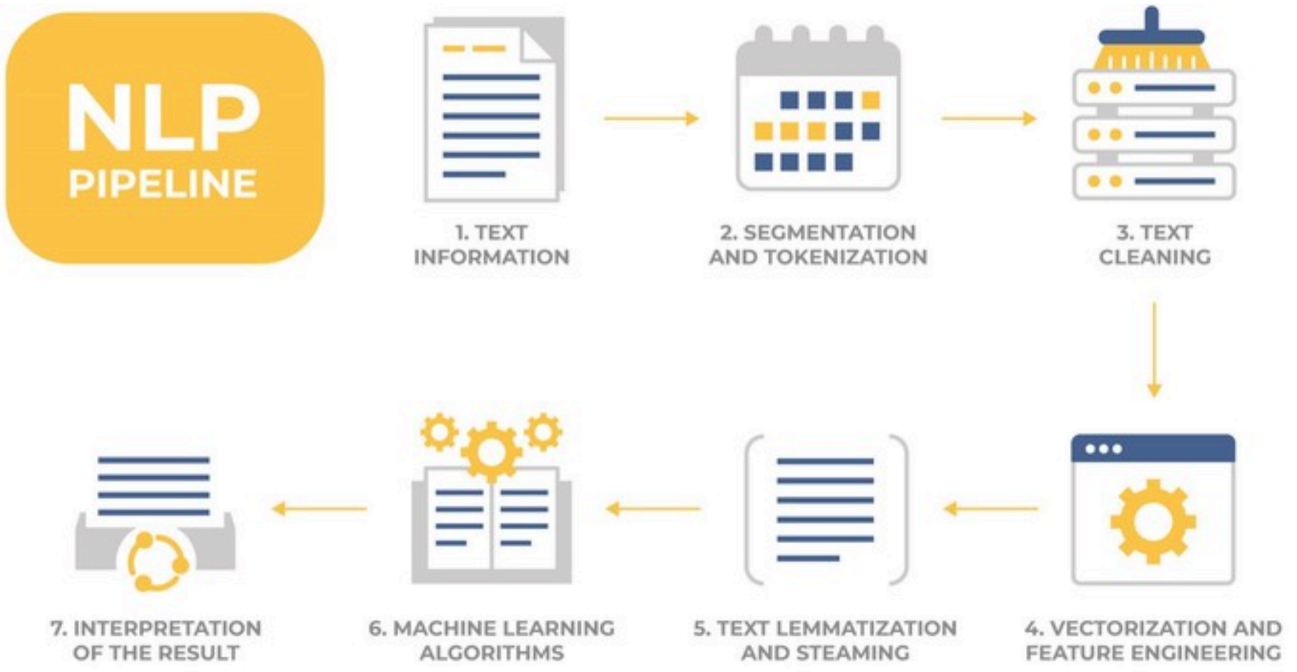
Basics

NLP Tasks

- Machine translation
- Grammar and spell checking
- Text classification
- Named-entity recognition
- Summarization/Topic-modeling
- Text generation



Similarity

- edit distance
- Euclidean distance
- cosine similarty (closer the cosine angle, the more similar words are)

Vectorization (preprocessing)

Process of converting words into numbers

Simple

- Bag of words: dictionary of words with integer indices representing count of the word

- TD-IDF (term frequency and inverse document frequency): estimate relative importance of a term compared to other terms
    - TF: frequency of term in document / total num of terms in document
    - IDF: displays importance of a term $log(n/m)$ where n are the number of documents in a corpus and m are the number of documents containing the term
        - weigh down frequent terms (like I, is, of, that) and scale up rare terms
    - multiply TF and IDF together to get metric

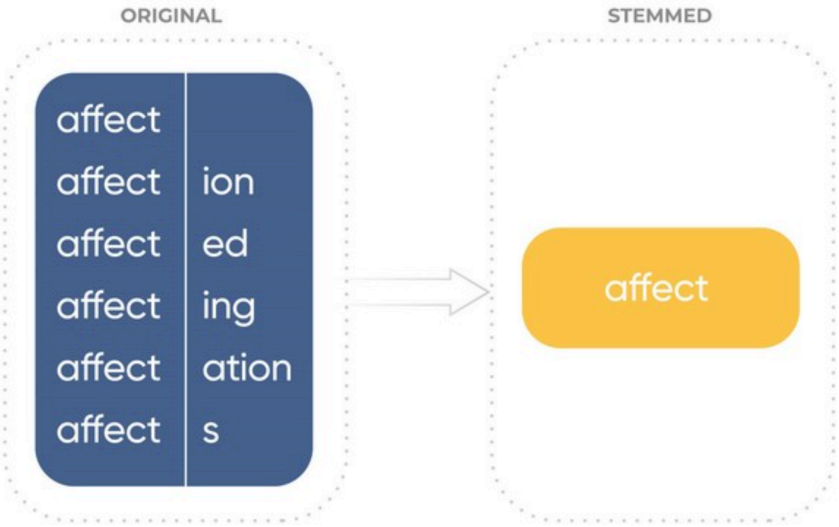
      Purpose: unimportant terms will receive lower significantly lower weight

