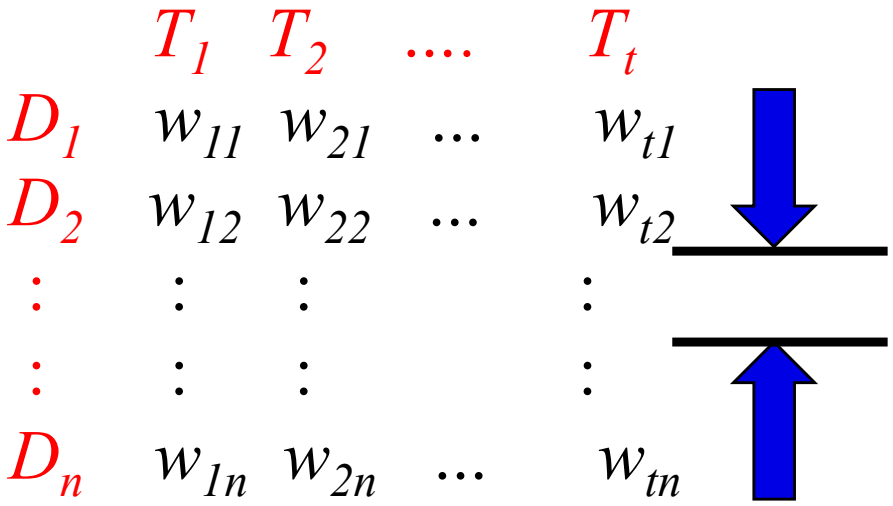
- Remember its early use for automated query expansion.
- Vector for a word is high-dimensional and sparse

	w_1	w_2	w_3	.....	w_n
w_1	c_{11}	c_{12}	c_{13}	.....	c_{1n}
w_2	c_{21}				
w_3	c_{31}				
.	.				
.	.				
w_n	c_{n1}				

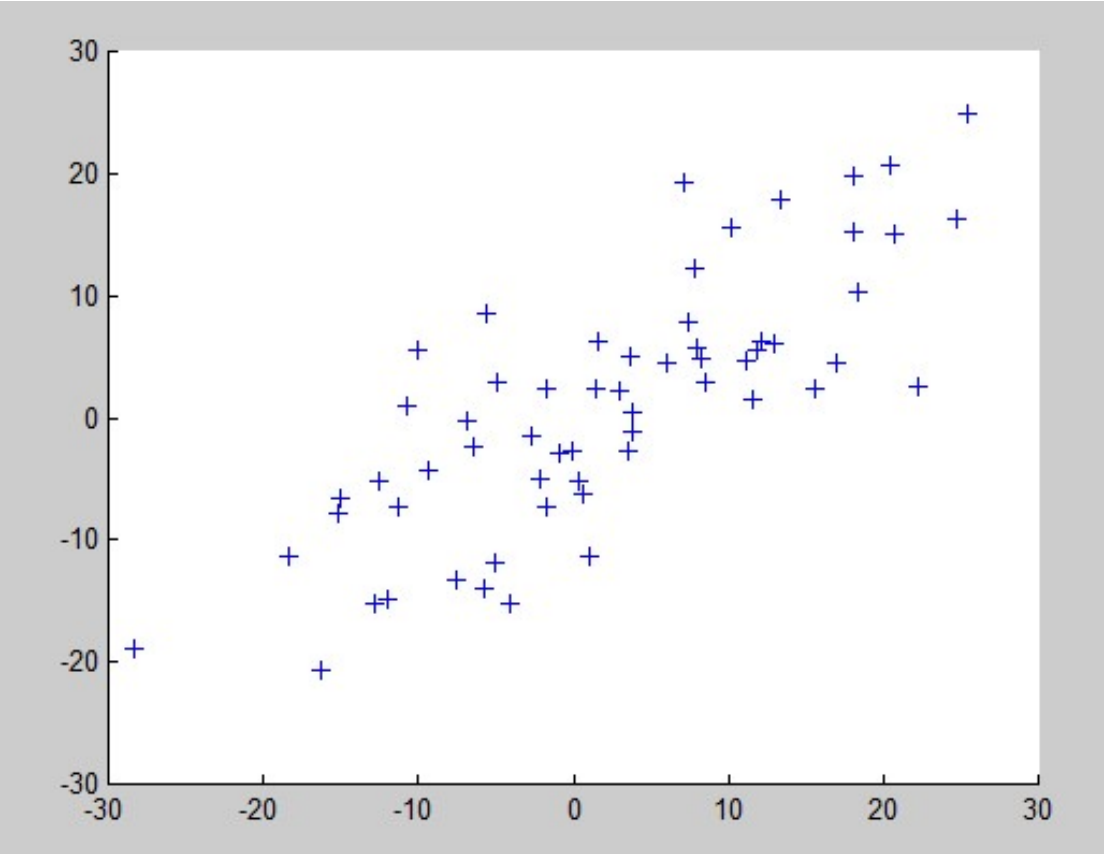
	T_1	T_2	....	T_t
D_1	w_{11}	w_{21}	...	w_{t1}
D_2	w_{12}	w_{22}	...	w_{t2}
:	:	:		:
:	:	:		:
D_n	w_{1n}	w_{2n}	...	w_{tn}

Latent Semantic Analysis (LSA, 1998)

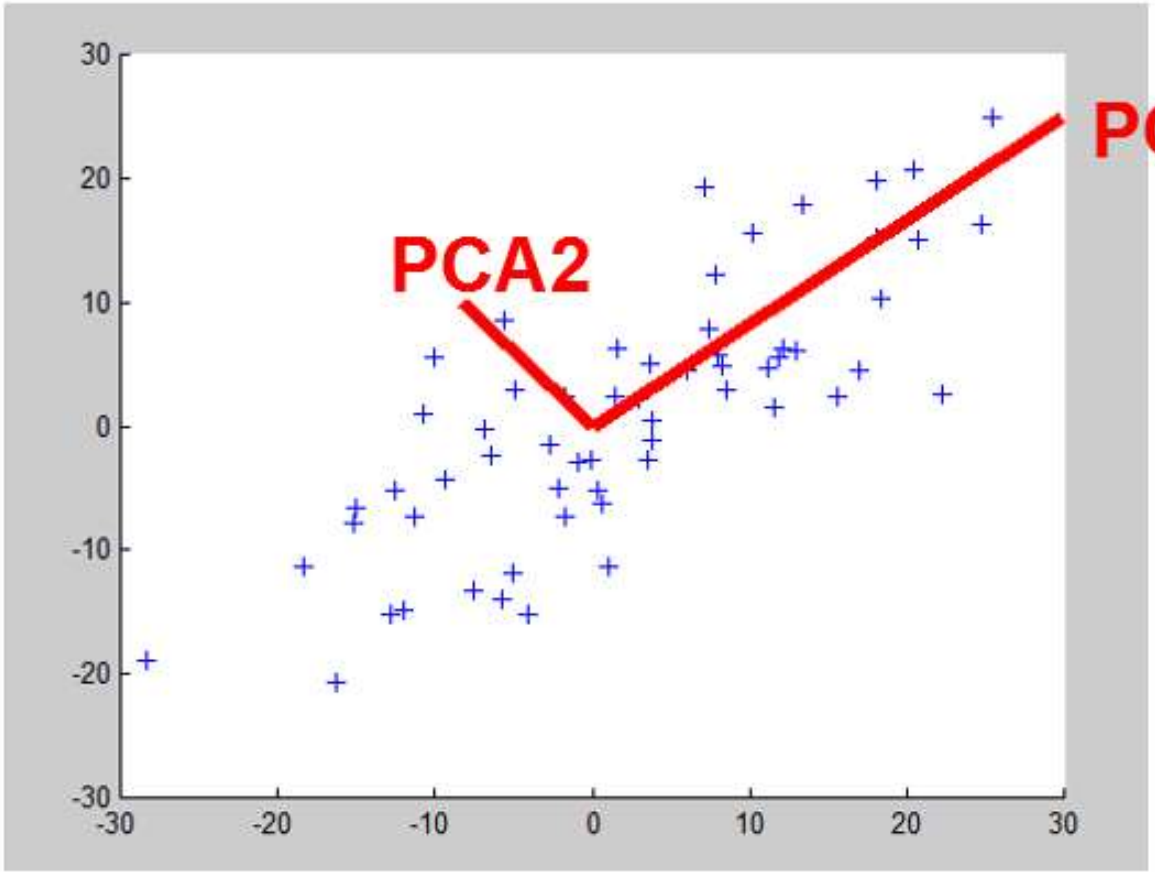
- Use dimensionality reduction methods (Singular Value Decomposition, SVD) on the term-document matrix to compute a dense reduced-dimensional vector for each word.
- Maintain distance between word vectors while reducing their dimensionality.



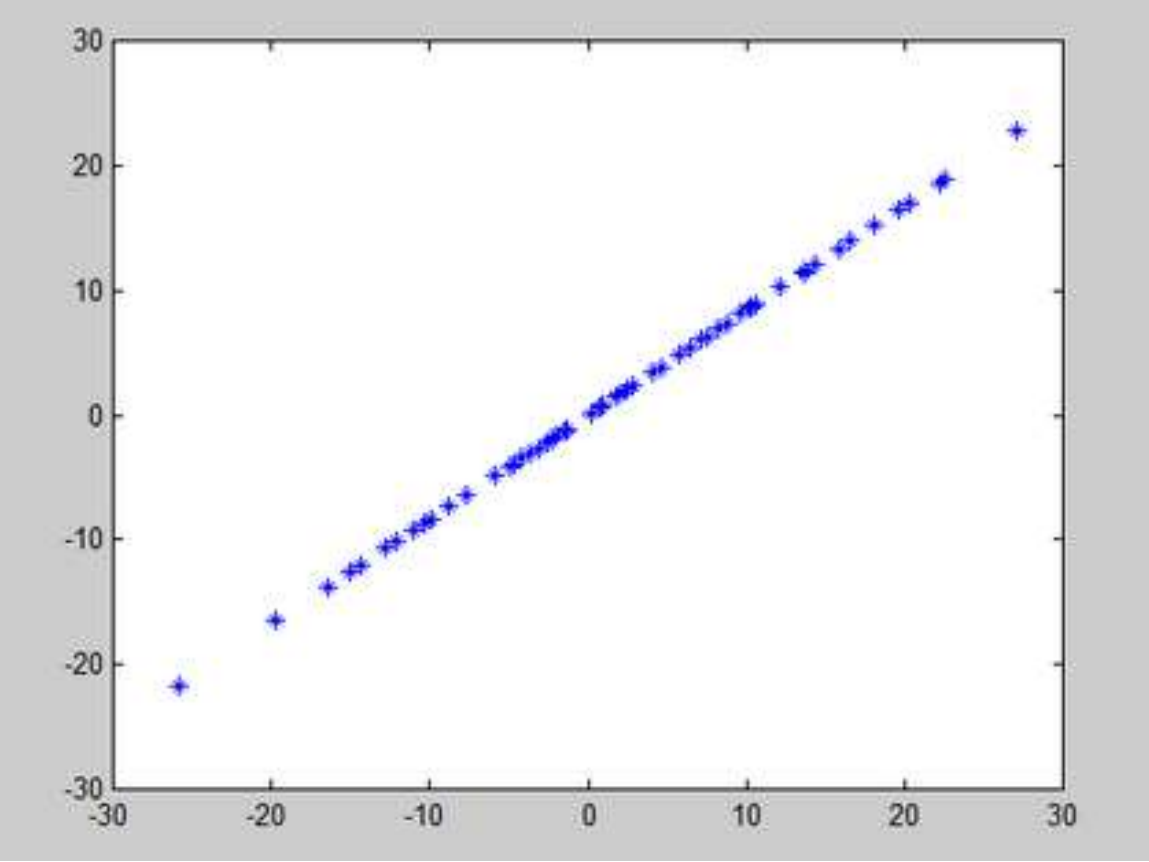
Sample 2 to 1 D Dimensionality Reduction



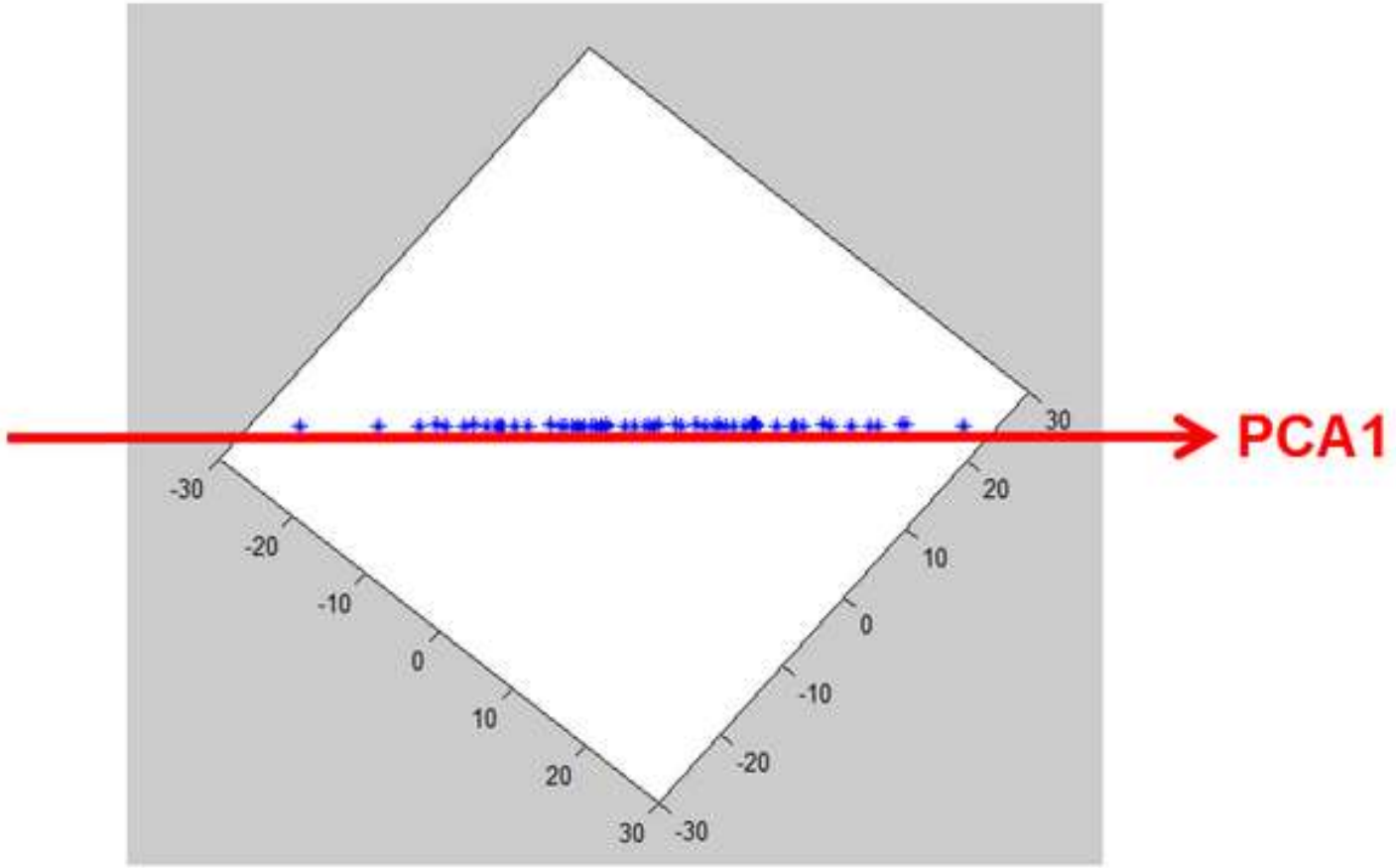
Sample 2 to 1 D Dimensionality Reduction