More Complicated

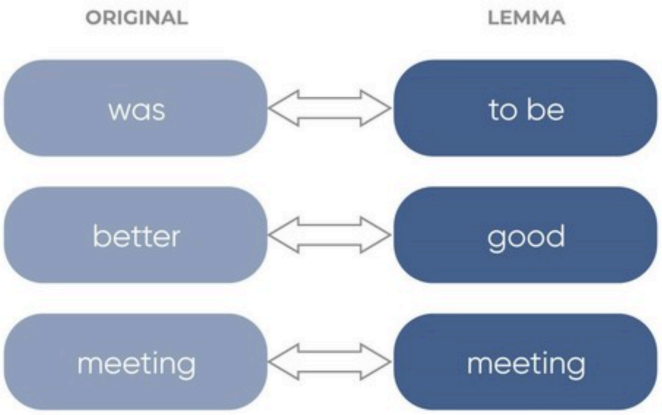
- Word2Vec
    - Uses neural network and sliding window for context
    - Skip-gram and CBow methods
- GloVe (global vectors)
    - Doesn't rely simply on local information, uses co-occurance matrix
    - $J(\theta) = \frac{1}{2}\sum_{i,j}^{n} fP_{(i,j)}(u_i^T v_j - logP_{ij})^2$
        - want to minimizethe difference between the product of word embeddings and log of co-occurence P_{ij}
        - $fP_{(i,j)}$ is the weighted least squares term that gives lower weights to frequent word co-occurrences
- Fasttext
    - Generalize to unseen words (don't have to rely on pretrained dictionary)
    - uses character embeddings

Text Normalization (preprocessing)

- Context-independent normalization: removing non-alphanumeric characters or other symbols
- Canonicalization: convert to "canonical" form
- Stemming: extract word's root
    - reduce words to their root form: often use heuristic that chops off the end of words



- Lemmatization: transform word to its lemma (basic word) using vocabulary, morphological analysis, and parts of speech analysis (using language dictionary)



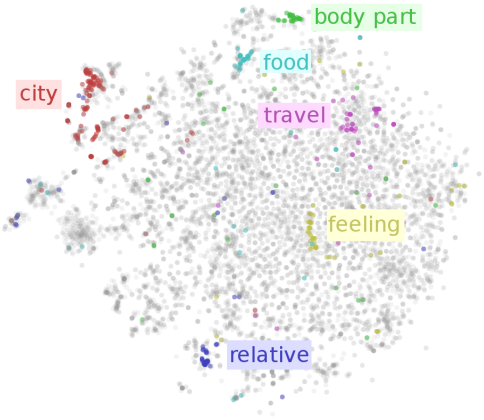
Naive Bayes

- Use bayes theorem with independence assumption between words
  - $arg\max_c[P(c)\sum_{i=1}^{n}P(w_i|c)]$ where w_i are the words and c are the classes
- Algorithm is simple and fast
- Applications
  - Text classification
  - Spam filtering
  - Text tonality analysis

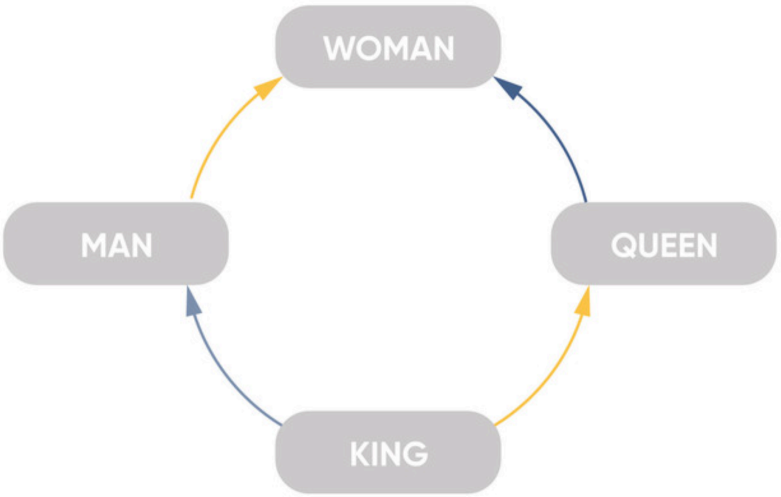
Word Embedding

- Set of various methods that associate words, word forms, or phrases with number vectors

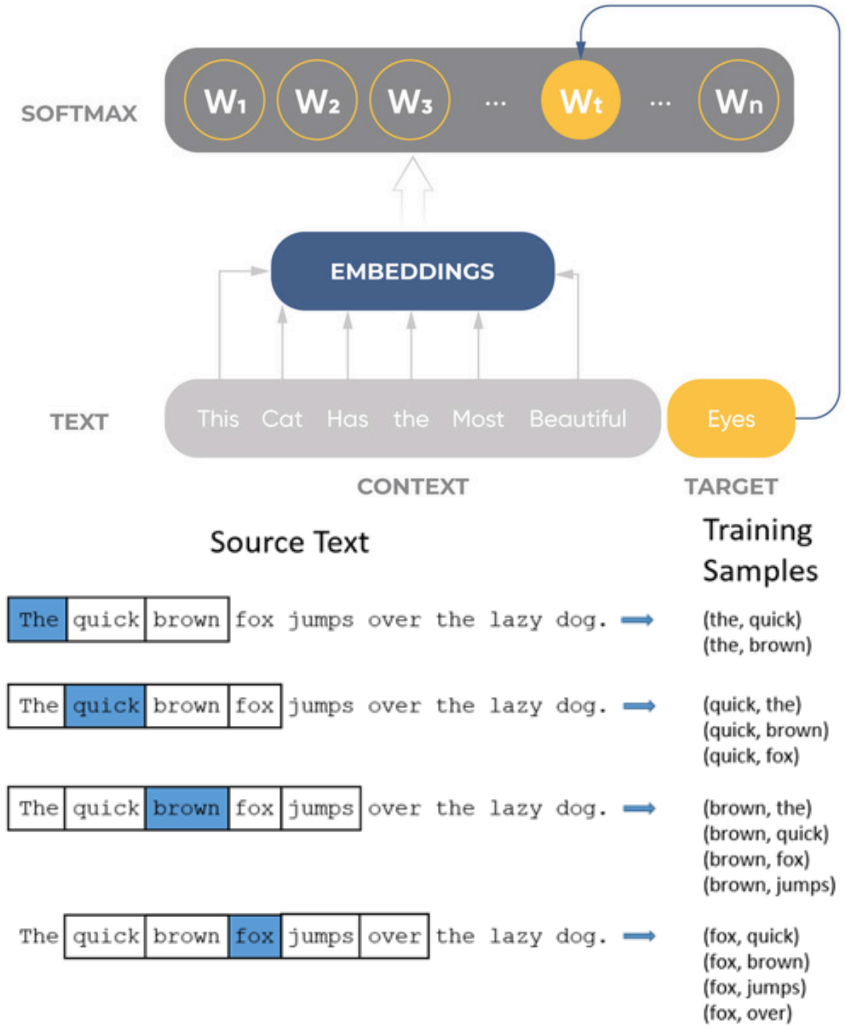
  Words that are similar in meaning/related are closer together



  - Word2Vec: uses neural networks to calculated word embeddings based on context
    - uses one hot encoding
  - TF-IDF: weigh down common words and give more importance to words that only appear in a few documents (sparse vectors)
  - GloVe: uses combination of word vectors that describes probability of co-occurence of words
    - unsupervised method (dense vectors) utilizing neural embeddings (like word2vec)
    - puts emphasis on word-word co-occurences (certain words appear next to each other often)
  - FastText: similar to word2vec but but uses parts and symbols; word becomes context
- Once transformed into embeddings, we can use the vectorized values to measure...
  - similarity (cosine similarity)
  - compare use of word accross different corpora (ex: comparing "immigration" in Democrat/Republican corpus using cosine similarity)
  - analyzing relationships (picture below)
- Embedding