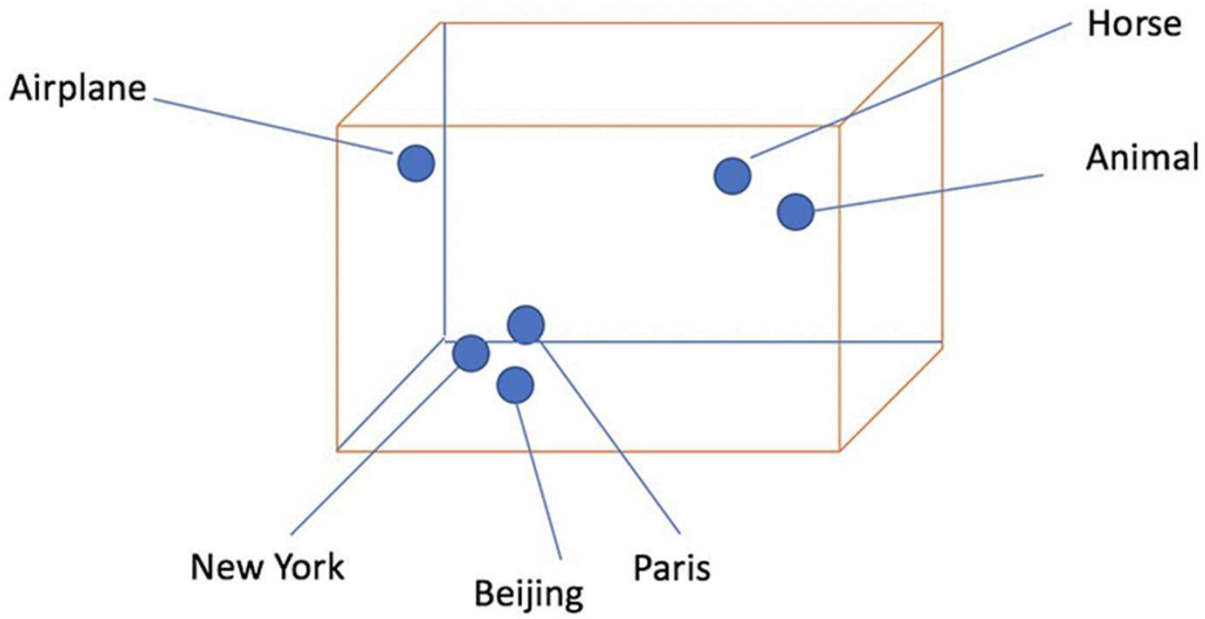
Sample 2 to 1 D Dimensionality Reduction



Sample 2 to 1 D Dimensionality Reduction



Sample Word Embedding Visualization



Neural Net Word Embeddings

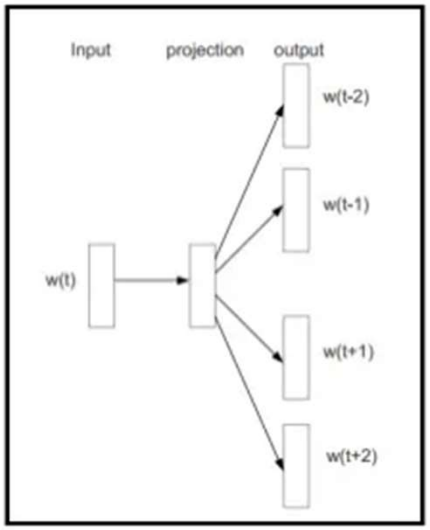
- Train a neural network to predict nearby words for a given word.
- Use the hidden layer representation in this neural network as the embedding.

Word2Vec (Mikolov et al., 2013)

- Input and output words are represented as sparse "one hot" encodings.
- Hidden layer learns a dense encoding of the input word that allows it to predict its neighbors.

Word2Vec

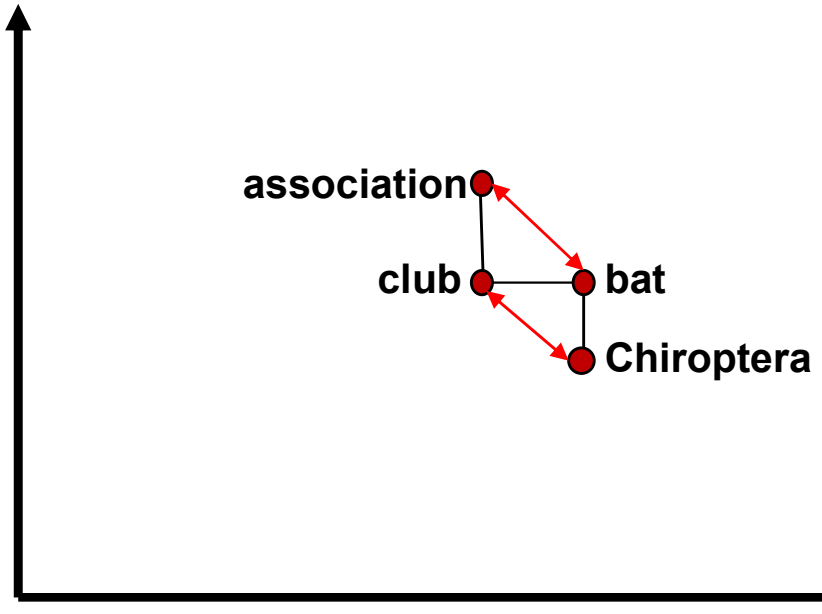
word2vec uses skip-gram neural network to predict neighbor context



Problem with Polysemy and Homonymy

- Word meaning depends on context and having a fixed embedding for a word independent of context is problematic.
- Semantic similarity of words does not obey the "triangle inequality" and therefore cannot be represented as Euclidian distance between word types.

Triangle Inequality Violation Examples



association is similar to club

club is similar to bat

But association is **not** similar to bat

bat is similar to Chirpotera

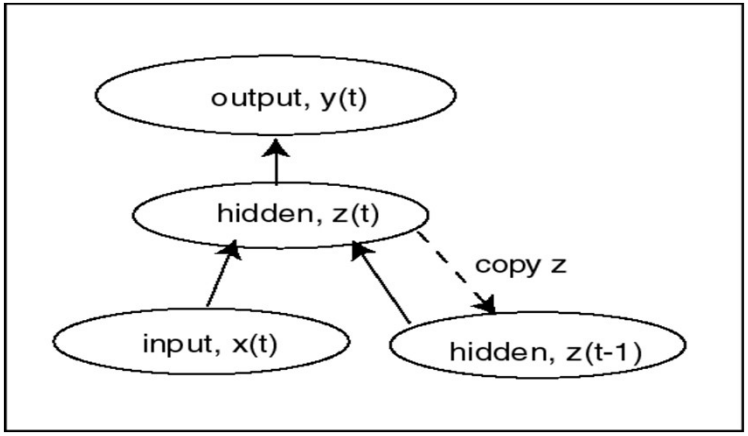
But club is not similar to Chiroptera

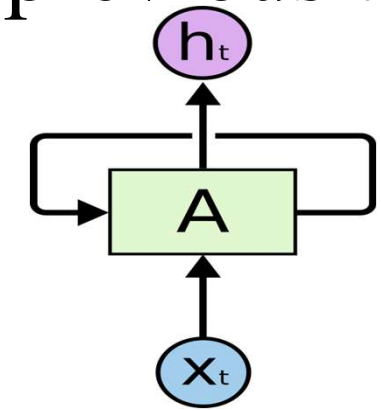
Recurrent Neural Networks (RNNs)

- Add feedback loops where some units' current outputs determine some future network inputs.
- RNNs can model dynamic finite-state machines, beyond the static combinatorial circuits modeled by feed-forward networks.

Simple Recurrent Network (SRN)

- Initially developed by Jeff Elman ("*Finding structure in time*," 1990).
- Additional input to hidden layer is the state of the hidden layer in the previous time

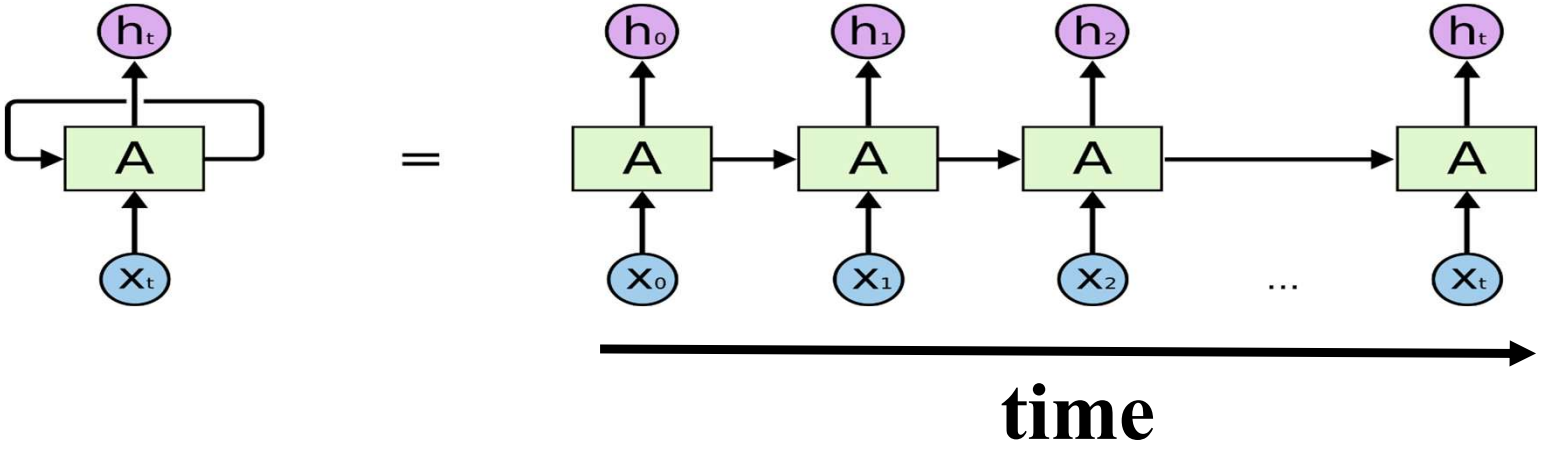




http://colah.github.io/posts/2015-08-Understanding-LSTMs/

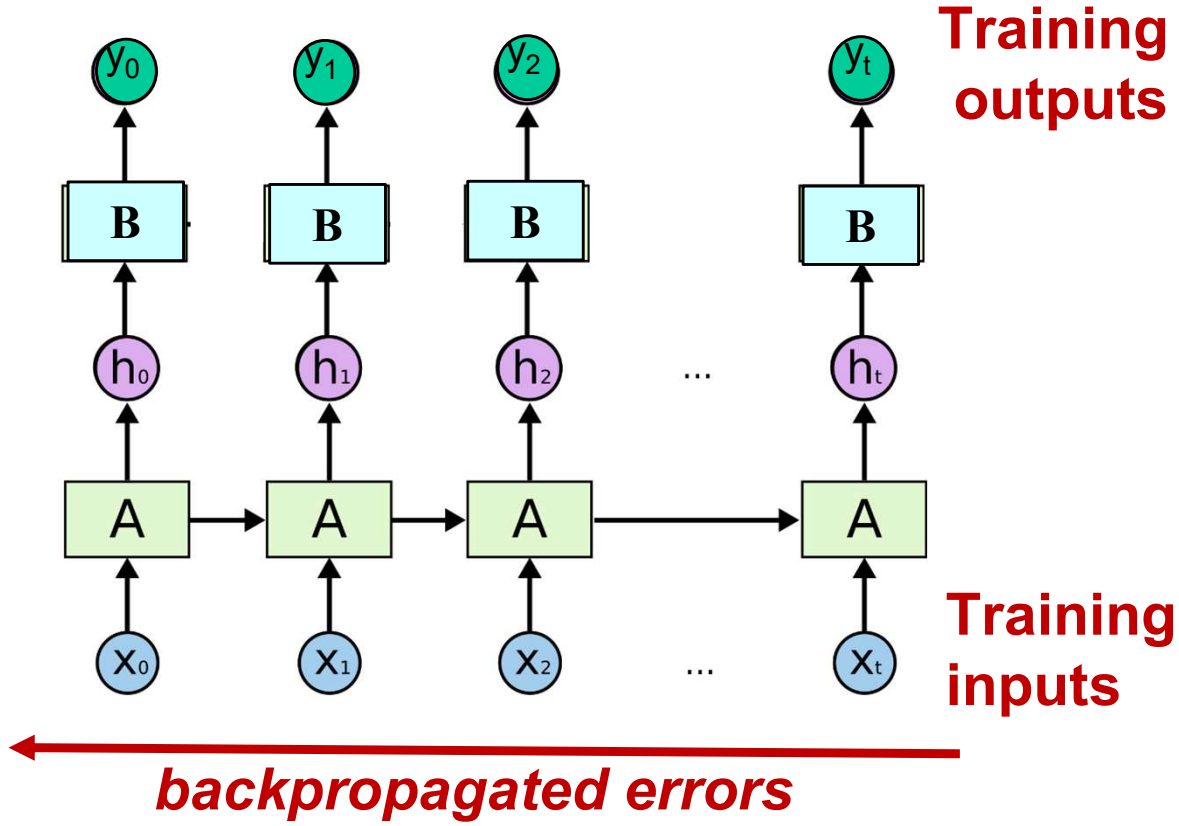
Unrolled RNN

- Behavior of RNN is perhaps best viewed by "unrolling" the network over time.



Training RNN's

- RNNs can be trained using "backpropagation through time."
- Can viewed as applying normal backprop to the unrolled network.

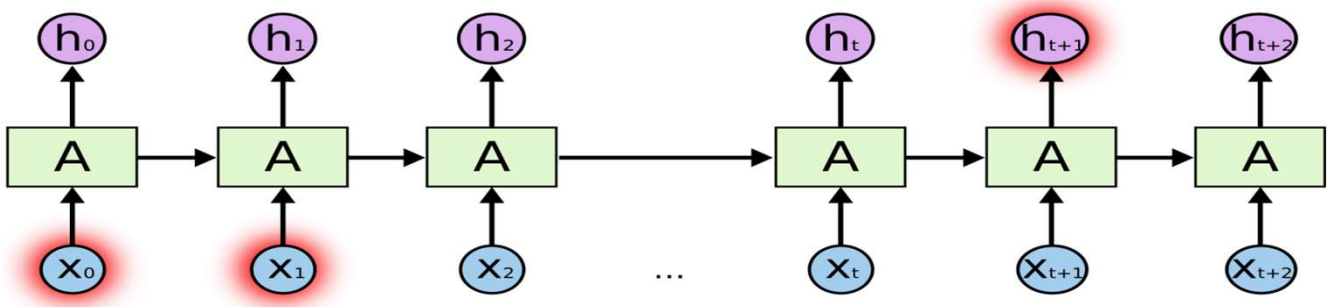


Vanishing/Exploding Gradient Problem

- Backpropagated errors multiply at each layer, resulting in exponential decay (if derivative is small) or growth (if derivative is large).
- Makes it very difficult train deep networks, or simple recurrent networks over many time steps.