- Word2Vec

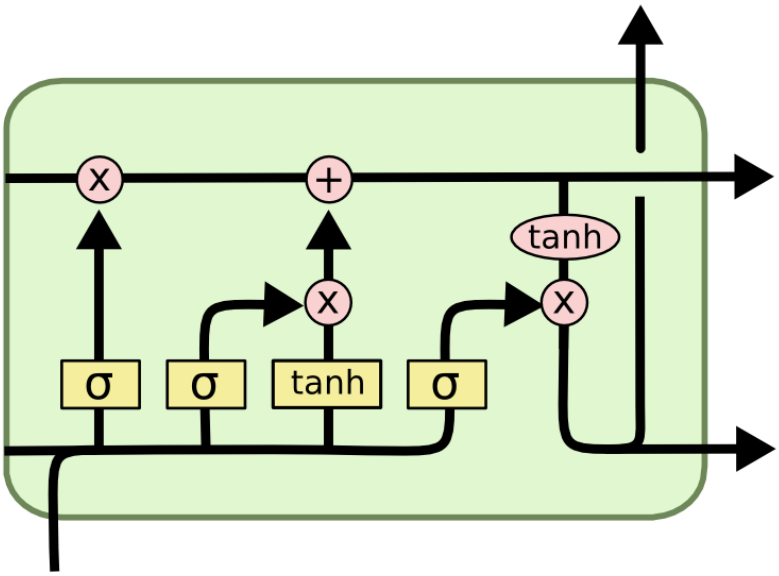


Models

Long Short Term Memory (LSTM)

- NN architecture capable of training long term dependencies; type of recurrent neural network
  - uses input, previous output (or n previous states), and state to determine output
  - addresses short term memory caused by vanishing gradient problem from backprop (weights shared across each time step)
- LSTM gate layers
- forget, storage (input) gate, output
  - cell state
  - input gate layer
  - cell status update (forget gate)
  - output data, makes prediction and also updates hidden state
- Filter layer can remove info from cell state

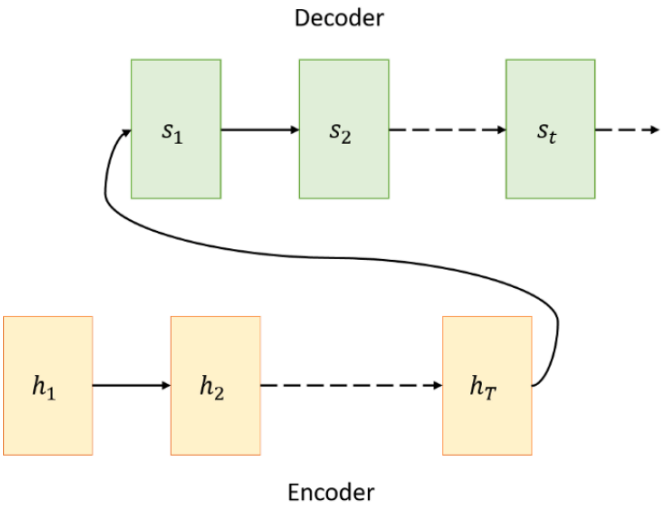
- Variations
  - "Peephole" connections: filter layers can additionally control cell state
  - "Forgotten" and input filters: adds option to ignore or keep in memory information as joint decision by network cells
  - Gated recurrent unit (GRU): the forgetting and input filters combine into one "update" filter (update gate) resulting in simpler and faster LSTM model



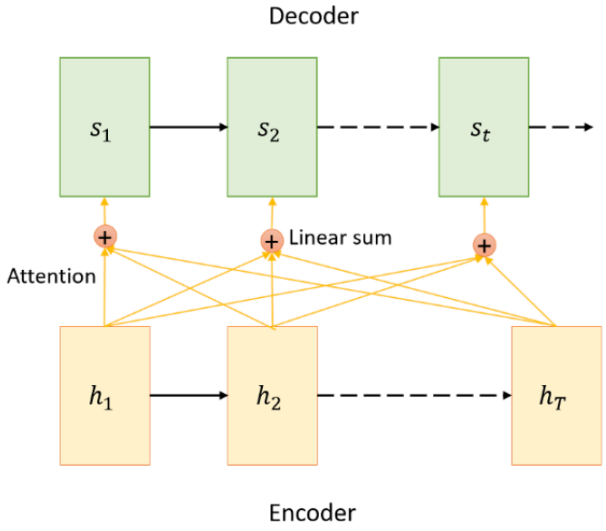
- forget: $\sigma(h_{t-1} + x_t)$
- input: $\sigma(h_{t-1} + x_t) \times tanh(h_{t-1} + x_t)$
  - update: $c_t = f_t \times c_{t-1} + i_t \times \tilde{c}_t$, where $\tilde{c}_t$ is the candidate cell state, f_t, i_t are the forget and input gate outputs
- output: $h_t = \sigma(h_{t-1} + x_t) \times tanh(c_t)$