Long Distance Dependencies

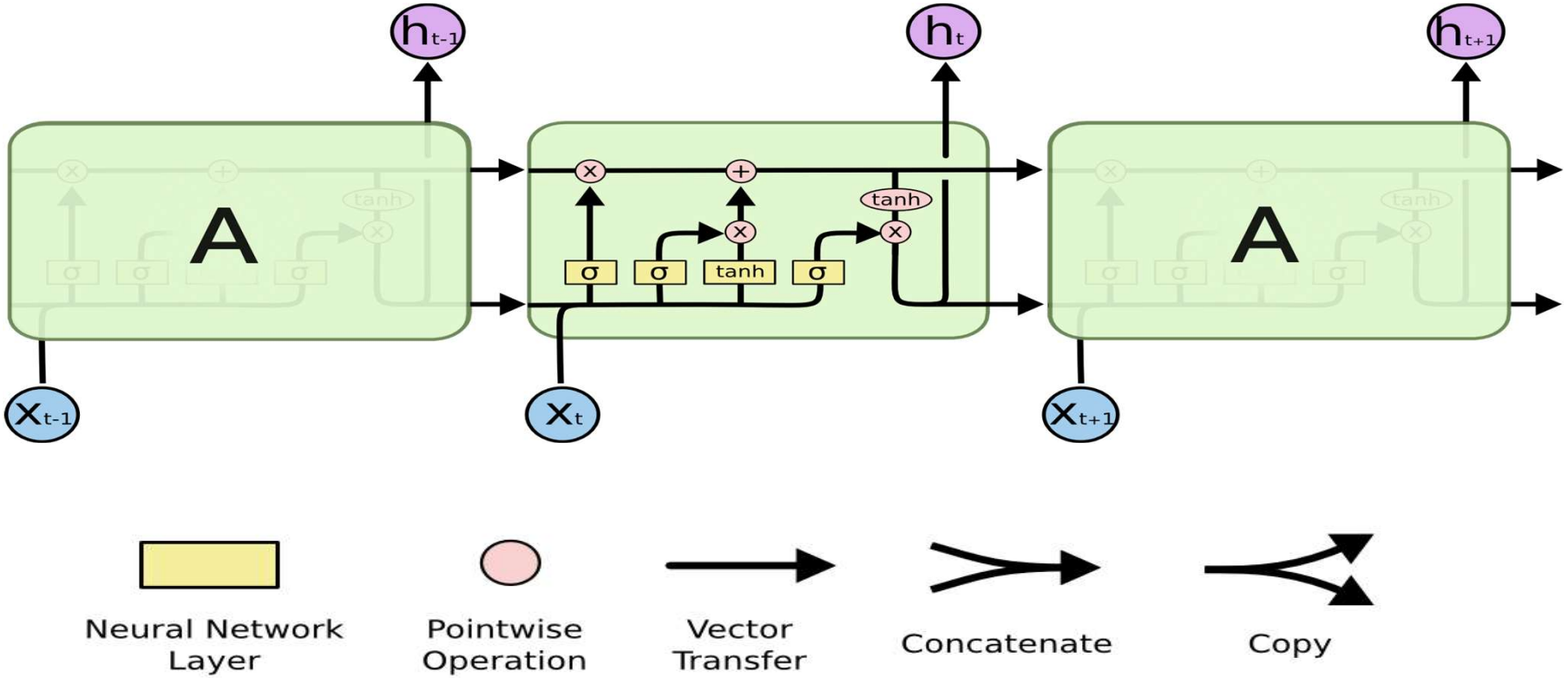
- It is very difficult to train SRNs to retain information over many time steps.
- This make is very difficult to learn SRNs that handle long-distance dependencies, such as subject-verb agreement.



Long Short Term Memory (LSTM)

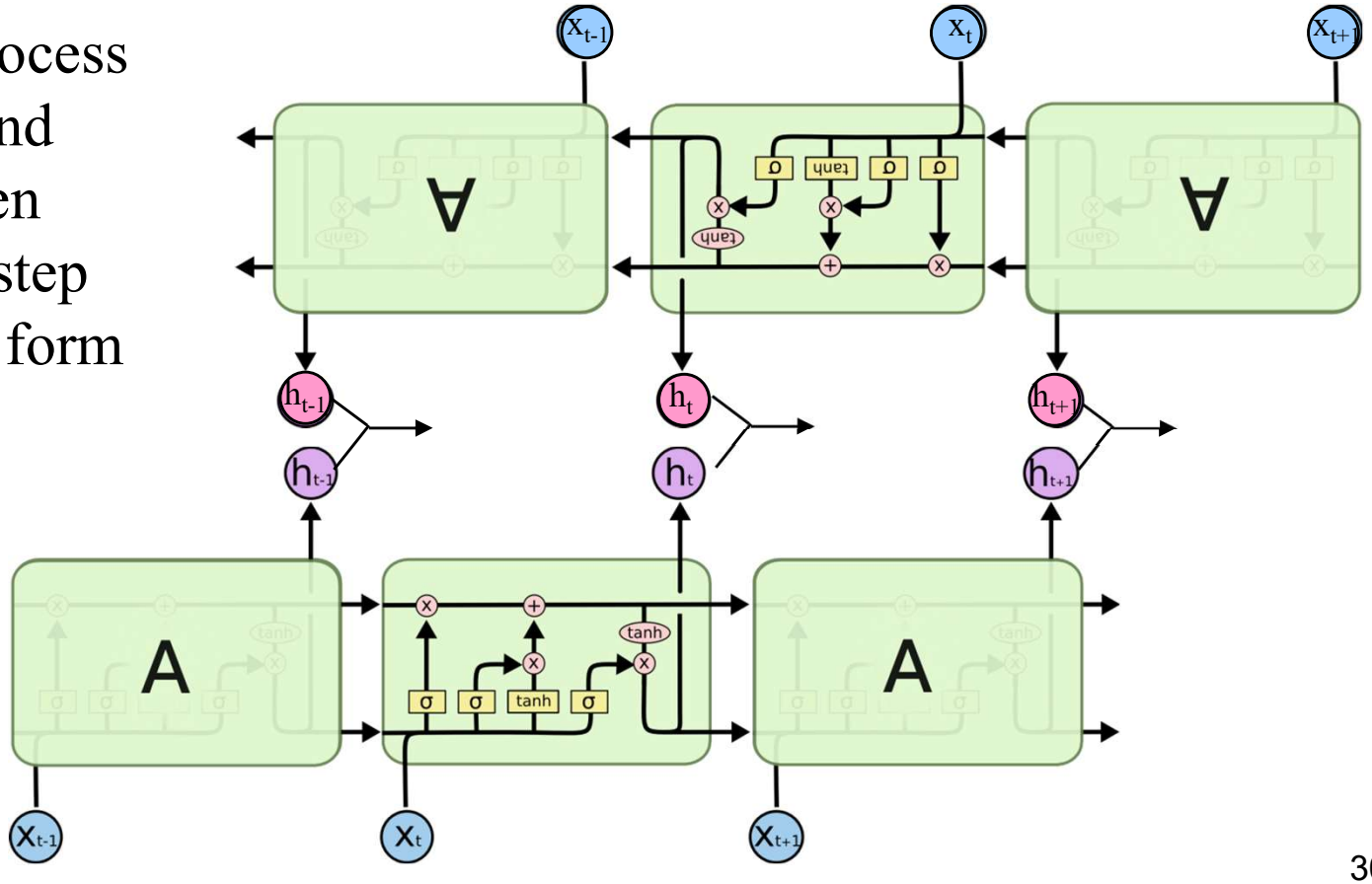
- LSTM networks, add additional gating units in each memory cell (Hochreiter & Schmidhuber, 1997).
  - Forget gate
  - Input gate
  - Output gate
- Prevents vanishing/exploding gradient problem and allows network to retain state information over longer periods of time.

LSTM Network Architecture



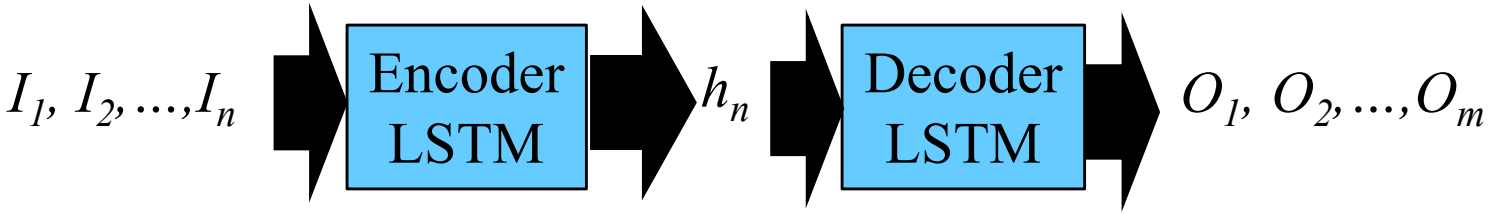
Bi-directional LSTM (Bi-LSTM)

- Separate LSTMs process sequence forward and backward and hidden layers at each time step are concatenated to form the cell output.



Sequence to Sequence (Seq2Seq) Transduction

- Encoder/Decoder framework maps one sequence to a "deep vector" then another LSTM maps this vector to an output sequence (Sutskever et al., 2014).



- Train model "end to end" on I/O pairs of sequences.

Neural Machine Translation (NMT)

- LSTM Seq2Seq has led to a new approach to translating human language.
- NMT modestly outperforms previous statistical learning approaches to MT (SMT).

NMT Results (Wu et al., 2016)

- Experimental results using automated (BLEU) and human evaluation for English→ French translation.

Method	BLEU	Human Rating
SMT	37.0	3.87
NMT	40.35	4.46
Human		4.82

LSTM Application Architectures

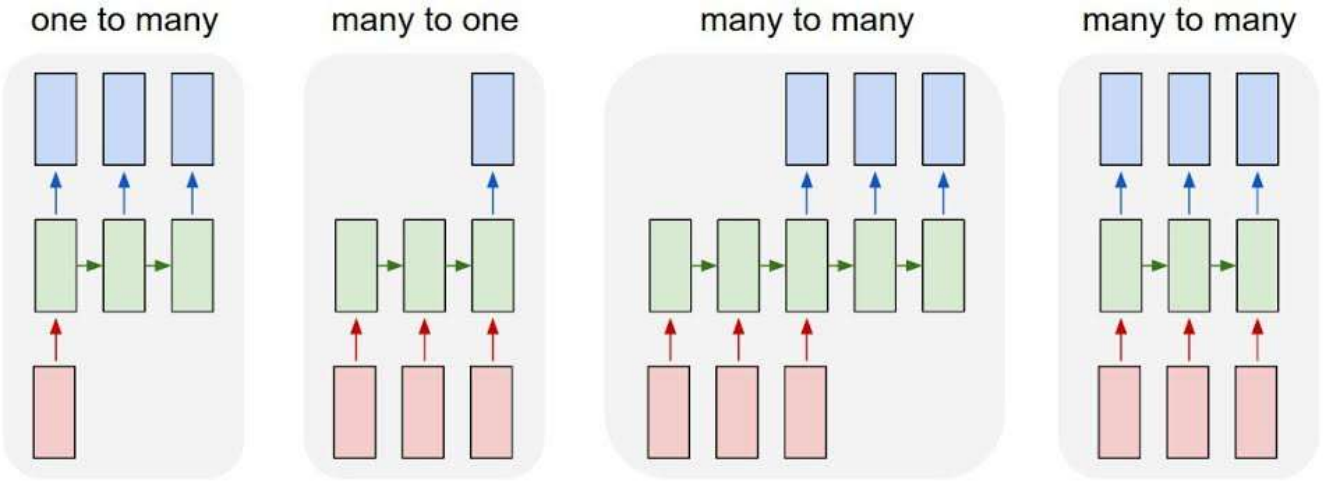


Image Captioning

Video Activity Recog
Text Classification

Video Captioning
Machine Translation

POS Tagging, IE,
Language Modeling

Bidirectional Language Model

- A standard statistical language model predicts the probability of the next word based on the previous context.
  - Your program for Project 4 does not _____
- A bidirectional language model (BiLM) predicts the word at each position based on both prior and posterior context encoded using an RNN (e.g. LSTM).

Contextualized Word Embeddings

- Produce a vector representation for a specific occurrence of a word, by using textual context to compute its meaning.
- ELMo (Embeddings from Language Models, Peters et al., 2018) uses the hidden state of a BiLM to compute contextualized word embeddings.

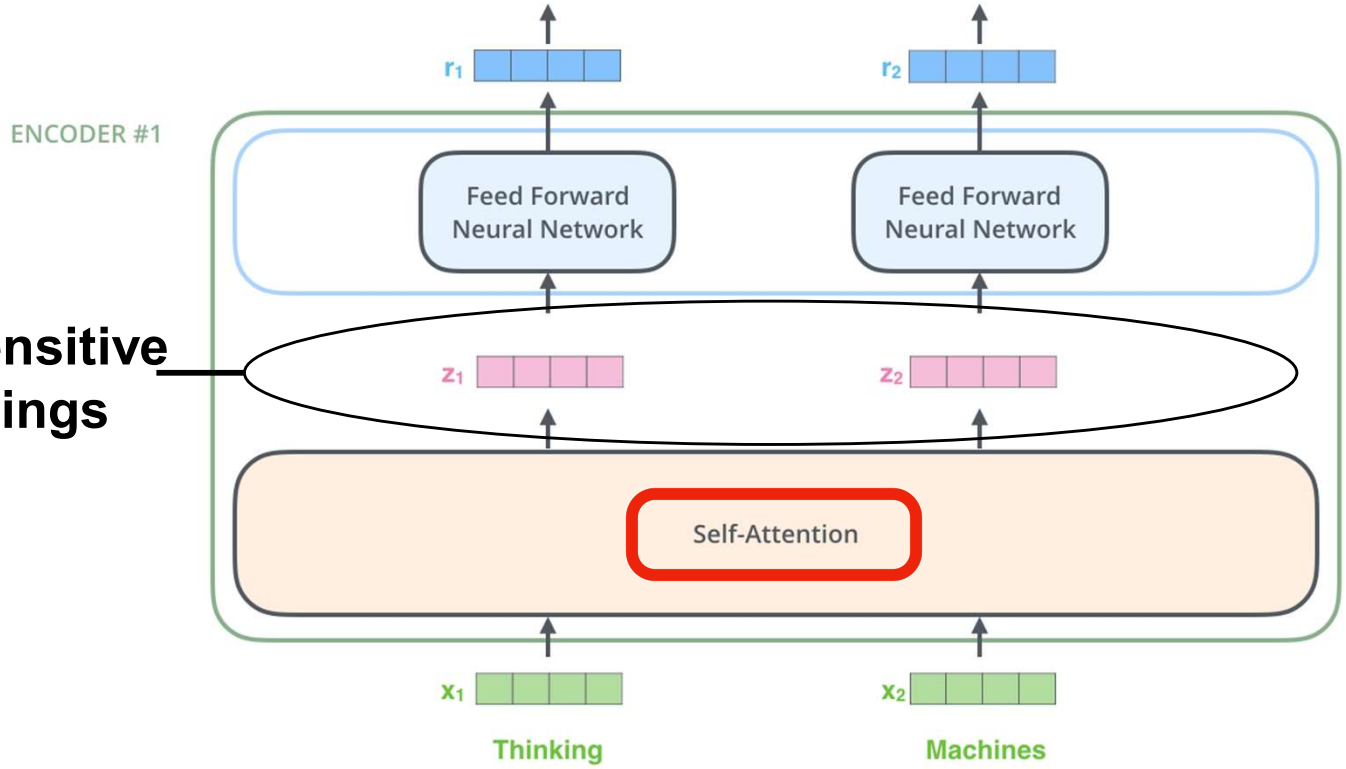
Transformer Networks

- An alternate Seq2Seq neural architecture based on attention rather than recurrence “Attention is all you need”(Vaswani et al., 2017).
- See website on “The Ilustrated Transfomer” by J. Alammar for more details.
- Attention mechanisms compute the output at each position in the sequence by varying “attention” across different positions in the input sequence.

Self-Attention

- Compute a context-sensitive embedding for each word as a weighted sum of initial embeddings of all words in the input.