Seq2Seq: encoder, decoder structure using single-layered RNNs

- potential issue with encoder decoder structure is that the neural network needs to compress all the necessary information of a source sentence into a fixed length vector (from encoder) to be passed to decoder



(a) Vanilla Encoder Decoder Architecture



(b) Attention Mechanism

Vector to Sequence

- Ex: captioning images

Sequence to Vector

- Ex: sentiment analysis

Sequence to Sequence

- Language translation
- Stock prediction

Transformer

Encoder components

1. Input embedding + positional encoding (vector that gives context based on position of word in sentence), can use sin/cos
2. Multi-head attention (average multiple attention vectors)
    1. attention: importance of ith word to current indexed word in vector form
3. Feed forward nn
    1. run ff nn on all attention vectors
    2. attention blocks independent of each other so parrellization can be utilized

Output: *set of encoded vectors for every word*

Decoder components

1. Output embeding + positional encoding
2. Masked multi-head attention
    1. need masking for decoder attention so that future is not given for present prediction
3. Multi-head attention (encoder, decoder attention)
4. Feed forward nn, linear + softmax

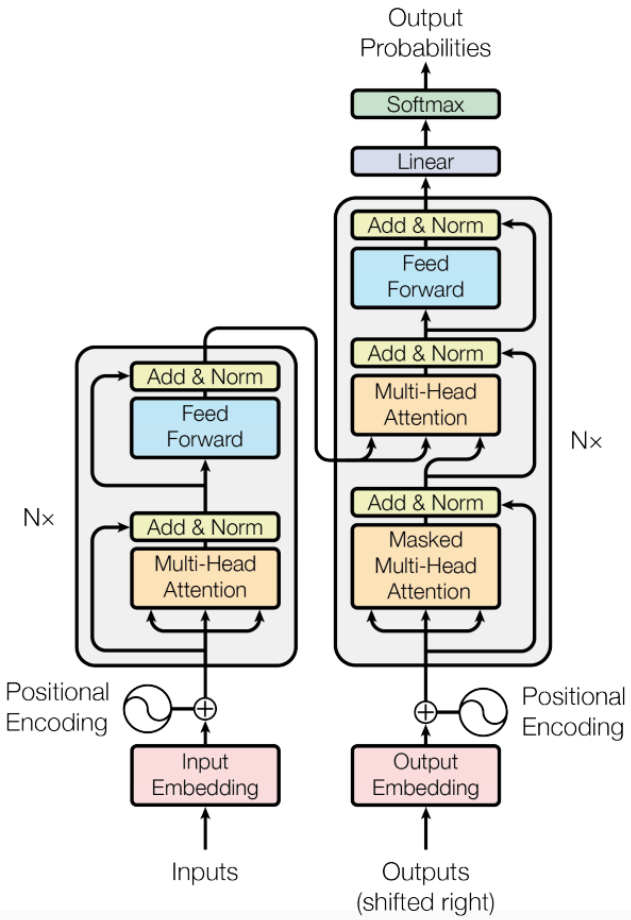


Figure 1: The Transformer - model architecture.

- typically after each layer batch normalization is used (recentering and rescaling and allows for the use of larger learning rates)
- layer normalization (normalization across each feature rather than each sample) can be used, better for stabilization

Papers

- Transformer: "Attention is all you need" (2017): https://arxiv.org/pdf/1706.03762.pdf
- Bidaf (2016): https://arxiv.org/pdf/1611.01603.pdf

GPT paper: https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

LSTMS: https://www.youtube.com/watch?v=8HyCNIVRbSU

Transformer: https://www.youtube.com/watch?v=TQQlZhbC5ps&t=368s

- Query, key, value meaning: https://stats.stackexchange.com/questions/421935/what-exactly-are-keys-queries-and-values-in-attention-mechanisms

RNN, LSTM, Attention: https://medium.com/swlh/a-simple-overview-of-rnn-lstm-and-attention-mechanism-9e844763d07b

https://www.machinelearningplus.com/nlp/natural-language-processing-guide/

https://towardsdatascience.com/a-practitioners-guide-to-natural-language-processing-part-i-processing-understanding-text-9f4abfd13e72

https://www.datasciencecentral.com/profiles/blogs/top-nlp-algorithms-amp-concepts

https://geekyhumans.com/best-nlp-algorithms/