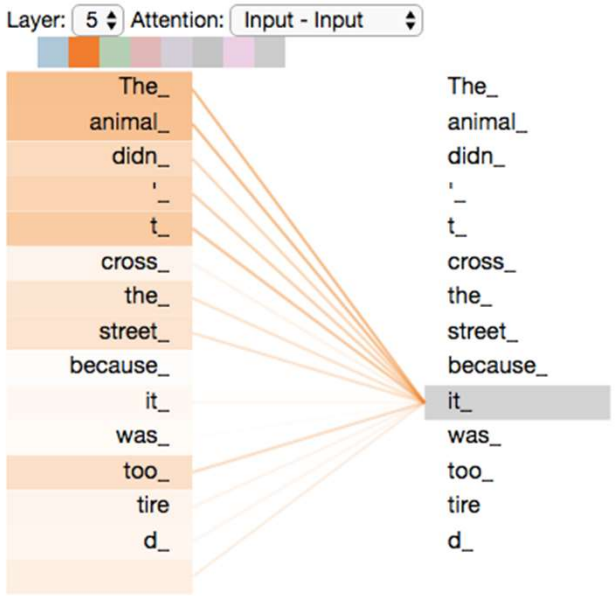
Transformer Encoder



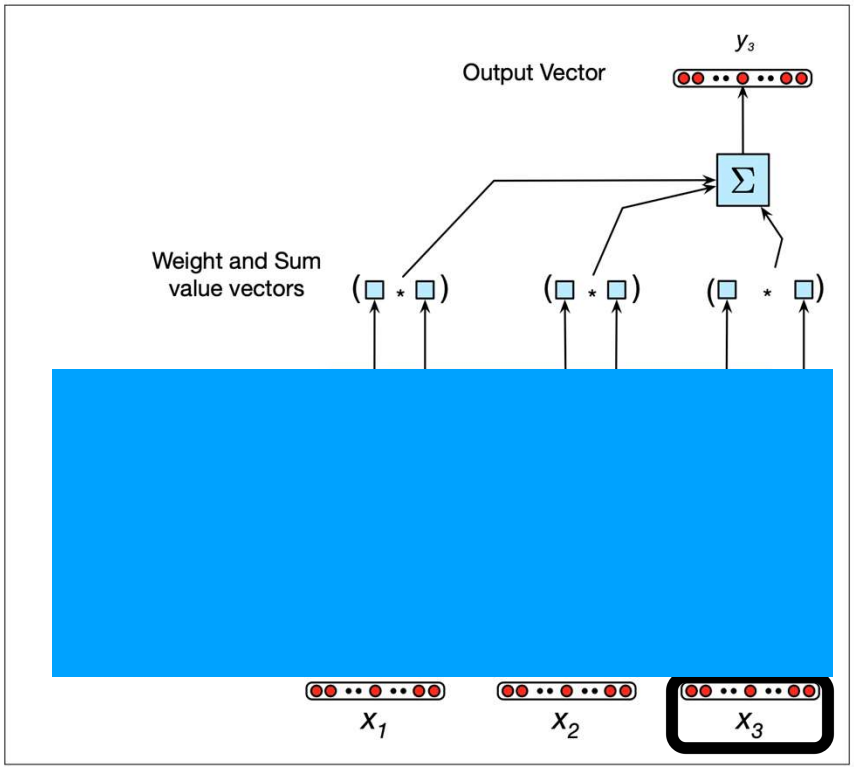
Context sensitive Embeddings

Self-attention

- Consider: “The animal didn't cross the street because it was too tired”
- What meaning should we associate with the word “it”?



Self-attention



$$y_i = \sum_j \alpha_{ij} x_j$$

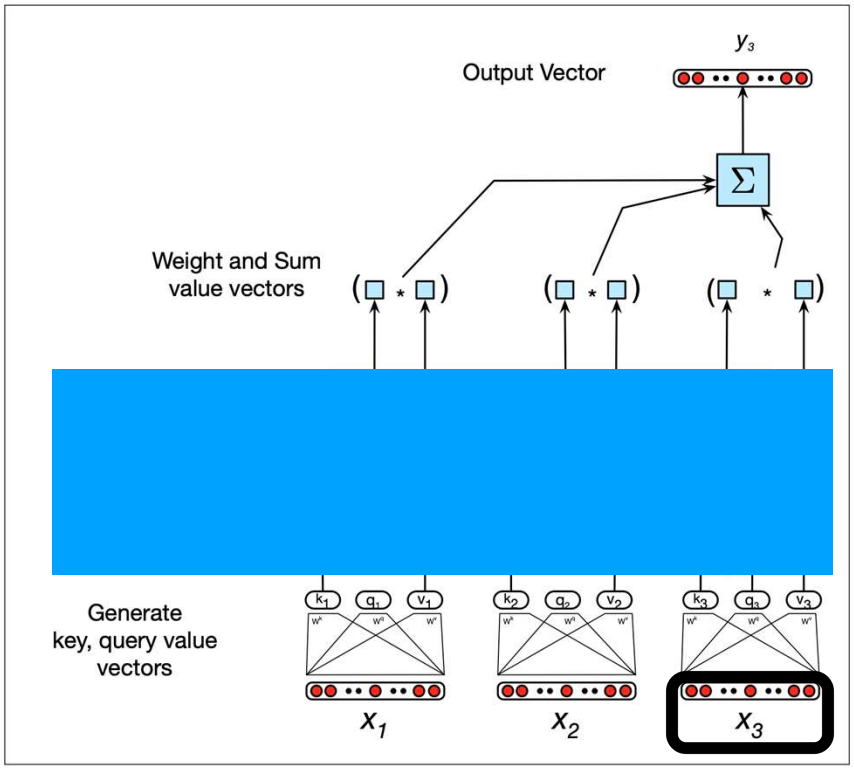
$$\alpha_{ij} = \text{softmax}(score(x_i, x_j))$$

$$= \frac{\exp(score(x_i, x_j))}{\sum_{k=1}^{i} \exp(score(x_i, x_k))}$$

$$score(x_i, x_j) = x_i \cdot x_j$$

But there is nothing to learn here?

Self-attention

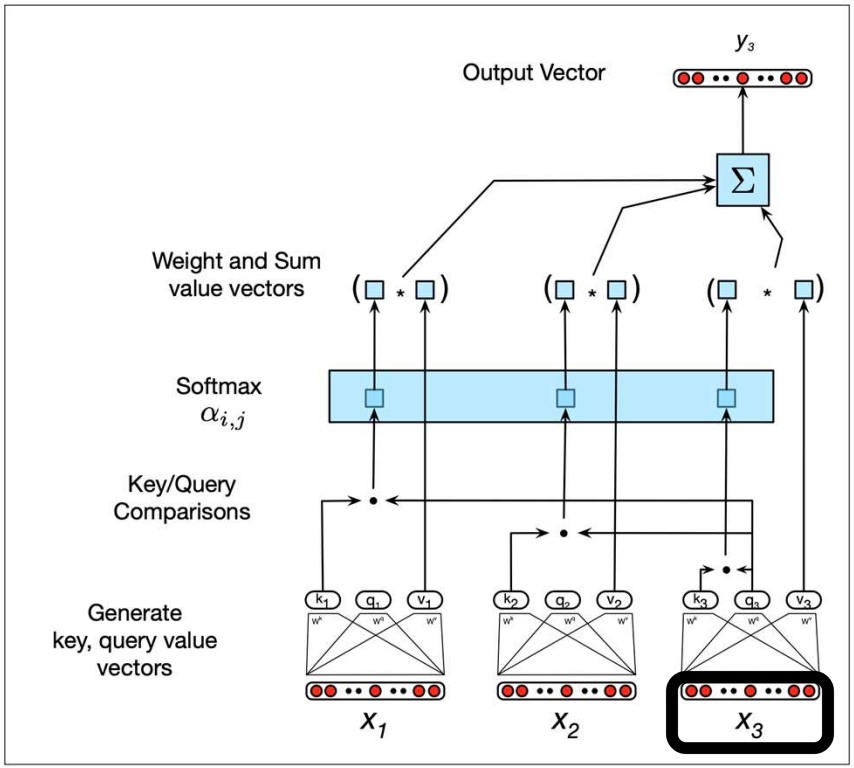


Introducing 3 types of weights, corresponding to 3 roles of each word ***w*:**

- **Query**: w is the current word under question
- **Key**: w is the word in context being compared with
- **Value**: learnable weights for the output

$$q_i = W^Q x_i; \;\; k_i = W^K x_i; \;\; v_i = W^V x_i$$

Self-attention



$$y_i = \sum_j \alpha_{ij} v_j$$

also multiply with v

$$\alpha_{ij} = \text{softmax}(score(x_i, x_j))$$

normalize

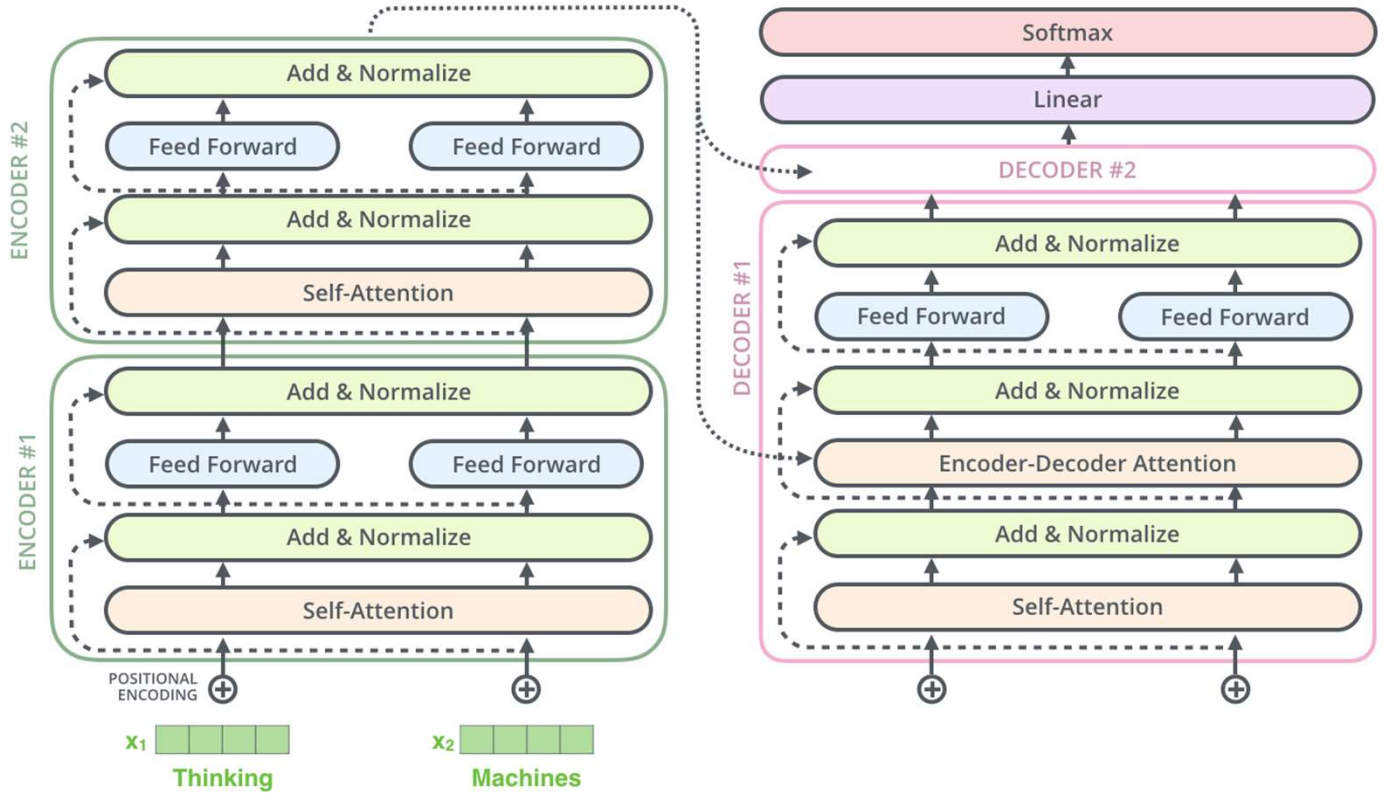
$$score(x_i, x_j) = q_i \cdot k_j$$

$$q_i = W^Q x_i; \;\; k_i = W^K x_i; \;\; v_i = W^V x_i$$

Omitted Transformer Details

- Multiheaded attending
- Positional encoding
- Residual connections and layer normalization

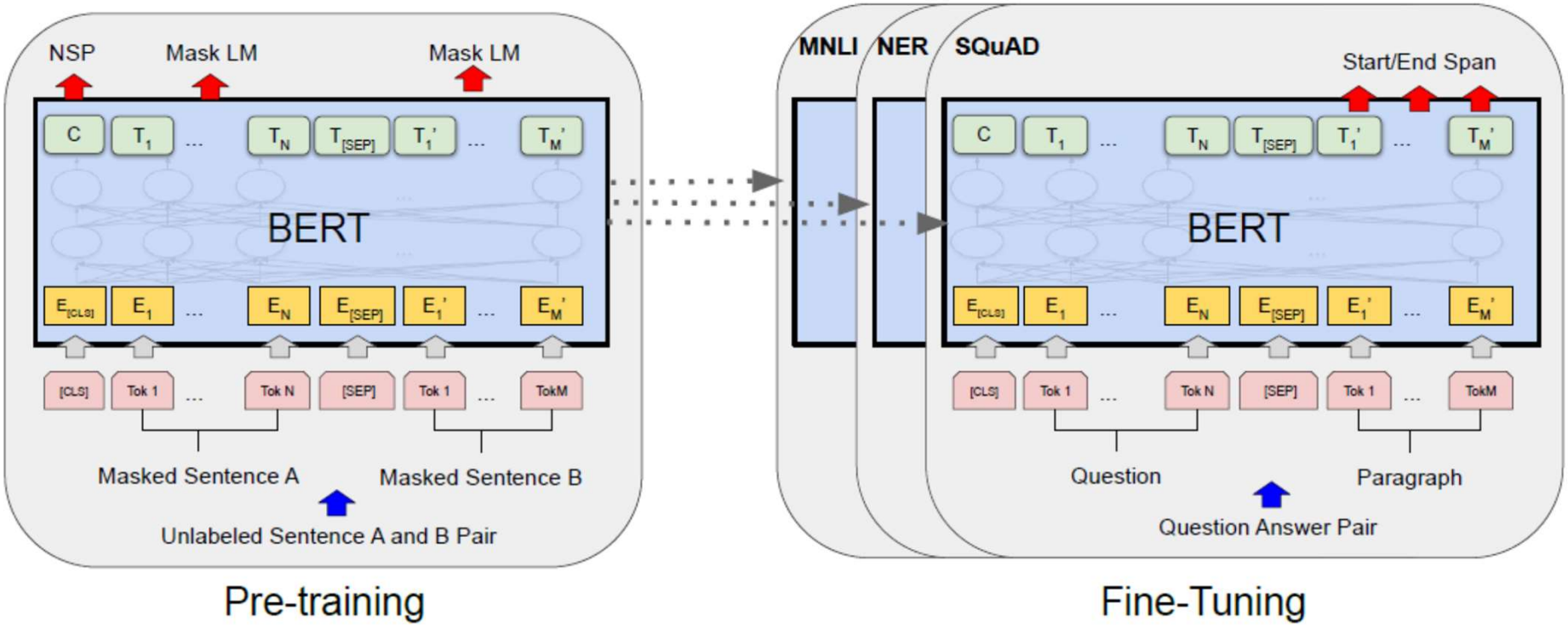
Full Seq2Seq Transformer Architecture



BERT Pretrained Language Transformer

- Bidirectional Encoder Representations from Transformers (BERT, Devlin et al., 2018)
- Trains a transformer network to predict a fraction of “masked” tokens in an input sentence, or predict the next sentence.
- Pretrained network can then be “fine tuned” to perform an end task such as text classification using a full document representation for a special “CLS” token added to the input.

BERT Architecture



Neural Information Retrieval

- Word embeddings have been used to improve IR by allowing matching words based on semantic similarity.
- Can embed entire sentences or documents using an RNN or Transformer and perform vector-based retrieval on these dense vectors rather than sparse BOW vectors.

Nearest Neighbor Retrieval

- Find the document embedding that is most similar to the query embedding using Euclidian distance or cosine similarity.
- Simplest is linear search of the list of documents to find the most similar item.
- Better approach is to use a retrieval index, but an inverted index does not work for dense vectors.

Locality Sensitive Hashing

- A hashing technique that hashes “similar” input items into the same “buckets” with high probability.
- Attempts to maximize hash collisions rather than minimize them.
- Allows for efficient approximate nearest neighbor (ANN) search (see NearPy on gethub).

Conclusions

- Progress on training deep neural architectures such as CNNs, RNNs, and Transformers.
- These techniques can produce dense vector embeddings of words, sentences, and documents.
- Vector-space IR can be based on these deep embeddings rather than sparse BOWs.